

```

1 /*
2 <span id="gsh" data-title="GShell" data-author="sato@its-more.jp">
3 <meta charset="UTF-8">
4 <meta name="viewport" content="width=device-width, initial-scale=1.0">
5 <link rel="icon" id="GshFaviconURL" href=""><!-- place holder -->
6 <span id="GshVersion" hidden="">gsh-0.8.4-2020-11-17--SatoxITS</span>
7 <title id="GshTitle">GShell-0.8.4 by SatoxITS</title>
8
9 <div id="GshHeading">
10 <div id="GshTopbar" class="MetaWindow"></div>
11 <div id="GshPerfMon"></div>
12 <div id="GshSidebar" class="SbSidebar" style=""><div id="GshIndexer" style=""></div></div>
13 </div>
14 <div id="GshMain">
15 <div id="GshHeader" height="100px" onclick="shiftBG();">
16 <div align="right"><note><a href="http://archive.gshell.org">GShell</a> version 0.8.4 // 2020-11-17 // SatoxITS</note></div>
17 </div>
18 <span id="FeaturesView"></span>
19 */
20
21
22
23
24
25 <!-- ===== Work { ===== -->
26 </span id="CloudChamber_WorkCodeSpan">
27 /*
28 <details id="CloudChamber_Section" open="true"><summary id="CloudChamber_Summary">Cloud Chamber</summary>
29 </----- Cloud Chamber // 2020-1117 SatoxITS { -->
30 <h2>Cloud Chamber</h2>
31
32 <div id="CChamber_1">
33 <h3>Cloud Chamber</h3>
34 <input id="CChamber_1_Clear" class="HtmlCodeViewButton" type="button" value="Clear">
35 <div id="CChamber_1_Sheet" class="CChamberSheet">
36 <div id="CChamber_1_Box" class="CChamber"></div>
37 </div>
38 </div>
39
40 <style>
41 @keyframes CCdot {
42 0% {
43   background-color:rgba(0,0,200,1.0);
44   //background-color:rgba(255,255,255,1.0);
45   background-opacity:1.0;
46   xtop:0px;
47 }
48 20% {
49   background-opacity:1.0;
50   xtop:2px;
51 }
52 80% {
53   background-opacity:0.0;
54   xtop:8px;
55 }
56 100% {
57   background-opacity:0.0;
58   xtop:10px;
59 }
60 }
61 .CChamberSheet {
62   border:1px solid #000;
63   padding:0px;
64   overflow:visible;
65   height:200px;
66   xbackground-color:#202040ff;
67 }
68 .CChamber {
69   xposition:absolute;
70   position:relative;
71   padding:0px;
72 }
73 .CCpoint {
74   display:block;
75   position:absolute;
76   animation-name:CCdot;
77   animation-duration:5s;
78   animation-iteration-count:infinite;
79   border:1px solid #ffffff00;
80   width:30px;
81   height:30px;
82 }
83 </style>
84 <script>
85 var CCpointId = 0;
86 var CCpointMax = 200;
87 var CTrace = [
88   [-20,10],
89   [50,50],
90   [150,70],
91   [250,75],
92   [300,70],
93   [330,65],
94   [360,60],
95   [380,55],
96 ];
97 function addCCpoint(){
98   t = event.target;
99   //console.log('t.id='+t.id);
100   if( t != CChamber_1_Sheet ){
101     return;
102   }
103   //x0 = t.getBoundingClientRect().left.toFixed(0);
104   //y0 = t.getBoundingClientRect().top.toFixed(0);
105   x = event.offsetX; //event.x; // - event.pageX - event.offsetX;
106   y = event.offsetY; //event.y; // - event.pageY - event.offsetY;
107   CTrace[CTrace.length] = [x,y];
108   CCadd(x,y);
109   //console.log('points['+CTrace.length+' ] '+x+', '+y);
110 }
111 function CChamberClear(){
112   while( 0 < CChamber_1_Box.children.length ){
113     CChamber_1_Box.removeChild(CChamber_1_Box.children[0]);
114   }
115 }
116 function CCaddlist(){
117   for( i = 0; i < CTrace.length; i++ ){
118     ccl = CTrace[i];
119     CCadd(ccl[0],ccl[1]);
120   }
121 }
122 function CCadd(x,y){
123   ccp = document.createElement('span');
124   ccp.setAttribute('class','CCpoint');
125   CCpointId += 1;
126   ccp.id = 'ccp_' + CCpointId;
127   ccp.style.left = x + 'px';
128   ccp.style.top = y + 'px';
129   //ccp.style.backgroundColor = 'rgba(0,0,200,1.0)';
130   CChamber_1_Box.appendChild(ccp);
131   while( CCpointMax < CChamber_1_Box.children.length ){
132     CChamber_1_Box.removeChild(CChamber_1_Box.children[0]);
133   }
134 }
135
136 function CChamber_Setup(){
137   CCaddlist();
138   CChamber_1_Sheet.addEventListener('mousemove',addCCpoint);
139   CChamber_1_Clear.addEventListener('click',CChamberClear);
140
141   fi = document.getElementById('FeaturesView');
142   if( fi != null ){
143     fi.appendChild(CChamber_1);
144   }
145 }
146 CChamber_Setup();
147 </script>
148
149 <input id="CloudChamber_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
150 <input id="CloudChamber_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
151 <input id="CloudChamber_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
152 <span id="CloudChamber_WorkCodeView"></span>
153 <script id="CloudChamber_WorkScript">
154 function CloudChamber_openWorkCodeView(){
155   function CloudChamber_showWorkCode(){
156     showHtmlCode(CloudChamber_WorkCodeView,CloudChamber_WorkCodeSpan);
157   }
158   CloudChamber_WorkCodeViewOpen.addEventListener('click',CloudChamber_showWorkCode);
159 }
160 CloudChamber_openWorkCodeView(); // should be invoked by an event
161 </script>
162 </details>
163 <!-- CloudChamber_WorkCodeSpan } -->
164 </span>
165 <!-- ===== Work } ===== -->
166
167
168
169
170
171 <!-- ===== Work { ===== -->
172 </span id="Topbar_WorkCodeSpan">
173 /*
174 <details><summary>Topbar</summary>
175 </----- Topbar // 2020-1008 SatoxITS { -->
176 <h2>Topbar</h2>

```

```

177 <input id="Topbar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
178 <input id="Topbar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
179 <input id="Topbar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
180 <span id="Topbar_WorkCodeView"></span>
181 </details>
182
183 <style>
184 #GshHeading {
185   display:inline;
186   overflow:visible;
187 }
188 .ConfigIcon {
189   position:absolute;
190   top:6px;
191   left:92%;
192   width:32px;
193   height:32px;
194 }
195 .MetaWindow {
196   z-index:1000;
197   position:relative;
198   display:block;
199   overflow:visible !important;
200   width:99.9%;
201   height:22px;
202   top:-22px;
203   border:1px solid #22a;
204   margin:0px;
205   left:0.0%;
206   line-height:1.0;
207   font-family:Georgia;
208   color:#fff;
209   font-size:12pt;
210   text-align:center;
211   vertical-align:middle;
212   padding:4px;
213   xxbackground-color:rgba(0,8,170,0.8);
214   background-color:#3a4861;xxx-PBlue;
215   vertical-align:middle;
216 }
217 .MetaWindow:hover {
218   color:#000;
219   border:1px solid #22a;
220   background-color:rgba(255,255,255,1.0);
221 }
222 #GshBanner {
223   overflow:visible;
224   display:block;
225   width:100%;
226   height:100px;
227   left:inherit !important;
228 }
229 </style>
230 <script>
231 function Topbar_openWorkCodeView(){
232   function Topbar_showWorkCode(){
233     showHtmlCode(#topbar_WorkCodeView,Topbar_WorkCodeSpan);
234   }
235   Topbar_WorkCodeViewOpen.addEventListener('click',Topbar_showWorkCode);
236 }
237 Topbar_openWorkCodeView();
238 function ConfigClick(){
239   if( 0 <= AffView.style.zIndex ){
240     AffView.style.saved_zIndex = AffView.style.zIndex;
241     AffView.style.zIndex = -1000;
242     GshSidebar.style.zIndex = -1;
243     GshPerfMon.style.zIndex = -1;
244   }else{
245     //AffView.style.zIndex = AffView.style.saved_zIndex;
246     AffView.style.zIndex = 1;
247     GshSidebar.style.zIndex = 1;
248     GshPerfMon.style.zIndex = 1;
249     GMenu.style.zIndex = 10000000;
250   }
251   console.log('AffzIndex='+AffView.style.zIndex);
252 }
253 function Gshell_initTopbar(){
254   GshTopbar.innerHTML = GshTitle.innerHTML;
255   //<img id="ConfigIcon" class="ConfigIcon">
256   if( true ){
257     cfigl = document.createElement('img');
258     cfigl.id = 'ConfigIcon';
259     cfigl.setAttribute('class','ConfigIcon');
260     GshTopbar.appendChild(cfigl);
261     cfigl.src = ConfigIcon_DATA;
262     //cfigl.style.zIndex = 10000000000;
263     //cfigl.addEventListener('click',ConfigClick);
264     GshTopbar.addEventListener('click',ConfigClick);
265   }
266 }
267 </script>
268 <!-- Topbar WorkCodeSpan -->
269 *! //</span>
270 //<!-- Work } ----- -->
271
272
273
274 //<!-- Work { ----- -->
275 //<span id="Indexer_WorkCodeSpan">
276 /*
277 <details><summary>Indexer</summary>
278 <!-- Indexer // 2020-1007 SatoxITS { -->
279
280 <input id="Indexer_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
281 <input id="Indexer_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
282 <input id="Indexer_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
283 <span id="Indexer_WorkCodeView"></span>
284 </details>
285 <style id="SidebarIndex">
286 #gsh {
287   display:block;
288   xxoverflow:scroll !important;
289 }
290 #GshMain {
291   z-index:1;
292   position:relative;
293   display:block;
294   width:80% !important;
295   left:19.5% !important;
296 }
297 #GshSidebar {
298   z-index:0;
299   position:relative !important;
300   overflow:auto;
301   resize:both !important;
302   xxoverflow-y:hidden !important;
303   xxheight:100px !important;
304   xxdisplay:inline !important;
305   left:0px;
306   top:0px;
307   width:19.5%;
308   min-width:80px;
309   xxheight:100% !important;
310   height:0px;
311   color:#f00;
312   xxbackground-color:rgba(64,64,64,0.5);
313   xxbackground-color:#DFE3EB;xxx-PBlue;
314   background-color:#eaeaeae;xxx-PBlue;
315 }
316 #GshPerfMon {
317   position:relative;
318   display:block;
319   overflow:visible;
320   z-index:0 !important;
321   xxheight:12pt;
322   font-family:monospace, Courier New !important;
323   font-size:9pt !important;
324   color:#f84;
325   top:-20px;
326 }
327 #GshPerfMon:hover {
328   z-index:3 !important;
329 }
330 #GshSidebar:hover {
331   z-index:2;
332   overflow:xxvisible !important;
333   background-color:rgba(255,255,255,0.7);
334   width:50%;
335 }
336 #GshIndexer {
337   z-index:0;
338   position:relative;
339   resize:both !important;
340   height:100%;
341   left:0px;
342   top:0px;
343   scroll-behavior: overflow !important;
344   padding-left:4pt;
345   font-size:0.5em;
346   white-space:nowrap;
347   xxbackground-color:rgba(64,160,64,0.6) !important;
348   color:#7794c6;xxx-PBlue;
349   xxbackground-color:#DFE3EB;xxx-PBlue;
350   background-color:#eaeaeae;xxx-PBlue;
351 }
352 #GshIndexer:hover {
353   z-index:10000000;

```

```

354 overflow-x:visible !important;
355 color:#000000 !important;xxx-PBlue;
356 xxxbackground-color:#FFFFFF;xxx-PBlue;
357 background-color:rgba(255,255,255,0.7);
358 padding-right:0px;
359 width:80%;
360 }
361 #GshIndexer:select {
362 color:#000000 !important;xxx-PBlue;
363 background-color:#FFFFFF;xxx-PBlue;
364 }
365 .IndexLine {
366 font-size:8pt !important;
367 font-family:Georgia;
368 display:block;
369 xxx-color:#fff;
370 xxx-color:#efff5;xxx-PBlue;
371 xxx-color:#41516d;xxx-PBlue;
372 xxx-color:#7794c6;xxx-PBlue;
373 padding-right:4pt;
374 }
375 .IndexLine:hover {
376 font-size:10pt !important;
377 xxx-color:#222;
378 xxx-background-color:#fff;
379 xxxcolor:#fff;xxx-PBlue;
380 color:#516487;xxx-PBlue;
381 background-color:rgba(220,220,255,1.0);xxx-PBlue;
382 xxxtext-shadow:1px 1px #3f3;
383 text-shadow:1px 1px #eee;
384 xxxbackground-color:#516487;xxx-PBlue;
385 xxxtext-decoration:underline !important;
386 }
387 </style>
388
389 <script id="Indexer WorkScript">
390 function Indexer_openWorkCodeView(){
391 function Indexer_showWorkCode(){
392 showHtmlCode(Indexer_WorkCodeView,Indexer_WorkCodeSpan);
393 }
394 Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_showWorkCode);
395 }
396 //Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_openWorkCodeView);
397 Indexer_openWorkCodeView();
398
399 var startPerfDate = new Date();
400 var prevPerfDate = startPerfDate;
401 function ShowResourceUsage(){
402 d = new Date();
403 perf = '';
404 perf += '<'+'font color="gray">UA:' + window.navigator.userAgent + '<'+'font><br>\n';
405 perf += DateShort0(startPerfDate) + '<br>\n';
406 perf += DateShort1() + '<br>\n';
407 elps = d.getTime() - startPerfDate.getTime();
408 itvl = d.getTime() - prevPerfDate.getTime();
409 perf += 'Elapsed: ' + elps/1000 + ' s<br>\n';
410 perf += 'Skew: ' + (itvl-1000) + ' ms<br>\n';
411 prevPerfDate = d;
412
413 if( performance.memory != undefined ){
414 m0 = performance.memory;
415 mu0 = (m0.usedJSHeapSize / 1000000.0); //.toFixed(6);
416 perf += 'Memory: '+mu0+' MB<br>\n';
417 }
418 perf += '<br>\n';
419
420 //GshSidebar.innerHTML = perf;
421 GshPerfMon.innerHTML = perf;
422 //GshIndexer.innerHTML = 'Memory: '+mu0+' MB';
423 //console.log('-- PerfMon heap: '+mu0+'/'+m0.totalHeapSize+'/'+m0.jsHeapSizeLimit);
424 if( true ){
425 GshSidebar.style.zIndex = 1000;
426 GshIndexer.style.zIndex = 0;
427 GshPerfMon.style.zIndex = 1;
428 //GshSidebar.appendChild(GshPerfMon);
429 if( document.getElementById('primary') == null ){ // not in WordPress
430 // GshPerfMon.style.position = 'absolute';
431 }
432 GshPerfMon.style.display = 'block';
433 GshPerfMon.style.marginLeft = '4px';
434 //GshPerfMon.style.top = '45px';
435 GshPerfMon.style.position = 'relative';
436 //GshPerfMon.style.position = 'absolute';
437 //topy = GshTopbar.getBoundingClientRect().top;
438 //topy = parseInt(topy) + 40;
439 //GshPerfMon.style.top = topy + 'px';
440 GshPerfMon.style.left = '0px';
441
442 GshMain.style.top = -GshPerfMon.getBoundingClientRect().height;
443 }
444 }
445 function ResetPerfMon(){
446 GshPerfMon.removeAttribute('style');
447 GshSidebar.removeAttribute('style');
448 }
449
450 var iserno = 0;
451 var GeneratedId = 0;
452 function generateIndex(n1,e,chw,nch,ht){
453 // https://developer.mozilla.org/en-US/docs/Web/API/Element
454 c = '';
455 if( e.classList != null ){
456 c = e.classList.value;
457 }
458 //console.log('-- <'+e.nodeName+'> #' + e.id + ' .'+c+' '+e.attributes);
459 if( e.nodeName == '#text' ) return '';
460 if( e.nodeName == '#comment' ) return '';
461 if( e.nodeName == 'H2' || e.nodeName == 'H3' ){
462 id = e.innerHTML;
463 GeneratedId += 1;
464 eid = 'GeneratedId-'+GeneratedId;
465 e.id = eid;
466 }else
467 if( e.nodeName == 'SUMMARY' ){
468 id = e.innerHTML;
469 GeneratedId += 1;
470 eid = 'GeneratedId-'+GeneratedId;
471 e.id = eid;
472 }else
473 if( and(e.nodeName == 'DIV',c='xxxxxxxxxxxxxxxxentry-content') ){
474 console.log('-- DIV entry-content begin');
475 id = e.innerHTML;
476 GeneratedId += 1;
477 eid = 'GeneratedId-'+GeneratedId;
478 e.id = eid;
479 console.log('-- DIV entry-content end hash-child=',e.hasChildNodes());
480 }else
481 if( e.id == '' || e.id == 'undefined' ){
482 return '';
483 }else{
484 id = '#' + e.id;
485 e.id = e.id;
486 }
487 iserno += 1;
488 ht = '<'+'div id="GeneratedEref'+iserno+' class="IndexLine" href="'+eid+'"'>
489 + iserno+' '+n1+' '+e.nodeName+' ': ' + id;
490 if( e.id == '' || e.id == 'undefined' ){ return ht + '<'+'div>'; }
491 if( !e.hasChildNodes() ) return ht + '<'+'div>'; }
492 chw = e.childNodes;
493 nch = e.childNodes.length;
494 if( chw != null ){ nch = chw.length; }
495 ht += ('+nch+') + '<'+'div>';
496 for( let i = 0; i < chw.length; i++){
497 sec = ni+' '+i;
498 if( ni == '' ){ sec = i; }
499 ht += generateIndex(sec,chw[i],null,0);
500 }
501 return ht;
502 }
503 function onClickIndex(e){
504 tid = e.target.id;
505 tge = document.getElementById(tid);
506 eid = tge.getAttribute('href');
507 rx = tge.getBoundingClientRect().left.toFixed(0)
508 ry = tge.getBoundingClientRect().top.toFixed(0)
509 if( false ){
510 alert('index clicked mouse(x='+e.x+', y='+e.y+')'
511 + '\ntid='+tid + ' rx'+rx + ',ry'+ry
512 + '\neid=' + eid + '\n'
513 + '\nhtml='+ tge.outerHTML);
514 }
515 ee = document.getElementById(eid);
516 sx = 'NaN';
517 sy = ee.getBoundingClientRect().top;
518 console.log('sx'+sx+',sy'+sy);
519 ee.scrollIntoView()
520 window.scrollTo(sx,sy)
521 //window.scrollTo({left:'Non',top:sy,behavior:'smooth'});
522 }
523 function Indexer_afterLoaded(){
524 sideindex = document.getElementById('GshIndexer');
525 ht = '<'+'h3>G-Index<'+'h3>';
526 ht += generateIndex('',document.getElementById('gsh'),null,0,'');
527 if( (pri = document.getElementById('primary')) != null ){
528 ht += generateIndex('',pri,null,0,'');
529 }
530 ht += '<'+'br>';

```

```

531 ht += '<+br>';
532 ht += '<+br>';
533 ht += '<+br>';
534 sideindex.innerHTML = ht;
535 sideindex.addEventListener('click',onClickListener);
536
537 if( (pri = document.getElementById('primary')) != null ){
538 console.log('-- Seems in WordPress');
539 pri.style.zIndex = 2000;
540
541 GshSidebar.style.setProperty('position','relative','important');
542 GshSidebar.style.top = '-1400px';
543 //GshSidebar.style.setProperty('position','absolute','important');
544 //GshSidebar.style.top = '0px';
545
546 GshSidebar.style.setProperty('width','200px','important');
547 GshSidebar.style.setProperty('overflow','scroll','important');
548 GshSidebar.style.resize = 'both';
549
550 GshSidebar.style.left = '-100px';
551 GshIndexer.style.left = '100px';
552 GshIndexer.style.height = '1400px';
553 gsh.appendChild(GshSidebar); // change parent
554 }else{
555 console.log('-- Seems not in WordPress');
556 GshSidebar.style.setProperty('position','fixed','important');
557 }
558 }
559 //document.addEventListener('load',Indexer_afterLoaded);
560
561 DestroyIndexerBar = function(){
562 sideindex = document.getElementById('GshIndexer');
563 sideindex.innerHTML = "";
564 sideindex.style = "";
565 }
566 </script>
567
568 <!-- Indexer_WorkCodeSpan -->
569 <!-- Work -->
570
571
572
573
574 /*
575 <h2>GShell // a General purpose Shell built on the top of Golang</h2>
576 <p>
577 <note>
578 It is a shell for myself, by myself, of myself. --SatoxITS("-")
579 <a href="gsh-0.6.2.go.html">prev.</a>
580 </note>
581 </p>
582 <div id="GJFactory_x"></div>
583
584 <div>
585 <span id="GshMenu" class="GshMenu">
586 <span class="GshMenu" id="GshMenuEdit" onclick="html_edit();">Edit</span>
587 <span class="GshMenu" id="GshMenuSave" onclick="html_save();">Save</span>
588 <span class="GshMenu" id="GshMenuLoad" onclick="html_load();">Load</span>
589 <span class="GshMenu" id="GshMenuVers" onclick="html_ver0();">Vers</span>
590 <span id="gsh-win" onclick="win_jump('0.1');"></span>
591 <span class="GshMenu" id="gsh-menu-exit" onclick="html_close();"></span>
592 <span class="GshMenu" id="gsh-menu-fork" onclick="html_fork();">Fork</span>
593 <span class="GshMenu" id="GshMenuStop" onclick="html_stop(this,true);">Stop</span>
594 <span class="GshMenu" id="GshMenuFold" onclick="html_fold(this);">Unfold</span>
595 <span class="GshMenu" id="gsh-menu-cksum" onclick="html_digest();">Digest</span>
596 <span class="GshMenu" id="GshMenuSign" onclick="html_sign(this);" style="background-color:#f0f0f0;">Source</span>
597 </div>
598 </div>
599
600 /*
601 <details id="GshStatement" class="gsh-document"><summary>Statement</summary>
602 <h3>Fun to create a shell</h3>
603 <p>For a programmer, it must be far easy and fun to create his own simple shell
604 rightly fitting to his favor and necessities, than learning existing shells with
605 complex full features that he never use.
606 I, as one of programmers, am writing this tiny shell for my own real needs,
607 totally from scratch, with fun.
608 </p><p>
609 For a programmer, it is fun to learn new computer languages. For long years before
610 writing this software, I had been specialized to C and early HTML2 .
611 Now writing this software, I'm learning Go language, HTML5, JavaScript and CSS
612 on demand as a novice of these, with fun.
613 </p><p>
614 This single file "gsh.go", that is executable by Go, contains all of the code written
615 in Go. Also it can be displayed as "gsh.go.html" by browsers. It is a standalone
616 HTML file that works as the viewer of the code of itself, and as the "home page" of
617 this software.
618 </p><p>
619 Because this HTML file is a Go program, you may run it as a real shell program
620 on your computer.
621 But you must be aware that this program is written under situation like above.
622 Needless to say, there is no warranty for this program in any means.
623 </p>
624 <address>Aug 2020, SatoxITS (sato@its-more.jp)</address>
625 </details>
626
627 /*
628 <details id="GshFeatures" class="gsh-document"><summary>Features</summary><p>
629 </p>
630 <h3>Cross-browser communication</h3>
631 <p>
632 ... to be written ...
633 </p>
634 <h3>Vi compatible command line editor</h3>
635 <p>
636 The command line of GShell can be edited with commands compatible with
637 <a href="https://www.washington.edu/computing/unix/vi.html"><b>vi</b></a>.
638 As in vi, you can <b>h</b>-command mode/<b>o</b> by <b>ESC</b> key,
639 then move around in the history by <b>code>j k / ? n N</code>/<b>,
640 or within the current line by <b>code>l h f w b O $</code>/<b> or so.
641 </p>
642 </details>
643
644 /*
645 <details id="gsh-index">
646 <summary>Index</summary><div class="gsh-src">
647 Documents
648 <span class="gsh-link" onclick="jumpto_JavaScriptView();">Command summary</span>
649 Go lang part<span class="gsh-src" onclick="document.getElementById('gsh-gocode').open=true;">
650 Package structures
651 <a href="#import">import</a>
652 <a href="#struct">struct</a>
653 Main functions
654 <a href="#comexpansion">str-expansion</a> // macro processor
655 <a href="#finder">finder</a> // builtin find + du
656 <a href="#grep">grep</a> // builtin grep + wc + cksum + ...
657 <a href="#plugin">plugin</a> // plugin commands
658 <a href="#ex-commands">system</a> // external commands
659 <a href="#builtin">builtin</a> // builtin commands
660 <a href="#network">network</a> // socket handler
661 <a href="#remote-sh">remote-sh</a> // remote shell
662 <a href="#redirect">redirect</a> // StdIn/Out redirector
663 <a href="#history">history</a> // command history
664 <a href="#rusage">rusage</a> // resource usage
665 <a href="#encode">encode</a> // encode / decode
666 <a href="#lme">lme</a> // command line lme
667 <a href="#getline">getline</a> // line editor
668 <a href="#scanf">scanf</a> // string decomposer
669 <a href="#interpreter">interpreter</a> // command interpreter
670 <a href="#main">main</a>
671 </span>
672 JavaScript part
673 <a href="#script-src-view" class="gsh-link" onclick="jumpto_JavaScriptView();">Source</a>
674 <a href="#gsh-data-frame" class="gsh-link" onclick="jumpto_DataView();">Builtin data</a>
675 CSS part
676 <a href="#style-src-view" class="gsh-link" onclick="jumpto_StyleView();">Source</a>
677 References
678 <a href="#" class="gsh-link" onclick="jumpto_WholeView();">Internal</a>
679 <a href="#gsh-reference" class="gsh-link" onclick="jumpto_ReferenceView();">External</a>
680 Whole parts
681 <a href="#whole-src-view" class="gsh-link" onclick="jumpto_WholeView();">Source</a>
682 <a href="#whole-src-view" class="gsh-link" onclick="jumpto_WholeView();">Download</a>
683 <a href="#whole-src-view" class="gsh-link" onclick="jumpto_WholeView();">Dump</a>
684 </div>
685 </details>
686
687 <details id="gsh-gocode">
688 <summary>Go Source</summary><div class="gsh-src" onclick="document.getElementById('gsh-gocode').open=false;">
689 // gsh - Go lang based Shell
690 // (c) 2020 ITS more Co., Ltd.
691 // 2020-0807 created by SatoxITS (sato@its-more.jp)
692
693 package main // gsh main
694
695 // <a name="import">Imported packages</a> // <a href="https://golang.org/pkg/">Packages</a>
696 import (
697 	"fmt" // <a href="https://golang.org/pkg/fmt/">fmt</a>
698 	"errors"
699 	"strings" // <a href="https://golang.org/pkg/strings/">strings</a>
700 	"strconv" // <a href="https://golang.org/pkg/strconv/">strconv</a>
701 	"sort" // <a href="https://golang.org/pkg/sort/">sort</a>
702 	"time" // <a href="https://golang.org/pkg/time/">time</a>
703 	"bufio" // <a href="https://golang.org/pkg/bufio/">bufio</a>
704 	"io/ioutil" // <a href="https://golang.org/pkg/io/ioutil/">ioutil</a>

```

```

708 "os" // <a href="https://golang.org/pkg/os/">os</a>
709 "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
710 "plugin" // <a href="https://golang.org/pkg/plugin/">plugin</a>
711 "net" // <a href="https://golang.org/pkg/net/">net</a>
712 "net/http" // <a href="https://golang.org/pkg/net/http/">http</a>
713 // "html" // <a href="https://golang.org/pkg/html/">html</a>
714 "path/filepath" // <a href="https://golang.org/pkg/path/filepath/">filepath</a>
715 "go/types" // <a href="https://golang.org/pkg/go/types/">types</a>
716 "go/token" // <a href="https://golang.org/pkg/go/token/">token</a>
717 "encoding/base64" // <a href="https://golang.org/pkg/encoding/base64/">base64</a>
718 "unicode/utf8" // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
719 // "gshdata" // gshell's logo and source code
720 "hash/crc32" // <a href="https://golang.org/pkg/unicode/hash/crc32/">crc32</a>
721 "golang.org/x/net/websocket"
722 runtime
723 )
724
725 /*
726 #include <stdio.h> // </stdio.h> to be closed as HTML tag :-p
727 #ifdef WIN32
728 #include <windows.h> // </windows.h>
729 // 2020-1022 added -- terminal mode on Windows
730 // https://docs.microsoft.com/en-us/windows/console/setconsolemode
731 // https://docs.microsoft.com/en-us/windows/win32/inputdev/using-keyboard-input
732 int setTermRaw(){
733     HANDLE hStdin = GetStdHandle(STD_INPUT_HANDLE);
734     DWORD tmode = 0;
735     if( GetConsoleMode(hStdin,&tmode) ){
736         DWORD xmode = tmode;
737         xmode &= ~ENABLE_ECHO_INPUT;
738         xmode &= ~ENABLE_LINE_INPUT;
739         xmode |= ENABLE_PROCESSED_INPUT; // Control+C for SIGINT
740         if( SetConsoleMode(hStdin,xmode) ){
741             return tmode;
742         }
743     }
744     return 0;
745 }
746 int setTermMode(int tmode){
747     HANDLE hStdin = GetStdHandle(STD_INPUT_HANDLE);
748     SetConsoleMode(hStdin,tmode);
749     return 0;
750 }
751 #else
752 int setTermRaw(){
753     return -1;
754 }
755 int setTermMode(int tmode){
756     return 0;
757 }
758 #endif
759 #import "c"
760
761 // 2020-0906 added,
762 // <a href="https://golang.org/cmd/cgo/">CGO</a>
763 #include "poll.h" // <poll.h> // </poll.h> to be closed as HTML tag :-p
764 #typedef struct { struct pollfd fd[8]; } pollFdv;
765 // int pollx(pollFdv *fdv, int nfds, int timeout){
766 //     return poll(fdv->fdv,nfds,timeout);
767 // }
768 #import "c"
769
770 // 2020-1021 replaced poll() with channel/select
771 // 2020-0906 added,
772 func CFPollInl(fp*os.File, timeoutUs int)(ready uintptr){
773     var fdv = C.pollFdv{}
774     var nfds = 1
775     var timeout = timeoutUs/1000
776
777     fdv.fdv[0].fd = C.int(fp.Fd())
778     fdv.fdv[0].events = C.POLLIN
779     if( 0 < EventRecvFd {
780         fdv.fdv[1].fd = C.int(EventRecvFd)
781         fdv.fdv[1].events = C.POLLIN
782         nfds += 1
783     }
784     r := C.pollx(&fdv,C.int(nfds),C.int(timeout))
785     if( r <= 0 ){
786         return 0
787     }
788     if( (int(fdv.fdv[1].revents) & int(C.POLLIN)) != 0 {
789         //printf(stderr,"-De- got Event\n");
790         return uintptr(EventPdOffset + fdv.fdv[1].fd)
791     }
792     if( (int(fdv.fdv[0].revents) & int(C.POLLIN)) != 0 {
793         return uintptr(NormalPdOffset + fdv.fdv[0].fd)
794     }
795     return 0
796 }
797
798 const {
799     NAME = "gsh"
800     VERSION = "0.8.4"
801     DATE = "2020-11-17"
802     AUTHOR = "SatoxITS(-)"//
803 }
804 var {
805     GSH_HOME = ".gsh" // under home directory
806     GSH_PORT = 9999
807     MaxStreamsSize = int64(128*1024*1024*1024) // 128GiB is too large?
808     PROMPT = ">"
809     LINESIZE = (8*1024)
810     PATHSEP = ":" // should be ";" in Windows
811     DIRSEP = "/" // canbe \ in Windows
812     OnWindows = false;
813 }
814 func initGshEnv(){
815     if( runtime.GOOS == "windows" ){
816         PATHSEP = ";";
817         DIRSEP = "\\";
818         OnWindows = true;
819     }else{
820     }
821 }
822
823 // -x logging control
824 // -h all
825 // -i info.
826 // -d debug
827 // -t time and resource usage
828 // -w warning
829 // -e error
830 // -f fatal error
831 // -n network
832 // <a name="struct">Structures</a>
833
834 // 2020-1022 Unix/Windows
835 // -----
836 //type aStat_t syscall.Stat_t;
837 //type aStat_t struct { syscall.Stat_t }
838 type aStat_t struct {
839     Size int64
840     Mode os.FileMode
841     Rdev int64
842     Blocks int64
843     Nlink int64
844 }
845 func aLStat(path string, astat *aStat_t)(error){
846     /*
847     sstat := syscall.Stat_t{};
848     err := syscall.LStat(path,&sstat);
849     *astat = aStat_t(sstat);
850     */
851     fi,err := os.Stat(path);
852     if( err == nil ){
853         astat.Mode = fi.Mode();
854         astat.Size = fi.Size();
855     }
856     return err;
857 }
858 func aFStat(fd int, astat *aStat_t)(error){
859     /*
860     sstat := syscall.Stat_t{};
861     err := syscall.FStat(fd,&sstat);
862     *astat = aStat_t(sstat);
863     */
864     err := errors.New("NotImplemented-Fstat");
865     //fmt.Printf("---E- fstat(%v)\n",fd,err);
866     return err;
867 }
868 func aAccess(path string, mode uint32)(error){
869     //err := syscall.Access(path,mode);
870     //err := errors.New("NotImplemented-Access");
871     fi,err := os.Stat(path)
872     //fmt.Printf("--- Access(%v,%v)\n(%v)\n",path,mode,err);
873     if( err == nil ){
874         fmode := fi.Mode();
875         if( fmode.IsRegular() ){
876             perm := fmode.Perm();
877             if( (uint32(perm) & mode) != 0 ){
878                 return nil;
879             }
880         }
881     }
882 }

```

```

885     }
886     return errors.New("NotAccessible");
887 }
888     return errors.New("NotRegularFile");
889 }
890 return err;
891 }
892
893 // 2020-1022 Unix/Windows
894 // -----
895 type aRusage struct {
896     syscall.Rusage
897     Utime      time.Duration
898     Stime      time.Duration
899     //Sys      interface{}
900 }
901
902 /*
903 const aRUSAGE_SELF = syscall.RUSAGE_SELF
904 const aRUSAGE_CHILDREN = syscall.RUSAGE_CHILDREN
905 */
906 const aRUSAGE_SELF = 0
907 const aRUSAGE_CHILDREN = 1
908 func aGetRusage(sel int, ru *aRusage){
909     /*
910     sysru := syscall.Rusage{};
911     syscall.GetRusage(sel, &sysru);
912     ru.Utime = time.Duration(int64(sysru.Utime.Sec)*1000000000+int64(sysru.Utime.Usec)*1000);
913     ru.Stime = time.Duration(int64(sysru.Stime.Sec)*1000000000+int64(sysru.Stime.Usec)*1000);
914     */
915 }
916
917 func aSetRusage(ru *aRusage, ps *os.ProcessState){
918     ru.Utime = ps.UserTime();
919     ru.Stime = ps.SystemTime();
920 }
921
922 func showRusage(what string, argv []string, ru *aRusage){
923     fmt.Printf("%s: ", what);
924     //fmt.Printf("User=%d.%06ds", ru.Utime.Sec, ru.Utime.Usec)
925     //fmt.Printf("Sys=%d.%06ds", ru.Stime.Sec, ru.Stime.Usec)
926     fmt.Printf("User=%d.%06ds", ru.Utime/1000000000, (ru.Utime/1000)%1000000);
927     fmt.Printf(" Sys=%d.%06ds ", ru.Stime/1000000000, (ru.Stime/1000)%1000000);
928
929     /*
930     fmt.Printf(" Rsa=%vB", ru.Maxrss)
931     if isin("-l", argv) {
932         fmt.Printf(" MinFlt=%v", ru.Minflt)
933         fmt.Printf(" MajFlt=%v", ru.Majflt)
934         fmt.Printf(" IxRSS=%vB", ru.Ixrss)
935         fmt.Printf(" IdRSS=%vB", ru.Idrss)
936         fmt.Printf(" Nswap=%v", ru.Nswap)
937         fmt.Printf(" Read=%v", ru.Inblock)
938         fmt.Printf(" Write=%v", ru.Oblock)
939     }
940     fmt.Printf(" Snd=%v", ru.Msgsnd)
941     fmt.Printf(" Rcv=%v", ru.Msgrcv)
942     //if isin("-l", argv) {
943         //fmt.Printf(" Sig=%v", ru.Nsignals)
944     //}
945     */
946     fmt.Printf("\n");
947 }
948
949 type GCommandHistory struct {
950     StartAt      time.Time // command line execution started at
951     EndAt        time.Time // command line execution ended at
952     ResCode      int // exit code of (external) command
953     CmdError     error // error string
954     OutData      *os.File // output of the command
955     FoundFile    []string // output - result of ufind
956     Rusagev      [2]aRusage // Resource consumption, CPU time or so
957     Cmdid       int // maybe with identified with arguments or impact
958     WorkDir      string // redirection commands should not be the Cmdid
959     WorkDirX     int // working directory at start
960     WorkDirX     int // index in ChdirHistory
961     CmdLine      string // command line
962 }
963
964 type GChdirHistory struct {
965     Dlr          string
966     MovedAt    time.Time
967     CmdIndex    int
968 }
969
970 type CmdMode struct {
971     BackGround   bool
972 }
973
974 type Event struct {
975     when         time.Time
976     event        int
977     evarg        int64
978     CmdIndex    int
979 }
980
981 var CmdIndex int
982 var Events []Event
983
984 type PluginInfo struct {
985     Spec         *plugin.Plugin
986     Addr         plugin.Symbol
987     Name         string // maybe relative
988     Path         string // this is in Plugin but hidden
989 }
990
991 type GServer struct {
992     host         string
993     port         string
994 }
995
996 // <a href="https://tools.ietf.org/html/rfc3230">Digest</a>
997 const ( // SumType
998     SUM_ITEMS      = 0x000001 // items count
999     SUM_SIZE       = 0x000002 // data length (simply added)
1000    SUM_SYMDHASH   = 0x000004 // data length (hashed sequence)
1001    SUM_DATEHASH   = 0x000008 // date of data (hashed sequence)
1002    // also envelope attributes like time stamp can be a part of digest
1003    // hashed value of sizes or mod-date of files will be useful to detect changes
1004
1005    SUM_WORDS      = 0x000010 // word count is a kind of digest
1006    SUM_LINES      = 0x000020 // line count is a kind of digest
1007    SUM_SUM4       = 0x000040 // simple add of bytes, useful for human too
1008
1009    SUM_SUM32_BITS = 0x000100 // the number of true bits
1010    SUM_SUM32_2BYTE = 0x000200 // 16bits words
1011    SUM_SUM32_4BYTE = 0x000400 // 32bits words
1012    SUM_SUM32_8BYTE = 0x000800 // 64bits words
1013
1014    SUM_SUM16_BSD  = 0x001000 // UNIXsum -sum -bsd
1015    SUM_SUM16_SYSV = 0x002000 // UNIXsum -sum -sysv
1016    SUM_UNIXFILE   = 0x004000
1017    SUM_CRCIEEE   = 0x008000
1018 )
1019
1020 type CheckSum struct {
1021     Files      int64 // the number of files (or data)
1022     Size       int64 // content size
1023     Words      int64 // word count
1024     Lines      int64 // line count
1025     SumType    int
1026     Sum4       uint64
1027     Crc32Table crc32.Table
1028     Crc32Val   uint32
1029     Sum16      int
1030     CTime      time.Time
1031     ATime      time.Time
1032     MTime      time.Time
1033     Start      time.Time
1034     Done       time.Time
1035     RusgAtStart [2]aRusage
1036     RusgAtEnd   [2]aRusage
1037 }
1038
1039 type ValueStack [][]string
1040
1041 type GshContext struct {
1042     StartDir      string // the current directory at the start
1043     GetLine       string // gsh-getline command as a input line editor
1044     ChdirHistory  []GChdirHistory // the 1st entry is wd at the start
1045     //gshPA        syscall.ProcAttr
1046     gshPA         os.ProcAttr
1047     CommandHistory []GCommandHistory
1048     CmdCurrent    GCommandHistory
1049     BackGround    bool
1050     BackgroundJobs []os.ProcessState; [][]int
1051     LastRusage    aRusage
1052     GshHomeDir    string
1053     TerminalId    int
1054     CmdTrace      bool // should be [map]
1055     CmdTime       bool // should be [map]
1056     PluginFuncs   []PluginInfo
1057     iValues        []string
1058     iDelimiter    string // field separator of print out
1059     iFormat       string // default print format (of integer)
1060     iValStack     ValueStack
1061     LastServer    GServer
1062     RSRV          string // [gsh://host[:port]
1063     RWD           string // remote (target, there) working directory
1064     lastCheckSum CheckSum
1065 }
1066
1067 func nsleep(ns time.Duration){
1068     time.Sleep(ns)
1069 }
1070
1071 func usleep(ns time.Duration){
1072     nsleep(ns*1000)
1073 }
1074
1075 func msleep(ns time.Duration){

```

```

1062     nsleep(ns*1000000)
1063 }
1064 func sleep(ns time.Duration){
1065     nsleep(ns*1000000000)
1066 }
1067
1068 func strBegins(str, pat string)(bool){
1069     if len(pat) <= len(str){
1070         yes := str[0:len(pat)] == pat
1071         //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat,yes)
1072         return yes
1073     }
1074     //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat,false)
1075     return false
1076 }
1077 func isin(what string, list []string) bool {
1078     for v := range list {
1079         if v == what {
1080             return true
1081         }
1082     }
1083     return false
1084 }
1085 func isinX(what string,list[]string)(int){
1086     for i,v := range list {
1087         if v == what {
1088             return i
1089         }
1090     }
1091     return -1
1092 }
1093
1094 func env(opts []string) {
1095     env := os.Environ()
1096     if isin("-s", opts){
1097         sort.Slice(env, func(i,j int) bool {
1098             return env[i] < env[j]
1099         })
1100     }
1101     for _, v := range env {
1102         fmt.Printf("%v\n",v)
1103     }
1104 }
1105
1106 // - rewriting should be context dependent
1107 // - should postpone until the real point of evaluation
1108 // - should rewrite only known notation of symbol
1109 func scanInt(str string)(val int,leng int){
1110     leng = -1
1111     for i,ch := range str {
1112         if '0' <= ch && ch <= '9' {
1113             leng = i+1
1114         }else{
1115             break
1116         }
1117     }
1118     if 0 < leng {
1119         ival,_ := strconv.Atoi(str[0:leng])
1120         return ival,leng
1121     }else{
1122         return 0,0
1123     }
1124 }
1125 func substHistory(gshCtx *GshContext,str string,i int,rstr string)(leng int,rst string){
1126     if len(str[i+1:]) == 0 {
1127         return 0,rstr
1128     }
1129     hi := 0
1130     histlen := len(gshCtx.CommandHistory)
1131     if str[i+1] == '!' {
1132         hi = histlen - 1
1133         leng = 1
1134     }else{
1135         hi,leng = scanInt(str[i+1:])
1136         if leng == 0 {
1137             return 0,rstr
1138         }
1139         if hi < 0 {
1140             hi = histlen + hi
1141         }
1142     }
1143     if 0 <= hi && hi < histlen {
1144         var ext byte
1145         if 1 < len(str[i+leng:]){
1146             ext = str[i+leng:][1]
1147         }
1148         //fmt.Printf("--D-- %v(%c)\n",str[i+leng:],str[i+leng])
1149         if ext == 'f' {
1150             leng += 1
1151             xlist := []string{}
1152             list := gshCtx.CommandHistory[hi].FoundFile
1153             for v := range list {
1154                 //list[i] = escapeWhiteSP(v)
1155                 xlist = append(xlist,escapeWhiteSP(v))
1156             }
1157             //rstr += strings.Join(list, " ")
1158             rstr += strings.Join(xlist, " ")
1159         }else{
1160             if ext == 'q' || ext == 'd' {
1161                 // IN8 .. workdir at the start of the command
1162                 leng += 1
1163                 rstr += gshCtx.CommandHistory[hi].WorkDir
1164             }else{
1165                 rstr += gshCtx.CommandHistory[hi].CmdLine
1166             }
1167         }else{
1168             leng = 0
1169         }
1170     }
1171     return leng,rstr
1172 }
1173 func escapeWhiteSP(str string)(string){
1174     if len(str) == 0 {
1175         return "" // empty, to be ignored
1176     }
1177     rstr := ""
1178     for _,ch := range str {
1179         switch ch {
1180             case '\\': rstr += "\\\\"
1181             case ' ': rstr += "\\s"
1182             case '\t': rstr += "\\t"
1183             case '\r': rstr += "\\r"
1184             case '\n': rstr += "\\n"
1185             default: rstr += string(ch)
1186         }
1187     }
1188     return rstr
1189 }
1190 func unescapeWhiteSP(str string){ // strip original escapes
1191     rstr := ""
1192     for i := 0; i < len(str); i++ {
1193         ch := str[i]
1194         if ch == '\\' {
1195             if i+1 < len(str) {
1196                 switch str[i+1] {
1197                     case '2':
1198                         continue;
1199                 }
1200             }
1201             rstr += string(ch)
1202         }
1203     }
1204     return rstr
1205 }
1206 func unescapeWhiteSPV(strv []string)([]string){ // strip original escapes
1207     ustrv := []string{}
1208     for _,v := range strv {
1209         ustrv = append(ustrv,unescapeWhiteSP(v))
1210     }
1211     return ustrv
1212 }
1213 // <a name="comexpansion">str-expansion</a>
1214 // - this should be a macro processor
1215 func strsubst(gshCtx *GshContext,str string,histonly bool) string {
1216     rbuff := []byte{}
1217     if false { //@@@ Unicode should be cared as a character
1218         return str
1219     }
1220     //rstr := ""
1221     inEsc := 0 // escape characer mode
1222     for i := 0; i < len(str); i++ {
1223         //fmt.Printf("--D--Subst %v:%v\n",i,str[i:])
1224         ch := str[i]
1225         if inEsc == 0 {
1226             if ch == '!'' {
1227                 //leng,xrstr := substHistory(gshCtx,str,i,rstr)
1228                 //leng,rs := substHistory(gshCtx,str,i,"")
1229                 if 0 < leng {
1230                     //_,rs := substHistory(gshCtx,str,i,"")
1231                     rbuff = append(rbuff,[]byte(rs)...)
1232                     i += leng
1233                     //rstr += xrstr
1234                     continue
1235                 }
1236             }
1237         }
1238         switch ch {

```

```

1239         case '\\': inEsc = '\\'; continue
1240         //case '$': inEsc = '$'; continue
1241         case '$':
1242     }
1243 }
1244 switch inEsc {
1245 case '\\':
1246     switch ch {
1247     case '\\': ch = '\\'
1248     case '$': ch = '$'
1249     case 't': ch = '\t'
1250     case 'r': ch = '\r'
1251     case 'n': ch = '\n'
1252     case 'z': inEsc = 0; continue // empty, to be ignored
1253     }
1254     inEsc = 0
1255 case '$':
1256     switch {
1257     case ch == '$': ch = '$'
1258     case ch == 'T':
1259         //rstr = rstr + time.Now().Format(time.Stamp)
1260         rs := time.Now().Format(time.Stamp)
1261         rbuff = append(rbuff, []byte(rs)...)
1262         inEsc = 0
1263         continue;
1264     default:
1265         // postpone the interpretation
1266         //rstr = rstr + '$' + string(ch)
1267         rbuff = append(rbuff, ch)
1268         inEsc = 0
1269         continue;
1270     }
1271     inEsc = 0
1272 }
1273 //rstr = rstr + string(ch)
1274 rbuff = append(rbuff, ch)
1275 }
1276 //fmt.Printf("--D--subst(%s)(%s)\n",str,string(rbuff))
1277 return string(rbuff)
1278 //return rstr
1279 }
1280 func showFileInfo(path string, opts []string) {
1281     if !isIn("-l",opts) || !isIn("-ls",opts) {
1282         fi, err := os.Stat(path)
1283         if err != nil {
1284             fmt.Printf("----- ((%v)),err)
1285         }else{
1286             mod := fi.ModTime()
1287             date := mod.Format(time.Stamp)
1288             fmt.Printf("%v %v %s ",fi.Mode(),fi.Size(),date)
1289         }
1290     }
1291     fmt.Printf("%s",path)
1292     if !isIn("-sp",opts) {
1293         fmt.Printf(" ")
1294     }else
1295     if !isIn("-n",opts) {
1296         fmt.Printf("\n")
1297     }
1298 }
1299 func userHomeDir()(string,bool){
1300     /*
1301     homedir, _ = os.UserHomeDir() // not implemented in older Golang
1302     */
1303     homedir,found := os.LookupEnv("HOME")
1304     //fmt.Printf("--I-- HOME=%v(%v)\n",homedir,found)
1305     if !found {
1306         return "/tmp",found
1307     }
1308     return homedir,found
1309 }
1310 }
1311 func toFullPath(path string) (fullpath string) {
1312     if path[0] == '/' {
1313         return path
1314     }
1315     pathv := strings.Split(path,DIRSEP)
1316     switch {
1317     case pathv[0] == ".":
1318         pathv[0], _ = os.Getwd()
1319     case pathv[0] == "..": // all ones should be interpreted
1320         cwd, _ = os.Getwd()
1321         ppathv := strings.Split(cwd,DIRSEP)
1322         pathv[0] = strings.Join(ppathv,DIRSEP)
1323     case pathv[0] == "~":
1324         pathv[0], _ = userHomeDir()
1325     default:
1326         cwd, _ = os.Getwd()
1327         pathv[0] = cwd + DIRSEP + pathv[0]
1328     }
1329     return strings.Join(pathv,DIRSEP)
1330 }
1331 }
1332 func isRegFile(path string)(bool){
1333     fi, err := os.Stat(path)
1334     if err == nil {
1335         fm := fi.Mode()
1336         return fm.IsRegular();
1337     }
1338     return false
1339 }
1340 }
1341 // <a name="encode">Encode / Decode</a>
1342 // <a href="https://golang.org/pkg/encoding/base64/#example_NewEncoder">Encoder</a>
1343 func (gshCtx *GshContext)Enc(argv[]string){
1344     file := os.Stdin
1345     buff := make([]byte,LINESIZE)
1346     li := 0
1347     encoder := base64.NewEncoder(base64.StdEncoding,os.Stdout)
1348     for li = 0; li++ {
1349         count, err := file.Read(buff)
1350         if count <= 0 {
1351             break
1352         }
1353         if err != nil {
1354             break
1355         }
1356         encoder.Write(buff[0:count])
1357     }
1358     encoder.Close()
1359 }
1360 func (gshCtx *GshContext)Dec(argv[]string){
1361     decoder := base64.NewDecoder(base64.StdEncoding,os.Stdin)
1362     li := 0
1363     buff := make([]byte,LINESIZE)
1364     for li = 0; li++ {
1365         count, err := decoder.Read(buff)
1366         if count <= 0 {
1367             break
1368         }
1369         if err != nil {
1370             break
1371         }
1372         os.Stdout.Write(buff[0:count])
1373     }
1374 }
1375 // lnsfp [N] [-crlf][-C \\\
1376 func (gshCtx *GshContext)SplitLine(argv[]string){
1377     strRep := isin("-str",argv) // "...+"
1378     reader := bufio.NewReaderSize(os.Stdin,64*1024)
1379     ni := 0
1380     toi := 0
1381     for ni = 0; ni++ {
1382         line, err := reader.ReadString('\n')
1383         if len(line) <= 0 {
1384             if err != nil {
1385                 fmt.Fprintf(os.Stderr,"--I-- lnsfp %d to %d (%v)\n",ni,toi,err)
1386                 break
1387             }
1388         }
1389         off := 0
1390         ilen := len(line)
1391         remlen := len(line)
1392         if strRep { os.Stdout.Write([]byte("\n")) }
1393         for oi := 0; 0 < remlen; oi++ {
1394             olen := remlen
1395             addnl := false
1396             if 72 < olen {
1397                 olen = 72
1398                 addnl = true
1399             }
1400             fmt.Fprintf(os.Stderr,"--D-- write %d [%d.%d] %d %d/%d/%d\n",
1401                 toi,ni,oi,off,olen,remlen,ilen)
1402             toi += 1
1403             os.Stdout.Write([]byte(line[0:olen]))
1404             if addnl {
1405                 if strRep {
1406                     os.Stdout.Write([]byte("\n\n"))
1407                 }else{
1408                     //os.Stdout.Write([]byte("\r\n"))
1409                     os.Stdout.Write([]byte("\n"))
1410                     os.Stdout.Write([]byte("\n"))
1411                 }
1412             }
1413             line = line[olen:]
1414             off += olen
1415             remlen -= olen

```



```

1416     }
1417     if strRep { os.Stdout.Write([]byte("\n")) }
1418 }
1419 fmt.Fprintf(os.Stderr, "--I-- lnp %d to %d\n", ni, toi)
1420 }
1421 }
1422 // CRC32 <a href="http://golang.jp/pkg/hash-crc32">crc32</a>
1423 // 1 0000 0100 1100 0001 0001 1101 1011 0111
1424 var CRC32UNIX uint32 = uint32(0x04c11db7) // Unix cksum
1425 var CRC32IEEE uint32 = uint32(0x8b883200)
1426 func byteCRC32add(crc uint32, str []byte, len uint64)(uint32){
1427     var oi uint64
1428     for oi < len; oi++ {
1429         var oct = str[oi]
1430         for bi := 0; bi < 8; bi++ {
1431             //fprintf(stderr, "--CRC32 %d %X (%d.%d)\n", crc, oct, oi, bi)
1432             ovf1 := (crc & 0x00000000) != 0
1433             ovf2 := (oct & 0x80) != 0
1434             ovf := (ovf1 && !ovf2) || (!ovf1 && ovf2)
1435             oct <<= 1
1436             crc <<= 1
1437             if ovf { crc ^= CRC32UNIX }
1438         }
1439     }
1440     //fprintf(stderr, "--CRC32 return %d %d\n", crc, len)
1441     return crc;
1442 }
1443 func byteCRC32end(crc uint32, len uint64)(uint32){
1444     var slen = make([]byte, 4)
1445     var li = 0
1446     for li = 0; li < 4; {
1447         slen[li] = byte(len)
1448         li += 1
1449         len >>= 8
1450         if (len == 0) {
1451             break
1452         }
1453     }
1454     crc = byteCRC32add(crc, slen, uint64(li))
1455     crc ^= 0xFFFFFFFF
1456     return crc
1457 }
1458 func strCRC32(str string, len uint64)(crc uint32){
1459     crc = byteCRC32add(0, []byte(str), len)
1460     crc = byteCRC32end(crc, len)
1461     //fprintf(stderr, "--CRC32 %d %d\n", crc, len)
1462     return crc
1463 }
1464 func CRC32Finish(crc uint32, table *crc32.Table, len uint64)(uint32){
1465     var slen = make([]byte, 4)
1466     var li = 0
1467     for li = 0; li < 4; {
1468         slen[li] = byte(len & 0xFF)
1469         li += 1
1470         len >>= 8
1471         if (len == 0) {
1472             break
1473         }
1474     }
1475     crc = crc32.Update(crc, table, slen)
1476     crc ^= 0xFFFFFFFF
1477     return crc
1478 }
1479 }
1480 func (gsh*GshContext)xCksum(path string, argv []string, sum*Checksum)(int64){
1481     if isin("-type/f", argv) && !IsRegFile(path){
1482         return 0
1483     }
1484     if isin("-type/d", argv) && IsRegFile(path){
1485         return 0
1486     }
1487     file, err := os.OpenFile(path, os.O_RDONLY, 0)
1488     if err != nil {
1489         fmt.Fprintf("--E-- cksum %v (%v)\n", path, err)
1490         return -1
1491     }
1492     defer file.Close()
1493     if gsh.CmdTrace { fmt.Printf("--I-- cksum %v %v\n", path, argv) }
1494 }
1495 bi := 0
1496 var buff = make([]byte, 32*1024)
1497 var total int64 = 0
1498 var initTime = time.Time{}
1499 if sum.Start == initTime {
1500     sum.Start = time.Now()
1501 }
1502 for bi = 0; ; bi++ {
1503     count, err := file.Read(buff)
1504     if count <= 0 || err != nil {
1505         break
1506     }
1507     if (sum.SumType & SUM_SUM64) != 0 {
1508         s := sum.Sum64
1509         for _, c := range buff[0:count] {
1510             s += uint64(c)
1511         }
1512         sum.Sum64 = s
1513     }
1514     if (sum.SumType & SUM_UNIFILE) != 0 {
1515         sum.Crc32Val = byteCRC32add(sum.Crc32Val, buff, uint64(count))
1516     }
1517     if (sum.SumType & SUM_CRCIEEE) != 0 {
1518         sum.Crc32Val = crc32.Update(sum.Crc32Val, &sum.Crc32Table, buff[0:count])
1519     }
1520     // <a href="https://en.wikipedia.org/wiki/BSD_checksum">BSD checksum</a>
1521     if (sum.SumType & SUM_SUM16_BSD) != 0 {
1522         s := sum.Sum16
1523         for _, c := range buff[0:count] {
1524             s = (s >> 1) + ((s & 1) << 15)
1525             s += int(c)
1526             s &= 0xFFFF
1527             //fmt.Printf("BSDsum: %d%d %d\n", sum.Size+int64(i), i, s)
1528         }
1529         sum.Sum16 = s
1530     }
1531     if (sum.SumType & SUM_SUM16_SYSV) != 0 {
1532         for bj := 0; bj < count; bj++ {
1533             sum.Sum16 += int(buff[bj])
1534         }
1535     }
1536     total += int64(count)
1537 }
1538 sum.Done = time.Now()
1539 sum.Files += 1
1540 sum.Size += total
1541 if !isin("-s", argv) {
1542     fmt.Printf("%v ", total)
1543 }
1544 return 0
1545 }
1546 }
1547 // <a name="grep">grep</a>
1548 // "lines", "lin" or "lnp" for "(text) line processor" or "scanner"
1549 // a*, lab, c, ... sequential combination of patterns
1550 // what "LINE" is should be definable
1551 // generic line-by-line processing
1552 // grep [-v]
1553 // cat -n -v
1554 // uniq [-c]
1555 // tail -f
1556 // sed s/x/y/ or awk
1557 // grep with line count like wc
1558 // rewrite contents if specified
1559 func (gsh*GshContext)xGrep(path string, rexp []string)(int){
1560     file, err := os.OpenFile(path, os.O_RDONLY, 0)
1561     if err != nil {
1562         fmt.Printf("--E-- grep %v (%v)\n", path, err)
1563         return -1
1564     }
1565     defer file.Close()
1566     if gsh.CmdTrace { fmt.Printf("--I-- grep %v %v\n", path, rexp) }
1567     //reader := bufio.NewReaderSize(file, LINESIZE)
1568     reader := bufio.NewReaderSize(file, 80)
1569     li := 0
1570     found := 0
1571     for li = 0; ; li++ {
1572         line, err := reader.ReadString('\n')
1573         if len(line) <= 0 {
1574             break
1575         }
1576         if 150 < len(line) {
1577             // maybe binary
1578             break;
1579         }
1580         if err != nil {
1581             break
1582         }
1583         if 0 <= strings.Index(string(line), rexp[0]) {
1584             found += 1
1585             fmt.Printf("%s:%d: %s", path, li, line)
1586         }
1587     }
1588     //fmt.Printf("total %d lines %s\n", li, path)
1589     //if (0 < found) { fmt.Printf("(found %d lines %s)\n", found, path); }
1590     return found
1591 }
1592 }

```

```

1593 // <a name="finder">Finder</a>
1594 // finding files with it name and contents
1595 // file names are Osd
1596 // show the content with %x fmt list
1597 // ls -R
1598 // tar command by adding output
1599 type fileSum struct {
1600     Err int64 // access error or so
1601     Size int64 // content size
1602     DupSize int64 // content size from hard links
1603     Blocks int64 // number of blocks (of 512 bytes)
1604     DupBlocks int64 // Blocks pointed from hard links
1605     HLinks int64 // hard links
1606     Words int64
1607     Lines int64
1608     Files int64
1609     Dirs int64 // the num. of directories
1610     SymLink int64
1611     Flats int64 // the num. of flat files
1612     MaxDepth int64
1613     MaxNameLen int64 // max. name length
1614     nextRepo time.Time
1615 }
1616 func showFusage(dir string, fusage *fileSum) {
1617     bsume := float64(((fusage.Blocks-fusage.DupBlocks)/2)*1024)/1000000.0
1618     //bsumdup := float64((fusage.Blocks/2)*1024)/1000000.0
1619     fmt.Printf("%v: %v files (%vd %vs %vh) %3.6f MB (%.2f MBK)\n",
1620         dir,
1621         fusage.Files,
1622         fusage.Dirs,
1623         fusage.SymLink,
1624         fusage.HLinks,
1625         float64(fusage.Size)/1000000.0, bsume);
1626 }
1627
1628 const (
1629     S_IFMT = 0170000
1630     S_IFCHR = 0020000
1631     S_IFDIR = 0040000
1632     S_IFREG = 0100000
1633     S_IFLNK = 0120000
1634     S_IFSOCK = 0140000
1635 )
1636 func cumFinfo(fsum *fileSum, path string, stater error, fstat aStat_t, argv []string, verb bool)(*fileSum){
1637     now := time.Now()
1638     if time.Second <= now.Sub(fsum.nextRepo) {
1639         if fsum.nextRepo.IsZero(){
1640             tstamp := now.Format(time.Stamp)
1641             showFusage(tstamp, fsum)
1642         }
1643         fsum.nextRepo = now.Add(time.Second)
1644     }
1645     if stater != nil {
1646         fsum.Err += 1
1647         return fsum
1648     }
1649     fsum.Files += 1
1650     if l < fstat.Nlink {
1651         // must count only once...
1652         // at least ignore ones in the same directory
1653         //if finfo.Mode().IsRegular() {
1654             if (fstat.Mode & S_IFMT) == S_IFREG {
1655                 fsum.HLinks += 1
1656                 fsum.DupBlocks += int64(fstat.Blocks)
1657                 //fmt.Printf("---Dup HardLink %v %s\n", fstat.Nlink, path)
1658             }
1659         }
1660         //fsum.Size += finfo.Size()
1661         fsum.Size += fstat.Size
1662         fsum.Blocks += int64(fstat.Blocks)
1663         //if verb { fmt.Printf("%8dBk %s", fstat.Blocks/2, path) }
1664         if isin("-ls", argv){
1665             //if verb { fmt.Printf("%4d %8d ", fstat.Blksize, fstat.Blocks) }
1666             // fmt.Printf("%d\t", fstat.Blocks/2)
1667         }
1668         //if finfo.IsDir()
1669         if (fstat.Mode & S_IFMT) == S_IFDIR {
1670             fsum.Dirs += 1
1671         }
1672         //if (finfo.Mode() & os.ModeSymLink) != 0
1673         if (fstat.Mode & S_IFMT) == S_IFLNK {
1674             //if verb { fmt.Printf("symlink(%v,%s)\n", fstat.Mode, finfo.Name()) }
1675             //if verb { fmt.Printf("symlink(%s,%s)\n", fstat.Mode, finfo.Name()) }
1676             fsum.SymLink += 1
1677         }
1678         return fsum
1679     }
1680 }
1681 func (gsh *GshContext)xxFindEntv(depth int, total *fileSum, dir string, dstat aStat_t, ei int, entv []string, npatv []string, argv []string)(*fileSum){
1682     nois := isin("-grep", argv)
1683     // sort entv
1684     /*
1685     if isin("-t", argv){
1686         sort.Slice(filev, func(i, j int) bool {
1687             return 0 < filev[i].ModTime().Sub(filev[j].ModTime())
1688         })
1689     }
1690     */
1691     /*
1692     if isin("-u", argv){
1693         sort.Slice(filev, func(i, j int) bool {
1694             return 0 < filev[i].AccTime().Sub(filev[j].AccTime())
1695         })
1696     }
1697     if isin("-U", argv){
1698         sort.Slice(filev, func(i, j int) bool {
1699             return 0 < filev[i].CreateTime().Sub(filev[j].CreateTime())
1700         })
1701     }
1702     */
1703     if isin("-s", argv){
1704         sort.Slice(filev, func(i, j int) bool {
1705             return filev[j].Size() < filev[i].Size()
1706         })
1707     }
1708     /*
1709     for _, filename := range entv {
1710         For _, npat := range npatv {
1711             match := true
1712             if npat == "*" {
1713                 match = true
1714             } else {
1715                 match, _ = filepath.Match(npat, filename)
1716             }
1717             path := dir + DIRSEP + filename
1718             if !match {
1719                 continue
1720             }
1721             var fstat aStat_t
1722             stater := aLstat(path, &fstat)
1723             if stater != nil {
1724                 if isin("-w", argv){fmt.Printf("ufind: %v\n", stater)}
1725                 continue;
1726             }
1727             if isin("-du", argv) && (fstat.Mode & S_IFMT) == S_IFDIR {
1728                 // should not show size of directory in "-du" mode ...
1729             } else {
1730                 if !nois && isin("-s", argv) && (!isin("-du", argv) || isin("-a", argv)) {
1731                     if isin("-du", argv) {
1732                         fmt.Printf("%d\t", fstat.Blocks/2)
1733                     }
1734                     showFileInfo(path, argv)
1735                 }
1736                 if true { // && isin("-du", argv)
1737                     total = cumFinfo(total, path, stater, fstat, argv, false)
1738                 }
1739             }
1740             /*
1741             if isin("-wc", argv) {
1742                 /*
1743                 if gsh.LastCheckSum.SumType != 0 {
1744                     gsh.xCksum(path, argv, &gsh.LastCheckSum);
1745                 }
1746                 x := isinX("-grep", argv); // -grep will be convenient like -ls
1747                 if 0 <= x && x+1 <= len(argv) { // -grep will be convenient like -ls
1748                     if !isRegFile(path){
1749                         found := gsh.xGrep(path, argv[x+1])
1750                         if 0 < found {
1751                             foundv := gsh.CmdCurrent.FoundFile
1752                             if len(foundv) < 10 {
1753                                 gsh.CmdCurrent.FoundFile =
1754                                     append(gsh.CmdCurrent.FoundFile, path)
1755                             }
1756                         }
1757                     }
1758                 }
1759                 if isin("-r0", argv) { // -d 0 in du, -depth n in find
1760                     //total.Depth += 1
1761                     if (fstat.Mode & S_IFMT) == S_IFLNK {
1762                         continue
1763                     }
1764                     if dstat.Rdev != fstat.Rdev {
1765                         fmt.Printf("--I-- don't follow differnet device %v(%v) %v(%v)\n",
1766                             dir, dstat.Rdev, path, fstat.Rdev)
1767                     }
1768                     if (fstat.Mode & S_IFMT) == S_IFDIR {
1769                         total = gsh.xxFind(depth+1, total, path, npatv, argv)
1770                     }
1771                 }
1772             }
1773             */
1774         }
1775     }
1776 }

```

```

1770     }
1771     }
1772     }
1773     }
1774     return total
1775 }
1776 func (gsh*GshContext)xxFind(depth int,total *fileSum,dir string,npatv[]string,argv[]string)(*fileSum){
1777     nols := isin("-grep",argv)
1778     dirfile,oeer := os.OpenFile(dir,os.O_RDONLY,0)
1779     if oeer == nil {
1780         //fmt.Printf("--I-- %v(%v)[%d]\n",dir,dirfile,dirfile.Fd())
1781         defer dirfile.Close()
1782     }else{
1783     }
1784     prev := *total
1785     var dstat asStat_t
1786     staterr := alstat(dir,dstat) // should be flstat
1787
1788     if staterr != nil {
1789         if !isin("-w",argv){ fmt.Printf("ufind: %v\n",staterr) }
1790         return total
1791     }
1792     //file,err := ioutil.ReadDir(dir)
1793     //_,err := ioutil.ReadDir(dir) // ReadDir() heavy and bad for huge directory
1794     /*
1795     if err != nil {
1796         if !isin("-w",argv){ fmt.Printf("ufind: %v\n",err) }
1797         return total
1798     }
1799     */
1800     if depth == 0 {
1801         total = cumFinfo(total,dir,staterr,dstat,argv,true)
1802         if !nols && !isin("-s",argv) && (!isin("-du",argv) || isin("-a",argv)) {
1803             showFileInfo(dir,argv)
1804         }
1805     }
1806     // it it is not a directory, just scan it and finish
1807
1808     for ei := 0; ; ei++ {
1809         entv,rderr := dirfile.Readdirnames(8*1024)
1810         if len(entv) == 0 || rderr != nil {
1811             //if rderr != nil { fmt.Printf("[%d] len=%d (%v)\n",ei,len(entv),rderr) }
1812             break
1813         }
1814         if 0 < ei {
1815             fmt.Printf("--I-- xxFind[%d] %d large-dir: %s\n",ei,len(entv),dir)
1816         }
1817         total = gsh.xxFindEntv(depth,total,dir,dstat,ei,entv,npatv,argv)
1818     }
1819     if isin("-du",argv) {
1820         // if in "du" mode
1821         fmt.Printf("%d\t%s\n", (total.Blocks-prev.Blocks)/2,dir)
1822     }
1823     return total
1824 }
1825 }
1826
1827 // {ufind|fu|ls} [Files] [Names] [-- Expressions]
1828 // Files is "." by default
1829 // Names is "*" by default
1830 // Expressions is "--print" by default for "ufind", or -du for "fu" command
1831 func (gsh*GshContext)xFind(argv[]string){
1832     if 0 < len(argv) && strBegins(argv[0],"?"){
1833         showFound(gsh,argv)
1834         return
1835     }
1836     if isin("-cksum",argv) || isin("-sum",argv) {
1837         gsh.lastCheckSum = CheckSum{}
1838         if isin("-sum",argv) && isin("-add",argv) {
1839             gsh.lastCheckSum.SumType = SUM_SUM64
1840         }else
1841         if isin("-sum",argv) && isin("-size",argv) {
1842             gsh.lastCheckSum.SumType = SUM_SIZE
1843         }else
1844         if isin("-sum",argv) && isin("-bsd",argv) {
1845             gsh.lastCheckSum.SumType = SUM_SUM16_BSD
1846         }else
1847         if isin("-sum",argv) && isin("-sysv",argv) {
1848             gsh.lastCheckSum.SumType = SUM_SUM16_SYSV
1849         }else
1850         if isin("-sum",argv) {
1851             gsh.lastCheckSum.SumType = SUM_SUM64
1852         }
1853     }
1854     if isin("-unix",argv) {
1855         gsh.lastCheckSum.SumType = SUM_UNIXFILE
1856         gsh.lastCheckSum.Crc32Table = *Crc32.MakeTable(CRC32UNIX)
1857     }
1858     if isin("-leee",argv){
1859         gsh.lastCheckSum.SumType = SUM_CRCIEEE
1860         gsh.lastCheckSum.Crc32Table = *Crc32.MakeTable(CRC32IEEE)
1861     }
1862     gsh.lastCheckSum.RusgAtStart = Getrusagev()
1863 }
1864 var total = fileSum{}
1865 npats := []string{}
1866 for _,v := range argv {
1867     if 0 < len(v) && v[0] != '-' {
1868         npats = append(npats,v)
1869     }
1870     if v == "/" { break }
1871     if v == "--" { break }
1872     if v == "-grep" { break }
1873     if v == "-ls" { break }
1874 }
1875 if len(npats) == 0 {
1876     npats = []string{"*"}
1877 }
1878 cwd := "."
1879 // if to be fullpath :: cwd, _ := os.Getwd()
1880 if len(npats) == 0 { npats = []string{"*"} }
1881 fusage := gsh.xxFind(0,*total,cwd,npats,argv)
1882 if gsh.lastCheckSum.SumType != 0 {
1883     var sumi uint64 = 0
1884     sum := gsh.lastCheckSum
1885     if (sum.SumType & SUM_SIZE) != 0 {
1886         sumi = uint64(sum.Size)
1887     }
1888     if (sum.SumType & SUM_SUM64) != 0 {
1889         sumi = sum.Sum64
1890     }
1891     if (sum.SumType & SUM_SUM16_SYSV) != 0 {
1892         s := uint32(sum.Sum16)
1893         r := (s & 0xFFFF) + ((s & 0xFFFFFFF) >> 16)
1894         s = (r & 0xFFFF) + (r >> 16)
1895         sum.Crc32Val = uint32(s)
1896         sumi = uint64(s)
1897     }
1898     if (sum.SumType & SUM_SUM16_BSD) != 0 {
1899         sum.Crc32Val = uint32(sum.Sum16)
1900         sumi = uint64(sum.Sum16)
1901     }
1902     if (sum.SumType & SUM_UNIXFILE) != 0 {
1903         sum.Crc32Val = byteCrc32end(sum.Crc32Val,uint64(sum.Size))
1904         sumi = uint64(byteCrc32end(sum.Crc32Val,uint64(sum.Size)))
1905     }
1906     if 1 < sum.Files {
1907         fmt.Printf("%v %v // %v / %v files, %v/file\r\n",
1908             sum,sum.Size,
1909             absSize(sum.Size),sum.Files,
1910             absSize(sum.Size/sum.Files))
1911     }else{
1912         fmt.Printf("%v %v %v\n",
1913             sumi,sum.Size,npats[0])
1914     }
1915 }
1916 if !isin("-grep",argv) {
1917     showFusage("total",fusage)
1918 }
1919 if !isin("-s",argv){
1920     hits := len(gsh.CmdCurrent.FoundFile)
1921     if 0 < hits {
1922         fmt.Printf("--I-- %d files hits // can be refered with %!df\n",
1923             hits,len(gsh.CommandHistory))
1924     }
1925 }
1926 if gsh.lastCheckSum.SumType != 0 {
1927     if isin("-cu",argv) {
1928         sum := gsh.lastCheckSum
1929         sum.Done = time.Now()
1930         gsh.lastCheckSum.RusgAtEnd = Getrusagev()
1931         elps := sum.Done.Sub(sum.Start)
1932         fmt.Printf("--cksum-size: %v (%v) / %v files, %v/file\r\n",
1933             sum.Size,absSize(sum.Size),sum.Files,absSize(sum.Size/sum.Files))
1934         nanos := int64(elps)
1935         dnanos := time.Duration(nanos)
1936         fmt.Printf("--cksum-time: %v/total, %v/file, %!f files/s, %v\r\n",
1937             abbtTime(dnanos),
1938             abbtTime(time.Duration(nanos/sum.Files)),
1939             (float64(sum.Files)*1000000000.0)/float64(nanos),
1940             absSpeed(sum.Size,nanos))
1941         diff := RusageSub(sum.RusgAtEnd,sum.RusgAtStart)
1942         fmt.Printf("--cksum-rusg: %v\n",sRusagef("",argv,diff))
1943     }
1944     return
1945 }
1946 }

```

```

1947 func showFiles(files []string){
1948     sp := ""
1949     for i, file := range files {
1950         if 0 < i { sp = " " } else { sp = "" }
1951         fmt.Printf(sp+"%s",escapeWhiteSP(file))
1952     }
1953 }
1954 func showFound(gshCtx *GshContext, argv []string){
1955     for i, v := range gshCtx.CommandHistory {
1956         if 0 < len(v.FoundFile) {
1957             fmt.Printf("%d (%d) ", i, len(v.FoundFile))
1958             if !isin("-ls", argv){
1959                 fmt.Printf("\n")
1960                 for , file := range v.FoundFile {
1961                     fmt.Printf(" //sub number?")
1962                     showFileInfo(file, argv)
1963                 }
1964             }else{
1965                 showFiles(v.FoundFile)
1966                 fmt.Printf("\n")
1967             }
1968         }
1969     }
1970 }
1971 }
1972 func showMatchFile(filev []os.FileInfo, npat, dir string, argv []string)(string, bool){
1973     fname := ""
1974     found := false
1975     for _, v := range filev {
1976         match, _ := filepath.Match(npat, (v.Name()))
1977         if match {
1978             fname = v.Name()
1979             found = true
1980             //fmt.Printf("%d %s\n", i, v.Name())
1981             showIfExecutable(fname, dir, argv)
1982         }
1983     }
1984     return fname, found
1985 }
1986 func showIfExecutable(name, dir string, argv []string)(ffullpath string, ffound bool){
1987     var fullpath string
1988     if strBegins(name, DIRSEP){
1989         fullpath = name
1990     }else{
1991         if( len(dir) == 0 ){
1992             fullpath = name;
1993         }else{
1994             fullpath = dir + DIRSEP + name
1995         }
1996     }
1997     fi, err := os.Stat(fullpath)
1998     //fmt.Printf("-Dp-- \"%s\\\"%s\\\"%s\\\" %s\n", fullpath, err);
1999     if err != nil {
2000         fullpath = ".exe";
2001         fi, err = os.Stat(fullpath)
2002     }
2003     if err != nil {
2004         fullpath = dir + DIRSEP + name + ".go"
2005         fi, err = os.Stat(fullpath)
2006     }
2007     if err == nil {
2008         fm := fi.Mode()
2009         if fm.IsRegular() {
2010             // R_OK=4, W_OK=2, X_OK=1, F_OK=0
2011             if aAccess(fullpath, 5) == nil {
2012                 ffullpath = fullpath
2013                 ffound = true
2014                 if ! isin("-s", argv) {
2015                     showFileInfo(fullpath, argv)
2016                 }
2017             }
2018         }
2019     }
2020     return ffullpath, ffound
2021 }
2022 func which(list string, argv []string) (fullpathv []string, itis bool){
2023     if len(argv) <= 1 {
2024         fmt.Printf("Usage: which comand [-s] [-a] [-ls]\n")
2025         return []string{"", false}
2026     }
2027     path := argv[1]
2028     if strBegins(path, "/") {
2029         // should check if executable?
2030         , exOK := showIfExecutable(path, "/", argv)
2031         fmt.Printf("-D- - %v exOK=%v\n", path, exOK)
2032         return []string{path}, exOK
2033     }
2034     pathenv, efound := os.LookupEnv(list)
2035     if ! efound {
2036         fmt.Printf("-D- - which: no \"%s\" environment\n", list)
2037         return []string{"", false}
2038     }
2039     //fmt.Printf("PATH=%v\n", pathenv);
2040     showall := isin("-a", argv) || 0 <= strings.Index(path, "+")
2041     dirv := strings.Split(pathenv, PATHSEP)
2042     ffound := false
2043     ffullpath := path
2044     for dir := range dirv {
2045         if 0 <= strings.Index(path, "+") { // by wild-card
2046             list, _ := ioutil.ReadDir(dir)
2047             ffullpath, ffound = showMatchFile(list, path, dir, argv)
2048         }else{
2049             ffullpath, ffound = showIfExecutable(path, dir, argv)
2050         }
2051         //if ffound && !isin("-a", argv) {
2052             if ffound && !showall {
2053                 break;
2054             }
2055     }
2056     return []string{ffullpath}, ffound
2057 }
2058 func stripLeadingWSParg(argv []string) ([]string){
2059     for ; 0 < len(argv); {
2060         if len(argv[0]) == 0 {
2061             argv = argv[1:]
2062         }else{
2063             break
2064         }
2065     }
2066     return argv
2067 }
2068 func xEval(argv []string, nlend bool){
2069     argv = stripLeadingWSParg(argv)
2070     if len(argv) == 0 {
2071         fmt.Printf("eval [%sformat] [Go-expression]\n")
2072         return
2073     }
2074     pfmt := "%v"
2075     if argv[0][0] == '=' {
2076         pfmt = argv[0]
2077         argv = argv[1:]
2078     }
2079     if len(argv) == 0 {
2080         return
2081     }
2082     gocode := strings.Join(argv, " ");
2083     //fmt.Printf("eval [%v] [%v]\n", pfmt, gocode)
2084     fsat := token.NewFileSet()
2085     rval, _ := types.Eval(fsat, nil, token.NoPos, gocode)
2086     fmt.Printf(pfmt, rval.Value)
2087     if nlend { fmt.Printf("\n") }
2088 }
2089 }
2090 func getval(name string) (found bool, val int) {
2091     /* should expand the name here */
2092     if name == "gsh.pid" {
2093         return true, os.Getpid()
2094     }else{
2095         if name == "gsh.ppid" {
2096             return true, os.Getppid()
2097         }
2098     }
2099     return false, 0
2100 }
2101 }
2102 func echo(argv []string, nlend bool){
2103     for ai := 1; ai < len(argv); ai++ {
2104         if 1 < ai {
2105             fmt.Printf(" ");
2106         }
2107         arg := argv[ai]
2108         found, val := getval(arg)
2109         if found {
2110             fmt.Printf("%d", val)
2111         }else{
2112             fmt.Printf("%s", arg)
2113         }
2114     }
2115     if nlend {
2116         fmt.Printf("\n");
2117     }
2118 }
2119 }
2120 func resfile() string {
2121     return "gsh.tmp"
2122 }
2123 //var resF *File
2124 func resmap() {

```

```

2124 //_, err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, os.ModeAppend)
2125 // https://developpaper.com/solution-to-golang-bad-file-descriptor-problem/
2126 , err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, 0600)
2127 if err != nil {
2128     fmt.Printf("refF could not open: %s\n",err)
2129 }else{
2130     fmt.Printf("refF opened\n")
2131 }
2132 }
2133 }
2134 // @@2020-0821
2135 func gshScanArg(str string,strip int)(argv []string){
2136     var si = 0
2137     var sb = 0
2138     var inBracket = 0
2139     var arg1 = make([]byte,LINESIZE)
2140     var ax = 0
2141     debug := false
2142
2143     for ; si < len(str); si++ {
2144         if str[si] != ' ' {
2145             break
2146         }
2147     }
2148     sb = si
2149     for ; si < len(str); si++ {
2150         if sb <= si {
2151             if debug {
2152                 fmt.Printf("--Da- +%d %2d-%2d %s ... %s\n",
2153                     inBracket,sb,si,arg1[0:ax],str[si:])
2154             }
2155             ch := str[si]
2156             if ch == '{' {
2157                 inBracket += 1
2158                 if 0 < strip && inBracket <= strip {
2159                     //fmt.Printf("stripLEV %d <= %d?\n",inBracket,strip)
2160                     continue
2161                 }
2162             }
2163             if 0 < inBracket {
2164                 if ch == '}' {
2165                     inBracket -= 1
2166                     if 0 < strip && inBracket < strip {
2167                         //fmt.Printf("stripLEV %d < %d?\n",inBracket,strip)
2168                         continue
2169                     }
2170                 }
2171                 arg1[ax] = ch
2172                 ax += 1
2173                 continue
2174             }
2175             if str[si] == ' ' {
2176                 argv = append(argv,string(arg1[0:ax]))
2177                 if debug {
2178                     fmt.Printf("--Da- [%v] [%v-%v] %s ... %s\n",
2179                         -1*len(argv),sb,si,str[sb:si],string(str[si:]))
2180                 }
2181                 sb = si+1
2182                 ax = 0
2183                 continue
2184             }
2185             arg1[ax] = ch
2186             ax += 1
2187         }
2188     }
2189     if sb < si {
2190         argv = append(argv,string(arg1[0:ax]))
2191         if debug {
2192             fmt.Printf("--Da- [%v] [%v-%v] %s ... %s\n",
2193                 -1*len(argv),sb,si,string(arg1[0:ax]),string(str[si:]))
2194         }
2195     }
2196     if debug {
2197         fmt.Printf("--Da- %d [%s] => [%d]\v\n",strip,si,len(argv),argv)
2198     }
2199     return argv
2200 }
2201 }
2202 // should get stderr (into tmpfile ?) and return
2203 func (gsh*GshContext)Popen(name,mode string)(pin*os.File,pout*os.File,err bool){
2204     //var pv = []int{-1,-1}
2205     //syscall.Pipe(pv)
2206
2207     xarg := gshScanArg(name,1)
2208     name = strings.Join(xarg, " ")
2209
2210     //pin = os.NewFile(uintptr(pv[0]),"StdoutOf-"+name+"")
2211     //pout = os.NewFile(uintptr(pv[1]),"StdinOf-"+name+"")
2212     pin,pout,_ = os.Pipe()
2213
2214     fdix := 0
2215     dir := ""
2216     if mode == "r" {
2217         dir = "<"
2218         fdix = 1 // read from the stdout of the process
2219     }else{
2220         dir = ">"
2221         fdix = 0 // write to the stdin of the process
2222     }
2223
2224     gshPA := gsh.gshPA
2225     savfd := gshPA.Files[fdix]
2226
2227     var fd uintptr = 0
2228     if mode == "r" {
2229         //fd = pout.Fd()
2230         //gshPA.Files[fdix] = pout.Fd()
2231     }else{
2232         //fd = pin.Fd()
2233         //gshPA.Files[fdix] = pin.Fd()
2234         gshPA.Files[fdix] = pin;
2235     }
2236     // should do this by Goroutine?
2237     if false {
2238         fmt.Printf("--Ip- Opened fd[%v] %s %v\n",fd,dir,name)
2239         fmt.Printf("--BED1 [%d,%d,%d]->[%d,%d,%d]\n",
2240             os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd(),
2241             pin.Fd(),pout.Fd(),pout.Fd())
2242     }
2243
2244     savi := os.Stdin
2245     savo := os.Stdout
2246     save := os.Stderr
2247     os.Stdin = pin
2248     os.Stdout = pout
2249     os.Stderr = pout
2250     gsh.BackGround = true
2251     gsh.gshellh(name)
2252     gsh.BackGround = false
2253     os.Stdin = savi
2254     os.Stdout = savo
2255     os.Stderr = save
2256
2257     gshPA.Files[fdix] = savfd
2258     return pin,pout,false
2259 }
2260 // <a name="ex-commands">External commands</a>
2261 func (gsh*GshContext)excommand(exec bool, argv []string) (notf bool,exit bool) {
2262     if gsh.CmdTrace { fmt.Printf("--I-- excommand[%v](%v)\n",exec,argv) }
2263
2264     gshPA := gsh.gshPA
2265     fullpath, itis := which("PATH",[]string{"which",argv[0],"-s"})
2266     if itis == false {
2267         return true,false
2268     }
2269     fullpath := fullpathv[0]
2270     argv = unescapeWhiteSP(argv)
2271     if 0 < strings.Index(fullpath,".go") {
2272         nargv := argv // []string{}
2273         gofullpath, itis := which("PATH",[]string{"which","go","-s"})
2274         if itis == false {
2275             fmt.Printf("--P-- Go not found\n")
2276             return false,true
2277         }
2278         gofullpath := gofullpathv[0]
2279         nargv = []string{ gofullpath, "run", fullpath }
2280         fmt.Printf("--I-- %s (%s %s %s)\n",gofullpath,
2281             nargv[0],nargv[1],nargv[2])
2282         if exec {
2283             syscall.Exec(gofullpath,nargv,os.Environ())
2284         }else{
2285             //pid, _ := syscall.ForkExec(gofullpath,nargv,&gshPA)
2286             proc, _ := os.StartProcess(gofullpath,nargv,&gshPA);
2287             pstat, _ := proc.Wait();
2288             pid := pstat.Pid();
2289             if gsh.BackGround {
2290                 fmt.Printf(stderr,"--Ip- in Background pid[%d]&(%v)\n",pid,len(argv),nargv)
2291                 //gsh.BackGroundJobs = append(gsh.BackGroundJobs,pid)
2292                 gsh.BackGroundJobs = append(gsh.BackGroundJobs,*pstat)
2293             }else{
2294                 /*
2295                 rusage := aRusage {}
2296                 syscall.Wait4(pid,nil,0,&rusage)
2297                 gsh.LastRusage = rusage
2298                 gsh.CmdCurrent.Rusagev[1] = rusage
2299                 */
2300             }

```

```

2301 /*
2302 gsh.LastRusage = *pstat.SysUsage().(*aRusage);
2303 gsh.CmdCurrent.Rusagev[1] = *pstat.SysUsage().(*aRusage);
2304 */
2305 aSetRusage(&gsh.LastRusage,pstat);
2306 gsh.CmdCurrent.Rusagev[1] = gsh.LastRusage;
2307 }
2308 }
2309 }else{
2310     if exec {
2311         syscall.Exec(fullpath,argv,os.Environ())
2312     }else{
2313         //pid, _ := syscall.ForkExec(fullpath,argv,&gshPA)
2314         proc, _ := os.StartProcess(fullpath,argv,&gshPA);
2315         pstat._ := proc.Wait();
2316         pid := pstat.Pid();
2317         //fmt.Printf("%d\n",pid); // 's' to be background
2318     if( false ){
2319         fmt.Printf("Sys=%v\n",gshPA.Sys);
2320     if( gshPA.Sys != nil ){
2321         //fmt.Printf("inPG=%v\n",gshPA.Sys.Foreground);
2322     }
2323     }
2324     if gsh.BackGround {
2325         fmt.Fprintf(stderr,"--Ip- in Background pid[%d]&(v)\n",pid,len(argv),argv)
2326         //gsh.BackGroundJobs = append(gsh.BackGroundJobs,pid)
2327         gsh.BackGroundJobs = append(gsh.BackGroundJobs,*pstat)
2328     }else{
2329         /*
2330         rusage := aRusage {
2331         // syscall.Wait4(pid,nil,0,rusage);
2332         gsh.LastRusage = rusage
2333         gsh.CmdCurrent.Rusagev[1] = rusage
2334         */
2335         /*
2336         gsh.LastRusage = *pstat.SysUsage().(*aRusage);
2337         gsh.CmdCurrent.Rusagev[1] = *pstat.SysUsage().(*aRusage);
2338         */
2339         aSetRusage(&gsh.LastRusage,pstat);
2340         gsh.CmdCurrent.Rusagev[1] = gsh.LastRusage;
2341     }
2342     }
2343     }
2344     return false,false
2345 }
2346 }
2347 // <a name="builtin">Builtin Commands</a>
2348 func (gshCtx *GshContext) sleep(argv []string) {
2349     if len(argv) < 2 {
2350         fmt.Printf("Sleep 100ms, 100us, 100ns, ... \n")
2351         return
2352     }
2353     duration := argv[1];
2354     d, err := time.ParseDuration(duration)
2355     if err != nil {
2356         d, err = time.ParseDuration(duration+"s")
2357     if err != nil {
2358         fmt.Printf("duration ? %s (%s)\n",duration,err)
2359         return
2360     }
2361     }
2362     //fmt.Printf("Sleep %v\n",duration)
2363     time.Sleep(d)
2364     if 0 < len(argv[2:]) {
2365         gshCtx.gshellv(argv[2:])
2366     }
2367 }
2368 func (gshCtx *GshContext)repeat(argv []string) {
2369     if len(argv) < 2 {
2370         return
2371     }
2372     start0 := time.Now()
2373     for ri, _ := strconv.Atoi(argv[1]); 0 < ri; ri-- {
2374         if 0 < len(argv[2:]) {
2375             //start := time.Now()
2376             gshCtx.gshellv(argv[2:])
2377             end := time.Now()
2378             elps := end.Sub(start0);
2379             if( 1000000000 < elps ){
2380                 fmt.Printf("repeat%d %v\n",ri,elps);
2381             }
2382         }
2383     }
2384 }
2385 }
2386 func (gshCtx *GshContext)gen(argv []string) {
2387     gshPA := gshCtx.gshPA
2388     if len(argv) < 2 {
2389         fmt.Printf("Usage: %s N\n",argv[0])
2390         return
2391     }
2392     // should be repeated by "repeat" command
2393     count, _ := strconv.Atoi(argv[1])
2394     //fd := gshPA.Files[1] // Stdout
2395     //file := os.NewFile(fd,"internalStdout")
2396     file := gshPA.Files[1]; // Stdout
2397     fmt.Printf("--I-- Gen. Count=%d to %d\n",count,file.Fd())
2398     //buf := []byte{}
2399     outdata := "0123 5678 0123 5678 0123 5678 0123 5678\r"
2400     for gi := 0; gi < count; gi++ {
2401         file.WriteString(outdata)
2402     }
2403     //file.WriteString("\n")
2404     fmt.Printf("\n(%d B)\n",count*len(outdata));
2405     //file.Close()
2406 }
2407 }
2408 // <a name="rexec">Remote Execution</a> // 2020-0820
2409 func Elapsed(from time.Time)(string){
2410     elps := time.Now().Sub(from)
2411     if 1000000000 < elps {
2412         return fmt.Sprintf("%5d.%02ds",elps/1000000000,(elps%1000000000)/10000000)
2413     }else{
2414         if 1000000 < elps {
2415             return fmt.Sprintf("[%3d.%03dms]",elps/1000000,(elps%1000000)/1000)
2416         }else{
2417             return fmt.Sprintf("[%3d.%03dus]",elps/1000,(elps%1000))
2418         }
2419     }
2420 }
2421 //func abftime(nanos int64)(string){
2422 func abftime(nanos time.Duration)(string){
2423     if 1000000000 < nanos {
2424         return fmt.Sprintf("%d.%02ds",nanos/1000000000,(nanos%1000000000)/10000000)
2425     }else{
2426         if 1000000 < nanos {
2427             return fmt.Sprintf("%d.%03dms",nanos/1000000,(nanos%1000000)/1000)
2428         }else{
2429             return fmt.Sprintf("%d.%03dus",nanos/1000,(nanos%1000))
2430         }
2431     }
2432 }
2433 func absfsize(size int64)(string){
2434     fsz := float64(size)
2435     if 1024*1024*1024 < size {
2436         return fmt.Sprintf("%.2fGiB",fsz/(1024*1024*1024))
2437     }else{
2438         if 1024*1024 < size {
2439             return fmt.Sprintf("%.3fMiB",fsz/(1024*1024))
2440         }else{
2441             return fmt.Sprintf("%.3fKiB",fsz/1024)
2442         }
2443     }
2444 }
2445 func absaize(size int64)(string){
2446     fsz := float64(size)
2447     if 1024*1024*1024 < size {
2448         return fmt.Sprintf("%.2fGiB",fsz/(1024*1024*1024))
2449     }else{
2450         if 1024*1024 < size {
2451             return fmt.Sprintf("%.3fMiB",fsz/(1024*1024))
2452         }else{
2453             return fmt.Sprintf("%.3fKiB",fsz/1024)
2454         }
2455     }
2456 }
2457 func abspspeed(totalB int64,ns int64)(string){
2458     MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
2459     if 1000 <= MBs {
2460         return fmt.Sprintf("%6.3fGB/s",MBs/1000)
2461     }
2462     if 1 <= MBs {
2463         return fmt.Sprintf("%6.3fMB/s",MBs)
2464     }else{
2465         return fmt.Sprintf("%6.3fKB/s",MBs*1000)
2466     }
2467 }
2468 func abspspeed(totalB int64,ns time.Duration)(string){
2469     MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
2470     if 1000 <= MBs {
2471         return fmt.Sprintf("%6.3fGBps",MBs/1000)
2472     }
2473     if 1 <= MBs {
2474         return fmt.Sprintf("%6.3fMbps",MBs)
2475     }else{
2476         return fmt.Sprintf("%6.3fKBps",MBs*1000)
2477     }
2478 }
2479 func fileRelay(what string,in*os.File,out*os.File,size int64,bsiz int)(wcount int64){
2480     Start := time.Now()
2481     buff := make([]byte,bsiz)

```

```

2478 var total int64 = 0
2479 var rem int64 = size
2480 nio := 0
2481 Prev := time.Now()
2482 var PrevSize int64 = 0
2483
2484 fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v) START\n",
2485   what,absize(total),size,nio)
2486
2487 for i:= 0; ; i++ {
2488   var len = bsiz
2489   if int(rem) < len {
2490     len = int(rem)
2491   }
2492   Now := time.Now()
2493   Elps := Now.Sub(Prev);
2494   if 100000000 < Now.Sub(Prev) {
2495     fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v) %s\n",
2496       what,absize(total),size,nio,
2497       abspeed((total-PrevSize),Elps))
2498     Prev = Now;
2499     PrevSize = total
2500   }
2501   rlen := len
2502   if in != nil {
2503     // should watch the disconnection of out
2504     rcc,err := in.Read(buf[0:rlen])
2505     if err != nil {
2506       fmt.Printf(Elapsed(Start)+"--En- X: %s read(%v,%v)<v\n",
2507         what,rcc,err,in.Name())
2508       break
2509     }
2510     rlen = rcc
2511     if string(buf[0:10]) == "(SoftEOF " {
2512       var ecc int64 = 0
2513       fmt.Sscanf(string(buf),"((SoftEOF %v",&ecc)
2514       fmt.Printf(Elapsed(Start)+"--En- X: %s Recv ((SoftEOF %v))\v\n",
2515         what,ecc,total)
2516       if ecc == total {
2517         break
2518       }
2519     }
2520   }
2521
2522   wlen := rlen
2523   if out != nil {
2524     wcc,err := out.Write(buf[0:wlen])
2525     if err != nil {
2526       fmt.Printf(Elapsed(Start)+"--En- X: %s write(%v,%v)>v\n",
2527         what,wcc,err,out.Name())
2528       break
2529     }
2530     wlen = wcc
2531   }
2532   if wlen < rlen {
2533     fmt.Printf(Elapsed(Start)+"--En- X: %s incomplete write (%v/%v)\n",
2534       what,wlen,rlen)
2535     break;
2536   }
2537
2538   nio += 1
2539   total += int64(rlen)
2540   rem -= int64(rlen)
2541   if rem <= 0 {
2542     break
2543   }
2544 }
2545 Done := time.Now()
2546 Elps := float64(Done.Sub(Start))/100000000 //Seconds
2547 TotalMS := float64(total)/1000000 //MS
2548 MBps := totalMS/Elps
2549 fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v) %v %v %v\n",
2550   what,total,size,nio,absize(total),MBps)
2551 return total
2552 }
2553 func tcpPush(clnt *os.File){
2554   // shrink socket buffer and recover
2555   usleep(100);
2556 }
2557 func (gsh*GshContext)RexecServer(argv[]string){
2558   debug := true
2559   Start0 := time.Now()
2560   Start := Start0
2561   // if local == "": { local = "0.0.0.0:9999" }
2562   local := "0.0.0.0:9999"
2563
2564   if 0 < len(argv) {
2565     if argv[0] == "-s" {
2566       debug = false
2567       argv = argv[1:]
2568     }
2569   }
2570   if 0 < len(argv) {
2571     argv = argv[1:]
2572   }
2573   port, err := net.ResolveTCPAddr("tcp",local);
2574   if err != nil {
2575     fmt.Printf("--En- S: Address error: %s (%s)\n",local,err)
2576     return
2577   }
2578   fmt.Printf(Elapsed(Start)+"--In- S: Listening at %s...\n",local);
2579   sconn, err := net.ListenTCP("tcp", port)
2580   if err != nil {
2581     fmt.Printf(Elapsed(Start)+"--En- S: Listen error: %s (%s)\n",local,err)
2582     return
2583   }
2584
2585   reqbuf := make([]byte,LINESIZE)
2586   res := ""
2587   for {
2588     fmt.Printf(Elapsed(Start0)+"--In- S: Listening at %s...\n",local);
2589     aconn, err := sconn.AcceptTCP()
2590     Start = time.Now()
2591     if err != nil {
2592       fmt.Printf(Elapsed(Start)+"--En- S: Accept error: %s (%s)\n",local,err)
2593       return
2594     }
2595     clnt, _ := aconn.File()
2596     fd := clnt.Fd()
2597     ar := aconn.RemoteAddr()
2598     if debug { fmt.Printf(Elapsed(Start0)+"--In- S: Accepted TCP at %s [%d] <- %v\n",
2599       local,fd,ar) }
2600     res = fmt.Sprintf("220 GShell/%s Server\r\n",VERSION)
2601     fmt.Fprintf(clnt, "%s",res)
2602     if debug { fmt.Printf(Elapsed(Start)+"--In- S: %s",res) }
2603     count, err := clnt.Read(reqbuf)
2604     if err != nil {
2605       fmt.Printf(Elapsed(Start)+"--En- C: (%v %v) %v",
2606         count,err,string(reqbuf))
2607     }
2608     req := string(reqbuf[:count])
2609     if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",string(req)) }
2610     reqv := strings.Split(string(req),"\r")
2611     cmdv := gshScanArg(reqv[0],0)
2612     //cmdv := strings.Split(reqv[0]," ")
2613     switch cmdv[0]{
2614     case "HELO":
2615       res = fmt.Sprintf("250 %v",req)
2616     case "GET":
2617       // download {remotefile|-zN} [localfile]
2618       var dszize int64 = 32*1024*1024
2619       var bszize int = 64*1024
2620       var fname string = ""
2621       var in *os.File = nil
2622       var pseudoEOF = false
2623       if 1 <= len(cmdv) {
2624         fname = cmdv[1]
2625         if strBegins(fname,"-z") {
2626           fmt.Sscanf(fname[2:], "%d",&dszize)
2627         }else{
2628           if strBegins(fname,"(") {
2629             xin,xout,err := gsh.Popen(fname,"r")
2630             if err {
2631               }else{
2632                 xout.Close()
2633                 defer xin.Close()
2634                 in = xin
2635                 dszize = MaxStreamSize
2636                 pseudoEOF = true
2637               }
2638             }else{
2639               xin,err := os.Open(fname)
2640               if err != nil {
2641                 fmt.Printf("--En- GET (%v)\n",err)
2642               }else{
2643                 defer xin.Close()
2644                 in = xin
2645                 fi, _ := xin.Stat()
2646                 dszize = fi.Size()
2647               }
2648             }
2649           }
2650           //fmt.Printf(Elapsed(Start)+"--In- GET %v:%v\n",dszize,bszize)
2651           res = fmt.Sprintf("200 %v\r\n",dszize)
2652           fmt.Fprintf(clnt, "%s",res)
2653           tcpPush(clnt); // should be separated as line in receiver
2654           fmt.Printf(Elapsed(Start)+"--In- S: %v",res)

```

```

2655 wcount := fileRelay("SendGET",in,clnt,dsize,bsize)
2656 if pseudoEOF {
2657     in.Close() // pipe from the command
2658     // show end of stream data (its size) by OOB?
2659     SoftEOF := fmt.Sprintf("(SoftEOF %v)",wcount)
2660     fmt.Printf(Elapsed(Start)+"--In- S: Send %v\n",SoftEOF)
2661 }
2662 tcpPush(clnt); // to let SoftEOF data appear at the top of received data
2663 fmt.Printf(clnt, "%v\n", SoftEOF)
2664 tcpPush(clnt); // to let SoftEOF alone in a packet (separate with 200 OK)
2665 // with client generated random?
2666 //fmt.Printf("--In- L: close %v (%v)\n",in.Fd(),in.Name())
2667 }
2668 res = fmt.Sprintf("200 GET done\r\n")
2669 case "PUT":
2670     // upload {srcfile|-zN} [dstfile]
2671     var dsize int64 = 32*1024*1024
2672     var bsize int = 64*1024
2673     var fname string = ""
2674     var out *os.File = nil
2675     if 1 < len(cmdv) { // Localfile
2676         fmt.Sscanf(cmdv[1],"%d",&dsize)
2677     }
2678     if 2 < len(cmdv) {
2679         fname = cmdv[2]
2680         if fname == "-" {
2681             // nul dev
2682         } else
2683         if strBegins(fname,"{") {
2684             xin,xout,err := gsh.Popen(fname,"w")
2685             if err {
2686                 }else{
2687                 xin.Close()
2688                 defer xout.Close()
2689                 out = xout
2690             }
2691         }else{
2692             // should write to temporary file
2693             // should suppress "c on tty
2694             xout,err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0600)
2695             //fmt.Printf("--In- S: open(%v) out(%v) err(%v)\n",fname,xout,err)
2696             if err != nil {
2697                 }else{
2698                 }else{
2699                 }else{
2700                 }else{
2701                 }else{
2702                 }else{
2703                 }else{
2704                 }else{
2705                 }else{
2706                 }else{
2707                 }else{
2708                 }else{
2709                 }else{
2710                 }else{
2711                 }else{
2712                 }else{
2713                 }else{
2714                 }else{
2715                 }else{
2716                 }else{
2717                 }else{
2718                 }else{
2719                 }else{
2720                 }else{
2721                 }else{
2722                 }else{
2723                 }else{
2724                 }else{
2725                 }else{
2726                 }else{
2727                 }else{
2728                 }else{
2729                 }else{
2730                 }else{
2731                 }else{
2732                 }else{
2733                 }else{
2734                 }else{
2735                 }else{
2736                 }else{
2737                 }else{
2738                 }else{
2739                 }else{
2740                 }else{
2741                 }else{
2742                 }else{
2743                 }else{
2744                 }else{
2745                 }else{
2746                 }else{
2747                 }else{
2748                 }else{
2749                 }else{
2750                 }else{
2751                 }else{
2752                 }else{
2753                 }else{
2754                 }else{
2755                 }else{
2756                 }else{
2757                 }else{
2758                 }else{
2759                 }else{
2760                 }else{
2761                 }else{
2762                 }else{
2763                 }else{
2764                 }else{
2765                 }else{
2766                 }else{
2767                 }else{
2768                 }else{
2769                 }else{
2770                 }else{
2771                 }else{
2772                 }else{
2773                 }else{
2774                 }else{
2775                 }else{
2776                 }else{
2777                 }else{
2778                 }else{
2779                 }else{
2780                 }else{
2781                 }else{
2782                 }else{
2783                 }else{
2784                 }else{
2785                 }else{
2786                 }else{
2787                 }else{
2788                 }else{
2789                 }else{
2790                 }else{
2791                 }else{
2792                 }else{
2793                 }else{
2794                 }else{
2795                 }else{
2796                 }else{
2797                 }else{
2798                 }else{
2799                 }else{
2800                 }else{
2801                 }else{
2802                 }else{
2803                 }else{
2804                 }else{
2805                 }else{
2806                 }else{
2807                 }else{
2808                 }else{
2809                 }else{
2810                 }else{
2811                 }else{
2812                 }else{
2813                 }else{
2814                 }else{
2815                 }else{
2816                 }else{
2817                 }else{
2818                 }else{
2819                 }else{
2820                 }else{
2821                 }else{
2822                 }else{
2823                 }else{
2824                 }else{
2825                 }else{
2826                 }else{
2827                 }else{
2828                 }else{
2829                 }else{
2830                 }else{
2831                 }else{

```



```

2832         if err {
2833         }else{
2834             xout.Close()
2835             defer xin.Close()
2836             //in = xin
2837             local = xin
2838             fmt.Printf("--In- [%d] < Upload output of %v\n",
2839                 local.Fd(),fname)
2840             ofile = "-from."+fname
2841             dsize = MaxStreamSize
2842         }
2843     }else{
2844         xlocal,err := os.Open(fname)
2845         if err != nil {
2846             fmt.Printf("--En- (%s)\n",err)
2847             local = nil
2848         }else{
2849             local = xlocal
2850             fi,_ := local.Stat()
2851             dsize = fi.Size()
2852             defer local.Close()
2853             //fmt.Printf("--L- Rex in(%v / %v)\n",ofile,dsize)
2854         }
2855         ofile = fname
2856         fmt.Printf(Elapsed(Start)+"--In- L: open(%v,r)=%v %v (%v)\n",
2857             fname,dsize,local,err)
2858     }
2859 }
2860 if 2 < len(argv) && argv[2] != "" {
2861     ofile = argv[2]
2862     //fmt.Printf("(%d)%v B.ofile=%v\n",len(argv),argv,ofile)
2863 }
2864 //fmt.Printf(Elapsed(Start)+"--I- Rex out(%v)\n",ofile)
2865 fmt.Printf(Elapsed(Start)+"--In- PUT %v (/%v)\n",dsize,bsize)
2866 req = fmt.Sprintf("PUT %v %v \n",dsize,ofile)
2867 if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",req) }
2868 fmt.Fprintf(serv,"%v",req)
2869 count,err = serv.Read(res)
2870 if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res[0:count])) }
2871 fileRelay("SendPUT",local,remote,dsize,bsize)
2872 }else{
2873     req = fmt.Sprintf("%v\n",strings.Join(argv, " "))
2874     if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",req) }
2875     fmt.Fprintf(serv,"%v",req)
2876     //fmt.Printf("--In- sending RexRequest(%v)\n",len(req))
2877 }
2878 //fmt.Printf(Elapsed(Start)+"--In- waiting RexResponse...\n")
2879 count,err = serv.Read(res)
2880 res = ""
2881 if count == 0 {
2882     res = "(nil)\r\n"
2883 }else{
2884     res = string(res[:count])
2885 }
2886 if err != nil {
2887     fmt.Printf(Elapsed(Start)+"--En- S: (%d,%v) %v",count,err,res)
2888 }else{
2889     fmt.Printf(Elapsed(Start)+"--In- S: %v",res)
2890 }
2891 serv.Close()
2892 //conn.Close()
2893 }
2894 var stat string
2895 var rcode int
2896 fmt.Sscanf(res,"%d %s",&rcode,&stat)
2897 //fmt.Printf("--D- Client: %v (%v)",rcode,stat)
2898 return rcode,res
2899 }
2900
2901 // <a name="remote-sh">Remote Shell</a>
2902 // gcp file {...} [host]:[port]:[dir] | dir } // -p | -no-p
2903 func (gsh*GshContext)FileCopy(argv[]string){
2904     var host = ""
2905     var port = ""
2906     var upload = false
2907     var download = false
2908     var xargv = []string{"rex-gcp"}
2909     var srcv = []string{}
2910     var dstv = []string{}
2911     argv = argv[1:]
2912
2913     for _,v := range argv {
2914         /*
2915         if v[0] == '-' { // might be a pseudo file (generated date)
2916             continue
2917         }
2918         */
2919         obj := strings.Split(v,":")
2920         //fmt.Printf("%d %v %v\n",len(obj),v,obj)
2921         if 1 < len(obj) {
2922             host = obj[0]
2923             file = ""
2924             if 0 < len(host) {
2925                 gsh.LastServer.host = host
2926             }else{
2927                 host = gsh.LastServer.host
2928                 port = gsh.LastServer.port
2929             }
2930             if 2 < len(obj) {
2931                 port = obj[1]
2932                 if 0 < len(port) {
2933                     gsh.LastServer.port = port
2934                 }else{
2935                     port = gsh.LastServer.port
2936                 }
2937                 file = obj[2]
2938             }else{
2939                 file = obj[1]
2940             }
2941             if len(srcv) == 0 {
2942                 download = true
2943                 srcv = append(srcv,file)
2944             }
2945             continue
2946         }
2947         upload = true
2948         dstv = append(dstv,file)
2949         continue
2950     }
2951     /*
2952     idx := strings.Index(v,":")
2953     if 0 < idx {
2954         remote = v[0:idx]
2955         if len(srcv) == 0 {
2956             download = true
2957             srcv = append(srcv,v[idx+1:])
2958         }
2959         upload = true
2960         dstv = append(dstv,v[idx+1:])
2961         continue
2962     }
2963     */
2964     if download {
2965         dstv = append(dstv,v)
2966     }else{
2967         srcv = append(srcv,v)
2968     }
2969 }
2970 hostport := "@" + host + ":" + port
2971 if upload {
2972     if host != "" { xargv = append(xargv,hostport) }
2973     xargv = append(xargv,"PUT")
2974     xargv = append(xargv,srcv[0:]...)
2975     xargv = append(xargv,dstv[0:]...)
2976     //fmt.Printf("--I- FileCopy PUT gsh://%s/%v < %v // %v\n",hostport,dstv,srcv,xargv)
2977     fmt.Printf("--I- FileCopy PUT gsh://%s/%v < %v\n",hostport,dstv,srcv)
2978     gsh.RexecClient(xargv)
2979 }else{
2980     if download {
2981         if host != "" { xargv = append(xargv,hostport) }
2982         xargv = append(xargv,"GET")
2983         xargv = append(xargv,srcv[0:]...)
2984         xargv = append(xargv,dstv[0:]...)
2985         //fmt.Printf("--I- FileCopy GET gsh://%v/%v > %v // %v\n",hostport,srcv,dstv,xargv)
2986         fmt.Printf("--I- FileCopy GET gsh://%v/%v > %v\n",hostport,srcv,dstv)
2987         gsh.RexecClient(xargv)
2988     }else{
2989     }
2990 }
2991 }
2992 // target
2993 func (gsh*GshContext)Trelpath(rloc string)(string){
2994     cwd,_ := os.Getwd()
2995     os.Chdir(gsh.RWD)
2996     os.Chdir(rloc)
2997     twd,_ := os.Getwd()
2998     os.Chdir(cwd)
2999 }
3000 tpath := twd + "/" + rloc
3001 return tpath
3002 }
3003 // join to rremote GShell - [user@]host[:port] or cd host[:port]:path
3004 func (gsh*GshContext)Rjoin(argv[]string){
3005     if len(argv) <= 1 {
3006         fmt.Printf("--I- current server = %v\n",gsh.RSERV)
3007         return
3008     }

```

```

3009 serv := argv[1]
3010 servv := strings.Split(serv,":")
3011 if l <= len(servv) {
3012     if servv[0] == "lo" {
3013         servv[0] = "localhost"
3014     }
3015 }
3016 switch len(servv) {
3017     case 1:
3018         //if strings.Index(serv,":") < 0 {
3019             serv = servv[0] + ":" + fmt.Sprintf("%d",GSH_PORT)
3020         //}
3021     case 2: // host:port
3022         serv = strings.Join(servv,":")
3023 }
3024 xargv := []string{"rex-join", "@"+serv, "HELO"}
3025 rcode,stat := gsh.RexecClient(xargv)
3026 if (rcode / 100) == 2 {
3027     fmt.Printf("--I-- OK Joined (%v) %v\n",rcode,stat)
3028     gsh.RSERV = serv
3029 }else{
3030     fmt.Printf("--I-- NG, could not joined (%v) %v\n",rcode,stat)
3031 }
3032 }
3033 func (gsh*GshContext)Rexec(argv []string){
3034     if len(argv) <= 1 {
3035         fmt.Printf("--I-- rexec command [ | {file | | {command} ]\n",gsh.RSERV)
3036         return
3037     }
3038 }
3039 /*
3040 nargv := gshScanArg(strings.Join(argv, " "),0)
3041 fmt.Printf("--D-- nargc=%d [%v]\n",len(nargv),nargv)
3042 if nargv[1][0] != '{' {
3043     nargv[1] = "{" + nargv[1] + "}"
3044     fmt.Printf("--D-- nargc=%d [%v]\n",len(nargv),nargv)
3045 }
3046 argv = nargv
3047 */
3048 nargv := []string{}
3049 nargv = append(nargv, "{"+strings.Join(argv[1:], " ")+"}")
3050 fmt.Printf("--D-- nargc=%d [%v]\n",len(nargv),nargv)
3051 argv = nargv
3052 }
3053 xargv := []string{"rex-exec", "@"+gsh.RSERV, "GET"}
3054 xargv = append(xargv,argv...)
3055 xargv = append(xargv, "/dev/tty")
3056 rcode,stat := gsh.RexecClient(xargv)
3057 if (rcode / 100) == 2 {
3058     fmt.Printf("--I-- OK Rexec (%v) %v\n",rcode,stat)
3059 }else{
3060     fmt.Printf("--I-- NG Rexec (%v) %v\n",rcode,stat)
3061 }
3062 }
3063 func (gsh*GshContext)Rchdir(argv []string){
3064     if len(argv) <= 1 {
3065         return
3066     }
3067     cwd, _ := os.Getwd()
3068     os.Chdir(gsh.RWD)
3069     os.Chdir(argv[1])
3070     twd, _ := os.Getwd()
3071     gsh.RWD = twd
3072     fmt.Printf("--I-- JWD=%v\n",twd)
3073     os.Chdir(cwd)
3074 }
3075 func (gsh*GshContext)Rpwd(argv []string){
3076     fmt.Printf("%v\n",gsh.RWD)
3077 }
3078 func (gsh*GshContext)Rls(argv []string){
3079     cwd, _ := os.Getwd()
3080     os.Chdir(gsh.RWD)
3081     argv[0] = "-ls"
3082     gsh.xFind(argv)
3083     os.Chdir(cwd)
3084 }
3085 func (gsh*GshContext)Rput(argv []string){
3086     var local string = ""
3087     var remote string = ""
3088     if l < len(argv) {
3089         local = argv[1]
3090         remote = local // base name
3091     }
3092     if 2 < len(argv) {
3093         remote = argv[2]
3094     }
3095     fmt.Printf("--I-- jput from=%v to=%v\n",local,gsh.Trelpath(remote))
3096 }
3097 func (gsh*GshContext)Rget(argv []string){
3098     var remote string = ""
3099     var local string = ""
3100     if l < len(argv) {
3101         remote = argv[1]
3102         local = remote // base name
3103     }
3104     if 2 < len(argv) {
3105         local = argv[2]
3106     }
3107     fmt.Printf("--I-- jget from=%v to=%v\n",gsh.Trelpath(remote),local)
3108 }
3109 }
3110 // <a name="network">network</a>
3111 // -s, -s= bi-directional, source, sync (maybe socket)
3112 func (gshCtx*GshContext)sconnect(inTCP bool, argv []string) {
3113     gshPA := gshCtx.gshPA
3114     if len(argv) < 2 {
3115         fmt.Printf("Usage: -s [host]:[port[.udp]]\n")
3116         return
3117     }
3118     remote := argv[1]
3119     if remote == ":" { remote = "0.0.0.0:9999" }
3120 }
3121 if inTCP { // TCP
3122     dport, err := net.ResolveTCPAddr("tcp",remote);
3123     if err != nil {
3124         fmt.Printf("Address error: %s (%s)\n",remote,err)
3125         return
3126     }
3127     conn, err := net.DialTCP("tcp",nil,dport)
3128     if err != nil {
3129         fmt.Printf("Connection error: %s (%s)\n",remote,err)
3130         return
3131     }
3132     file, _ := conn.File();
3133     //fd := file.Fd()
3134     //fmt.Printf("Socket: connected to %s, socket[%d]\n",remote,fd)
3135     fmt.Printf("Socket: connected to %s, socket[%d]\n",remote,file.Fd())
3136 }
3137 savfd := gshPA.Files[1]
3138 //gshPA.Files[1] = fd;
3139 gshPA.Files[1] = file;
3140 gshCtx.gshellv(argv[2:])
3141 gshPA.Files[1] = savfd
3142 file.Close()
3143 conn.Close()
3144 }else{
3145     //dport, err := net.ResolveUDPAddr("udp4",remote);
3146     dport, err := net.ResolveUDPAddr("udp",remote);
3147     if err != nil {
3148         fmt.Printf("Address error: %s (%s)\n",remote,err)
3149         return
3150     }
3151     //conn, err := net.DialUDP("udp4",nil,dport)
3152     conn, err := net.DialUDP("udp",nil,dport)
3153     if err != nil {
3154         fmt.Printf("Connection error: %s (%s)\n",remote,err)
3155         return
3156     }
3157     file, _ := conn.File();
3158     //fd := file.Fd()
3159 }
3160 ar := conn.RemoteAddr()
3161 //al := conn.LocalAddr()
3162 fmt.Printf("Socket: connected to %s [%s], socket[%d]\n",
3163 //remote,ar.String(),fd)
3164 remote,ar.String(),file.Fd())
3165 }
3166 savfd := gshPA.Files[1]
3167 //gshPA.Files[1] = fd;
3168 gshPA.Files[1] = file;
3169 gshCtx.gshellv(argv[2:])
3170 gshPA.Files[1] = savfd
3171 file.Close()
3172 conn.Close()
3173 }
3174 }
3175 func (gshCtx*GshContext)saccept(inTCP bool, argv []string) {
3176     gshPA := gshCtx.gshPA
3177     if len(argv) < 2 {
3178         fmt.Printf("Usage: -ac [host]:[port[.udp]]\n")
3179         return
3180     }
3181     local := argv[1]
3182     if local == ":" { local = "0.0.0.0:9999" }
3183     if inTCP { // TCP
3184         port, err := net.ResolveTCPAddr("tcp",local);
3185         if err != nil {

```

```

3186     fmt.Printf("Address error: %s (%s)\n",local,err)
3187     return
3188 }
3189 //fmt.Printf("Listen at %s...\n",local);
3190 sconn, err := net.ListenTCP("tcp", port)
3191 if err != nil {
3192     fmt.Printf("Listen error: %s (%s)\n",local,err)
3193     return
3194 }
3195 //fmt.Printf("Accepting at %s...\n",local);
3196 aconn, err := sconn.AcceptTCP()
3197 if err != nil {
3198     fmt.Printf("Accept error: %s (%s)\n",local,err)
3199     return
3200 }
3201 file, _ := aconn.File()
3202 //fd := file.Fd()
3203 //fmt.Printf("Accepted TCP at %s [%d]\n",local,fd)
3204 fmt.Printf("Accepted TCP at %s [%d]\n",local,file.Fd())
3205
3206 savfd := gshPA.Files[0]
3207 //gshPA.Files[0] = fd;
3208 gshPA.Files[0] = file;
3209 gshCtx.gsheliv(argv[2:])
3210 gshPA.Files[0] = savfd
3211
3212 sconn.Close();
3213 aconn.Close();
3214 file.Close();
3215 }else{
3216 //port, err := net.ResolveUDPAddr("udp",local);
3217 port, err := net.ResolveUDPAddr("udp",local);
3218 if err != nil {
3219     fmt.Printf("Address error: %s (%s)\n",local,err)
3220     return
3221 }
3222 //fmt.Printf("Listen UDP at %s...\n",local);
3223 //uconn, err := net.ListenUDP("udp4", port)
3224 uconn, err := net.ListenUDP("udp", port)
3225 if err != nil {
3226     fmt.Printf("Listen error: %s (%s)\n",local,err)
3227     return
3228 }
3229 file, _ := uconn.File()
3230 //fd := file.Fd()
3231 ar := uconn.RemoteAddr()
3232 remote := ""
3233 if ar != nil { remote = ar.String() }
3234 if remote == "" { remote = "?" }
3235
3236 // not yet received
3237 //fmt.Printf("Accepted at %s [%d] <- %s\n",local,fd,"")
3238
3239 savfd := gshPA.Files[0]
3240 //gshPA.Files[0] = fd;
3241 gshPA.Files[0] = file;
3242 savenv := gshPA.Env
3243 gshPA.Env = append(savenv, "REMOTE_HOST="+remote)
3244 gshCtx.gsheliv(argv[2:])
3245 gshPA.Env = savenv
3246 gshPA.Files[0] = savfd
3247
3248 uconn.Close();
3249 file.Close();
3250 }
3251 }
3252
3253 // empty line command
3254 func (gshCtx*GshContext)xPwd(argv[]string){
3255 // execute context command, pwd + date
3256 // const notation, representation scheme, to be resumed at re-login
3257 cwd, _ := os.Getwd()
3258 switch {
3259 case isin("-a",argv):
3260     gshCtx.ShowChdirHistory(argv)
3261 case isin("-ls",argv):
3262     showFileInfo(cwd,argv)
3263 default:
3264     fmt.Printf("%s\n",cwd)
3265 case isin("-v",argv): // obsolete empty command
3266     t := time.Now()
3267     date := t.Format(time.UnixDate)
3268     exe, _ := os.Executable()
3269     host, _ := os.Hostname()
3270     fmt.Printf("PWD=\"%s\",cwd\n",cwd)
3271     fmt.Printf("HOST=\"%s\",host\n",host)
3272     fmt.Printf("DATE=\"%s\",date\n",date)
3273     fmt.Printf("TIME=\"%s\",t.String()\n",t.String())
3274     fmt.Printf("PID=\"%d\",os.Getpid()\n",os.Getpid())
3275     fmt.Printf("EXE=\"%s\",exe\n",exe)
3276     fmt.Printf("\n")
3277 }
3278 }
3279
3280 // <a name="history">History</a>
3281 // these should be browsed and edited by HTTP browser
3282 // show the time of command with -t and directoty with -ls
3283 // openfile-history, sort by -a -m -c
3284 // sort by elapsed time by -t -s
3285 // search by "more" like interface
3286 // edit history
3287 // sort history, and we or uniq
3288 // CPU and other resource consumptions
3289 // limit showing range (by time or so)
3290 // export / import history
3291 func (gshCtx *GshContext)xHistory(argv []string){
3292 atWorkDirX := -1
3293 if 1 < len(argv) && strBegins(argv[1],"@") {
3294     atWorkDirX, _ = strconv.Atoi(argv[1][1:])
3295 }
3296 //fmt.Printf("--D-- showHistory(%v)\n",argv)
3297 for i, v := range gshCtx.CommandHistory {
3298 // exclude commands not to be listed by default
3299 // internal commands may be suppressed by default
3300 if v.CmdLine == "" && !isin("-a",argv) {
3301     continue;
3302 }
3303 if 0 <= atWorkDirX {
3304     if v.WorkDirX != atWorkDirX {
3305         continue
3306     }
3307 }
3308 if !isin("-n",argv){ // like "fc"
3309     fmt.Printf("%s-%d ",i)
3310 }
3311 if !isin("-v",argv){ // should be with it date
3312     fmt.Println(v)
3313 }else{
3314     if !isin("-l",argv) || !isin("-10",argv) {
3315         elps := v.EndAt.Sub(v.StartAt);
3316         start := v.StartAt.Format(time.Stamp)
3317         fmt.Printf("%d ",v.WorkDir)
3318         fmt.Printf("[%v] %11v/t ",start,elps)
3319     }
3320     if !isin("-l",argv) && !isin("-10",argv){
3321         fmt.Printf("%v",Rusagef("%t %u/t// %s",argv,v.Rusagev))
3322     }
3323     if !isin("-at",argv) { // !isin("-ls",argv){
3324         dhi := v.WorkDirX // workdir history index
3325         fmt.Printf("%d %t",dhi,v.WorkDir)
3326         // show the FileInfo of the output command??
3327     }
3328     fmt.Printf("%s",v.CmdLine)
3329     fmt.Printf("\n")
3330 }
3331 }
3332 }
3333 // In - history index
3334 func searchHistory(gshCtx GshContext, gline string) (string, bool, bool){
3335 if gline[0] == '!' {
3336     hix, err := strconv.Atoi(gline[1:])
3337     if err != nil {
3338         fmt.Printf("--E-- (%s : range)\n",hix)
3339         return "", false, true
3340     }
3341     if hix < 0 || len(gshCtx.CommandHistory) <= hix {
3342         fmt.Printf("--E-- (%d : out of range)\n",hix)
3343         return "", false, true
3344     }
3345     return gshCtx.CommandHistory[hix].CmdLine, false, false
3346 }
3347 // search
3348 //for i, v := range gshCtx.CommandHistory {
3349 //}
3350 return gline, false, false
3351 }
3352 func (gsh*GshContext)cmdStringInHistory(hix int)(cmd string, ok bool){
3353 if 0 <= hix && hix < len(gsh.CommandHistory) {
3354     return gsh.CommandHistory[hix].CmdLine,true
3355 }
3356 return "",false
3357 }
3358
3359 // temporary adding to PATH environment
3360 // cd name -lib for LD_LIBRARY_PATH
3361 // chdir with directory history (date + full-path)
3362 // -s for sort option (by visit date or so)

```

```

3363 func (gsh*GshContext)ShowChdirHistory(i int,v GChdirHistory, argv []string){
3364     fmt.Printf("%d\n",v.CmdIndex) // the first command at this WorkDir
3365     fmt.Printf("%d\n",i)
3366     fmt.Printf("%v\n",v.MovedAt.Format(time.Stamp))
3367     showFileInfo(v.Dir,argv)
3368 }
3369 func (gsh*GshContext)ShowChdirHistory(argv []string){
3370     for i, v := range gsh.CkdirHistory {
3371         gsh.ShowChdirHistory(i,v,argv)
3372     }
3373 }
3374 func skipOpts(argv[]string)(int){
3375     for i,v := range argv {
3376         if strBegins(v, "-") {
3377             }else{
3378                 return i
3379             }
3380     }
3381     return -1
3382 }
3383 func (gshCtx*GshContext)xChdir(argv []string){
3384     cdhist := gshCtx.CkdirHistory
3385     if !isIn("?",argv) || !isIn("-t",argv) || !isIn("-a",argv) {
3386         gshCtx.ShowChdirHistory(argv)
3387         return
3388     }
3389     cwd, _ := os.Getwd()
3390     dir := ""
3391     if len(argv) <= 1 {
3392         dir = toFullPath("-")
3393     }else{
3394         i := skipOpts(argv[1:])
3395         if i < 0 {
3396             dir = toFullPath("-")
3397         }else{
3398             dir = argv[1+i]
3399         }
3400     }
3401     if strBegins(dir,"0") {
3402         if dir == "00" { // obsolete
3403             dir = gshCtx.StartDir
3404         }else{
3405             if dir == "01" {
3406                 index := len(cdhist) - 1
3407                 if 0 < index { index -= 1 }
3408                 dir = cdhist[index].Dir
3409             }else{
3410                 index, err := strconv.Atoi(dir[1:])
3411                 if err != nil {
3412                     fmt.Printf("--E-- xChdir(%v)\n",err)
3413                     dir = "?"
3414                 }else{
3415                     if len(gshCtx.CkdirHistory) <= index {
3416                         fmt.Printf("--E-- xChdir(history range error)\n")
3417                         dir = "?"
3418                     }else{
3419                         dir = cdhist[index].Dir
3420                     }
3421                 }
3422             }
3423         if dir != "?" {
3424             err := os.Chdir(dir)
3425             if err != nil {
3426                 fmt.Printf("--E-- xChdir(%s)(%v)\n",argv[1],err)
3427             }else{
3428                 cwd, _ := os.Getwd()
3429                 if cwd != pwd {
3430                     hist1 := GChdirHistory { }
3431                     hist1.Dir = cwd
3432                     hist1.MovedAt = time.Now()
3433                     hist1.CmdIndex = len(gshCtx.CommandHistory)+1
3434                     gshCtx.CkdirHistory = append(cdhist,hist1)
3435                     if !isIn("-s",argv){
3436                         //cwd, _ := os.Getwd()
3437                         //fmt.Printf("%s\n",cwd)
3438                         ix := len(gshCtx.CkdirHistory)-1
3439                         gshCtx.ShowChdirHistory1(ix,hist1,argv)
3440                     }
3441                 }
3442             }
3443         if !isIn("-ls",argv){
3444             cwd, _ := os.Getwd()
3445             showFileInfo(cwd,argv);
3446         }
3447     }
3448 }
3449 func TimeValSub(tv1 *time.Duration, tv2 *time.Duration){
3450     //tv1 := syscall.NsecToTimeval(tv1.Nano()) - tv2.Nano()
3451     *tv1 -= tv2;
3452 }
3453 func RusageSubv(ru1, ru2 [2]aRusage){[2]aRusage){
3454     TimeValSub(&ru1[0].Utime,&ru2[0].Utime)
3455     TimeValSub(&ru1[0].Stime,&ru2[0].Stime)
3456     TimeValSub(&ru1[1].Utime,&ru2[1].Utime)
3457     TimeValSub(&ru1[1].Stime,&ru2[1].Stime)
3458     return ru1
3459 }
3460 func TimeValAdd(tv1 time.Duration, tv2 time.Duration)(time.Duration){
3461     //tv1 := syscall.NsecToTimeval(tv1.Nano()) + tv2.Nano()
3462     tv1 += tv2;
3463     return tv1;
3464 }
3465 /*
3466 func RusageAddv(ru1, ru2 [2]aRusage){[2]aRusage){
3467     TimeValAdd(&ru1[0].Utime,&ru2[0].Utime)
3468     TimeValAdd(&ru1[0].Stime,&ru2[0].Stime)
3469     TimeValAdd(&ru1[1].Utime,&ru2[1].Utime)
3470     TimeValAdd(&ru1[1].Stime,&ru2[1].Stime)
3471     return ru1
3472 }
3473 */
3474 // <a name="rusage">Resource Usage</a>
3475 func rUsageOf(fmtspeg string, argv []string, ru [2]aRusage)(string){
3476     // ru[0] self , ru[1] children
3477     ut := TimeValAdd(&ru[0].Utime,&ru[1].Utime)
3478     st := TimeValAdd(&ru[0].Stime,&ru[1].Stime)
3479     //uu := (int64(ut.Sec)*1000000 + int64(ut.Usec)) * 1000
3480     //su := (int64(st.Sec)*1000000 + int64(st.Usec)) * 1000
3481     uu := ut // in nano sec
3482     su := st // in nano sec
3483     tu := uu + su
3484     ret := fmt.Sprintf("%v/sum",abtime(tu))
3485     ret = fmt.Sprintf(", %v/usr",abtime(uu))
3486     ret = fmt.Sprintf(", %v/sys",abtime(su))
3487     return ret
3488 }
3489 func RusageOf(fmtspeg string, argv []string, ru [2]aRusage)(string){
3490     ut := TimeValAdd(&ru[0].Utime,&ru[1].Utime)
3491     st := TimeValAdd(&ru[0].Stime,&ru[1].Stime)
3492     //fmt.Printf("%d.%06ds/u ",ut.Sec,ut.Usec) //ru[1].Utime.Sec,ru[1].Utime.Usec)
3493     //fmt.Printf("%d.%06ds/s ",st.Sec,st.Usec) //ru[1].Stime.Sec,ru[1].Stime.Usec)
3494     fmt.Printf("%d.%06ds/u ",ut/1000000000,(ut/1000)*1000000);
3495     fmt.Printf("%d.%06ds/s ",st/1000000000,(st/1000)*1000000);
3496     return ""
3497 }
3498 }
3499 }
3500 func GetRusagev(){[2]aRusage){
3501     var ruv = [2]aRusage{}
3502     aGetRusage(aRUSAGE_SELF,&ruv[0])
3503     aGetRusage(aRUSAGE_CHILDREN,&ruv[1])
3504     return ruv
3505 }
3506 func (gshCtx *GshContext)xTime(argv[]string)(bool){
3507     if 2 <= len(argv){
3508         gshCtx.LastRusage = aRusage{}
3509         rusagev1 := GetRusagev()
3510         fin := gshCtx.gshellv(argv[1:])
3511         rusagev2 := GetRusagev()
3512         showRusage(argv[1],argv,&gshCtx.LastRusage)
3513         rusagev := RusageSubv(rusagev2,rusagev1)
3514         showRusage("self",argv,rusagev[0])
3515         showRusage("chld",argv,rusagev[1])
3516         return fin
3517     }else{
3518         rusage := aRusage { }
3519         aGetRusage(aRUSAGE_SELF,&rusage)
3520         showRusage("self",argv, &rusage)
3521         aGetRusage(aRUSAGE_CHILDREN,&rusage)
3522         showRusage("chld",argv, &rusage)
3523         return false
3524     }
3525 }
3526 func (gshCtx *GshContext)xJobs(argv[]string){
3527     fmt.Printf("%d Jobs\n",len(gshCtx.BackGroundJobs))
3528     for ji, pid := range gshCtx.BackGroundJobs {
3529         //wstat := syscall.WaitStatus {0}
3530         rusage := aRusage { }
3531         //wpid, err := syscall.Wait4(pid,&wstat,syscall.WNOHANG,&rusage);
3532         //wpid, err := syscall.Wait4(pid,nil,syscall.WNOHANG,&rusage);
3533     }
3534     wpid := pid.Pid();
3535     err := errors.New("stab_NoError");
3536     if err != nil {
3537         fmt.Printf("--E-- %%d [%d] (%v)\n",ji,wpid,err)
3538     }else{

```

```

3540     fmt.Printf("%s\n",ji,vpid)
3541     showRusage("chld",argv,rusage)
3542     }
3543     }
3544 }
3545 func (gsh*GshContext)inBackground(argv []string)(bool){
3546     if gsh.CmdTrace { fmt.Printf("--I- inBackground(%v)\n",argv) }
3547     gsh.BackGround = true // set background option
3548     xfin := false
3549     xfin = gsh.gshelly(argv)
3550     gsh.BackGround = false
3551     return xfin
3552 }
3553 // -o file without command means just opening it and refer by #N
3554 // should be listed by "files" command
3555 func (gshCtx*GshContext)xOpen(argv []string){
3556     //var pv = [1]int{-1,-1}
3557     //err := syscall.Pipe(pv)
3558     //fmt.Printf("--I- pipe()=[#d,#d](%v)\n",pv[0],pv[1],err)
3559     pin,pout,err := os.Pipe();
3560     fmt.Printf("--I- pipe()=[#d,#d](%v)\n",pin.Fd(),pout.Fd(),err)
3561 }
3562 func (gshCtx*GshContext)fromPipe(argv []string){
3563 }
3564 func (gshCtx*GshContext)xClose(argv []string){
3565 }
3566 }
3567 // <a name="redirect">redirect</a>
3568 func (gshCtx*GshContext)redirect(argv []string)(bool){
3569     if len(argv) < 2 {
3570         return false
3571     }
3572 }
3573 cmd := argv[0]
3574 fname := argv[1]
3575 var file *os.File = nil
3576 fdix := 0
3577 mode := os.O_RDONLY
3578 }
3579 switch {
3580 case cmd == "-i" || cmd == "<":
3581     fdix = 0
3582     mode = os.O_RDONLY
3583 case cmd == "-o" || cmd == ">":
3584     fdix = 1
3585     mode = os.O_RDWR | os.O_CREATE
3586 case cmd == "-a" || cmd == ">>":
3587     fdix = 1
3588     mode = os.O_RDWR | os.O_CREATE | os.O_APPEND
3589 }
3590 if fname[0] == '#' {
3591     fd, err := strconv.Atoi(fname[1:])
3592     if err != nil {
3593         fmt.Printf("--E- (%v)\n",err)
3594         return false
3595     }
3596     file = os.NewFile(uintptr(fd),"MaybePipe")
3597 }else{
3598     xfile, err := os.OpenFile(argv[1], mode, 0600)
3599     if err != nil {
3600         fmt.Printf("--E- (%s)\n",err)
3601         return false
3602     }
3603     file = xfile
3604 }
3605 gshPA := gshCtx.gshPA
3606 savfd := gshPA.Files[fdix]
3607 //gshPA.Files[fdix] = file.Fd()
3608 gshPA.Files[fdix] = file;
3609 fmt.Printf("--I- Opened [%d] %s\n",file.Fd(),argv[1])
3610 gshCtx.gshelly(argv[2:])
3611 gshPA.Files[fdix] = savfd
3612 }
3613 }
3614 return false
3615 }
3616 //fmt.Fprintf(res, "GShell Status: %q", html.EscapeString(req.URL.Path))
3617 func httpHandler(res http.ResponseWriter, req *http.Request){
3618     path := req.URL.Path
3619     fmt.Printf("--I- Got HTTP Request(%s)\n",path)
3620     {
3621         gshCtxBuf, _ := setupGshContext()
3622         gshCtx := *gshCtxBuf
3623         fmt.Printf("--I- %s\n",path[1:])
3624         gshCtx.tgshelly(path[1:])
3625     }
3626     fmt.Fprintf(res, "Hello (-)//\n%s\n",path)
3627 }
3628 }
3629 func (gshCtx *GshContext) httpServer(argv []string){
3630     http.HandleFunc("/", httpHandler)
3631     accport := "localhost:9999"
3632     fmt.Printf("--I- HTTP Server Start at [%s]\n",accport)
3633     http.ListenAndServe(accport,nil)
3634 }
3635 func (gshCtx *GshContext)xGo(argv []string){
3636     go gshCtx.gshelly(argv[1:]);
3637 }
3638 func (gshCtx *GshContext) xPs(argv []string){}
3639 }
3640 }
3641 // <a name="plugin">Plugin</a>
3642 // plugin -ls [names] to list plugins
3643 // Reference: <a href="https://golang.org/src/plugin/">plugin</a> source code
3644 func (gshCtx *GshContext) whichPlugin(name string,argv []string)(pi *PluginInfo){
3645     pi = nil
3646     for , p := range gshCtx.PluginFuncs {
3647         if p.Name == name && pi == nil {
3648             pi = p
3649         }
3650         if !isin("-s",argv){
3651             //fmt.Printf("%v %v ",i,p)
3652             if !isin("-ls",argv){
3653                 showFileInfo(p.Path,argv)
3654             }else{
3655                 fmt.Printf("%s\n",p.Name)
3656             }
3657         }
3658     }
3659     return pi
3660 }
3661 func (gshCtx *GshContext) xPlugin(argv []string) (error) {
3662     if len(argv) == 0 || argv[0] == "-ls" {
3663         gshCtx.whichPlugin("",argv)
3664         return nil
3665     }
3666     name := argv[0]
3667     pin := gshCtx.whichPlugin(name,[]string{"-s"})
3668     if pin != nil {
3669         os.Args = argv // should be recovered?
3670         pin.Addr.(func())()
3671         return nil
3672     }
3673     sofile := toFullpath(argv[0] + ".so") // or find it by which($PATH)
3674     p, err := plugin.Open(sofile)
3675     if err != nil {
3676         fmt.Printf("--E- plugin.Open(%s)(%v)\n",sofile,err)
3677         return err
3678     }
3679     fname := "Main"
3680     f, err := p.Lookup(fname)
3681     if( err != nil ){
3682         fmt.Printf("--E- plugin.Lookup(%s)(%v)\n",fname,err)
3683         return err
3684     }
3685     pin := PluginInfo {p,f,name,sofile}
3686     gshCtx.PluginFuncs = append(gshCtx.PluginFuncs,pin)
3687     fmt.Printf("--I- added (%d)\n",len(gshCtx.PluginFuncs))
3688 }
3689 //fmt.Printf("--I- first call(%s:%s)\n",sofile,fname,argv)
3690 os.Args = argv
3691 f.(func())()
3692 return err
3693 }
3694 }
3695 func (gshCtx*GshContext)Args(argv []string){
3696     for i,v := range os.Args {
3697         fmt.Printf("[%v] %v\n",i,v)
3698     }
3699 }
3700 func (gshCtx *GshContext) showVersion(argv []string){
3701     if !isin("-l",argv) {
3702         fmt.Printf("%v %v (%v)",NAME,VERSION,DATE);
3703     }else{
3704         fmt.Printf("%v",VERSION);
3705     }
3706     if !isin("-a",argv) {
3707         fmt.Printf(" %s",AUTHOR)
3708     }
3709     if !isin("-n",argv) {
3710         fmt.Printf("\n")
3711     }
3712 }
3713 }
3714 // <a name="scanf">Scanf</a> // string decomposer
3715 // scanf [format] [input]
3716 func scanf(sstr string)(strv []string){

```

```

3717     strv = strings.Split(sstr, " ")
3718     return strv
3719 }
3720 func scanUntil(src,end string)(rstr string, leng int){
3721     idx := strings.Index(src,end)
3722     if 0 <= idx {
3723         rstr = src[0:idx]
3724         return rstr,idx+leng(end)
3725     }
3726     return src,0
3727 }
3728 // -bn -- display base-name part only // can be in some %fmt, for sed rewriting
3729 func (gsh+GshContext)printVal(fmts string, vstr string, optv[]string){
3730     //vint,err := strconv.Atoi(vstr)
3731     var ival int64 = 0
3732     n := 0
3733     err := error(nil)
3734     if strBegins(vstr, "-") {
3735         vx_ := strconv.Atoi(vstr[1:])
3736         if vx < len(gsh.iValues) {
3737             vstr = gsh.iValues[vx]
3738         }else{
3739         }
3740     }
3741     // should use Eval()
3742     if strBegins(vstr, "0x") {
3743         n,err = fmt.Sscanf(vstr[2:], "%x", &ival)
3744     }else{
3745         n,err = fmt.Sscanf(vstr, "%d", &ival)
3746     }
3747     //fmt.Printf("--D-- n=%d err=(%v) (%s)=%v\n", n,err,vstr, ival)
3748 }
3749 if n == 1 && err == nil {
3750     //fmt.Printf("--D-- formatn(%v) ival(%v)\n", fmts, ival)
3751     fmt.Printf("%"+fmts, ival)
3752 }else{
3753     if isin("-bn", optv){
3754         fmt.Printf("%"+fmts, filepath.Base(vstr))
3755     }else{
3756         fmt.Printf("%"+fmts, vstr)
3757     }
3758 }
3759 }
3760 func (gsh+GshContext)printfv(fmts,div string, argv[]string, optv[]string, list[]string){
3761     //fmt.Printf("{%d}", len(list))
3762     //curfmt := ""
3763     outlen := 0
3764     curfmt := gsh.iFormat
3765     if 0 < len(fmts) {
3766         for xi := 0; xi < len(fmts); xi++ {
3767             fch := fmts[xi]
3768             if fch == '%' {
3769                 if xi+1 < len(fmts) {
3770                     curfmt = string(fmts[xi+1])
3771                 }
3772                 gsh.iFormat = curfmt
3773                 xi += 1
3774                 if xi+1 < len(fmts) && fmts[xi+1] == '(' {
3775                     vals, leng := scanUntil(fmts[xi+2:], ")")
3776                     //fmt.Printf("--D-- show fmt(%v) val(%v) next(%v)\n", curfmt, vals, leng)
3777                     gsh.printVal(curfmt, vals, optv)
3778                     xi += 2+leng-1
3779                     outlen += 1
3780                 }
3781                 continue
3782             }
3783             if fch == ' ' {
3784                 hi, leng := scanInt(fmts[xi+1:])
3785                 if 0 < leng {
3786                     if hi < len(gsh.iValues) {
3787                         gsh.printVal(curfmt, gsh.iValues[hi], optv)
3788                         outlen += 1 // should be the real length
3789                     }else{
3790                         fmt.Printf("(out-range)")
3791                     }
3792                 }
3793                 xi += leng
3794                 continue;
3795             }
3796             fmt.Printf("%c", fch)
3797             outlen += 1
3798         }
3799     }else{
3800         //fmt.Printf("--D-- print (%s)\n")
3801         for l,v := range list {
3802             if 0 < l {
3803                 fmt.Printf(div)
3804             }
3805             gsh.printVal(curfmt, v, optv)
3806             outlen += 1
3807         }
3808     }
3809     if 0 < outlen {
3810         fmt.Printf("\n")
3811     }
3812 }
3813 }
3814 func (gsh+GshContext)Scanv(argv[]string){
3815     //fmt.Printf("--D-- Scanv(%v)\n", argv)
3816     if len(argv) == 1 {
3817         return
3818     }
3819     argv = argv[1:]
3820     fmts := ""
3821     if strBegins(argv[0], "-F") {
3822         fmts = argv[0]
3823         gsh.Delimiter = fmts
3824         argv = argv[1:]
3825     }
3826     input := strings.Join(argv, " ")
3827     if fmts == "" { // simple decomposition
3828         v := scanv(input)
3829         gsh.iValues = v
3830         //fmt.Printf("%v\n", strings.Join(v, ","))
3831     }else{
3832         v := make([]string, 8)
3833         n,err := fmt.Sscanf(input, fmts, &v[0], &v[1], &v[2], &v[3])
3834         fmt.Printf("--D-- Sscanf ->(%v) n=%d err=(%v)\n", v,n,err)
3835         gsh.iValues = v
3836     }
3837 }
3838 func (gsh+GshContext)Printv(argv[]string){
3839     if false { //@@@
3840         fmt.Printf("%v\n", strings.Join(argv[1:], " "))
3841         return
3842     }
3843     //fmt.Printf("--D-- Printv(%v)\n", argv)
3844     //fm := strings.Join(gsh.iValues, ",")
3845     div := gsh.iDelimiter
3846     fmts := ""
3847     argv = argv[1:]
3848     if 0 < len(argv) {
3849         if strBegins(argv[0], "-F") {
3850             div = argv[0][2:]
3851             argv = argv[1:]
3852         }
3853     }
3854 }
3855 }
3856 optv := []string{}
3857 for _,v := range argv {
3858     if strBegins(v, "-"){
3859         optv = append(optv, v)
3860         argv = argv[1:]
3861     }else{
3862         break;
3863     }
3864 }
3865 if 0 < len(argv) {
3866     fmts = strings.Join(argv, " ")
3867 }
3868 gsh.printfv(fmts, div, argv, optv, gsh.iValues)
3869 }
3870 func (gsh+GshContext)Basename(argv[]string){
3871     for i,v := range gsh.iValues {
3872         gsh.iValues[i] = filepath.Base(v)
3873     }
3874 }
3875 func (gsh+GshContext)Sortv(argv[]string){
3876     sv := gsh.iValues
3877     sort.Slice(sv, func(i,j int) bool {
3878         return sv[i] < sv[j]
3879     })
3880 }
3881 func (gsh+GshContext)Shiftv(argv[]string){
3882     vi := len(gsh.iValues)
3883     if 0 < vi {
3884         if isin("-r", argv) {
3885             top := gsh.iValues[0]
3886             gsh.iValues = append(gsh.iValues[1:], top)
3887         }else{
3888             gsh.iValues = gsh.iValues[1:]
3889         }
3890     }
3891 }
3892 func (gsh+GshContext)Enq(argv[]string){
3893 }

```

```

3894 func (gsh*GshContext)Deq(argv []string){
3895 }
3896 func (gsh*GshContext)Push(argv []string){
3897     gsh.iValStack = append(gsh.iValStack,argv[1:])
3898     fmt.Printf("depth=%d\n",len(gsh.iValStack))
3899 }
3900 func (gsh*GshContext)Dump(argv []string){
3901     for i,v := range gsh.iValStack {
3902         fmt.Printf("%d %v\n",i,v)
3903     }
3904 }
3905 func (gsh*GshContext)Pop(argv []string){
3906     depth := len(gsh.iValStack)
3907     if 0 < depth {
3908         v := gsh.iValStack[depth-1]
3909         if !isIn("-cat",argv){
3910             gsh.iValues = append(gsh.iValues,v...)
3911         }else{
3912             gsh.iValues = v
3913         }
3914         gsh.iValStack = gsh.iValStack[0:depth-1]
3915         fmt.Printf("depth=%d %s\n",len(gsh.iValStack),gsh.iValues)
3916     }else{
3917         fmt.Printf("depth=%d\n",depth)
3918     }
3919 }
3920
3921 // <a name="interpreter">Command Interpreter</a>
3922 func (gshCtx*GshContext)gshellv(argv []string) (fin bool) {
3923     fin = false
3924
3925     if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr,"--I-- gshellv(%d)\n",len(argv)) }
3926     if len(argv) <= 0 {
3927         return false
3928     }
3929     xargv := []string{}
3930     for ai := 0; ai < len(argv); ai++ {
3931         xargv = append(xargv,subst(gshCtx,argv[ai],false))
3932     }
3933     argv = xargv
3934     if false {
3935         for ai := 0; ai < len(argv); ai++ {
3936             fmt.Printf("%d %s [%d]T\n",
3937                 ai,argv[ai],len(argv[ai]),argv[ai])
3938         }
3939     }
3940     cmd := argv[0]
3941     if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr,"--I-- gshellv(%d)\n",len(argv),argv) }
3942     switch { // https://tour.golang.org/flowcontrol/11
3943     case cmd == "":
3944         gshCtx.xPwd([]string{}); // empty command
3945     case cmd == "-x":
3946         gshCtx.CmdTrace = ! gshCtx.CmdTrace
3947     case cmd == "-xt":
3948         gshCtx.CmdTime = ! gshCtx.CmdTime
3949     case cmd == "-ok":
3950         gshCtx.sconnect(true, argv)
3951     case cmd == "-ou":
3952         gshCtx.sconnect(false, argv)
3953     case cmd == "-it":
3954         gshCtx.saccept(true, argv)
3955     case cmd == "-iu":
3956         gshCtx.saccept(false, argv)
3957     case cmd == "-i" || cmd == "<" || cmd == ">" || cmd == "-a" || cmd == ">>" || cmd == "-s" || cmd == "><":
3958         gshCtx.redirect(argv)
3959     case cmd == "|":
3960         gshCtx.fromPipe(argv)
3961     case cmd == "args":
3962         gshCtx.Args(argv)
3963     case cmd == "bg" || cmd == "-bg":
3964         rfin := gshCtx.inBackground(argv[1:])
3965         return rfin
3966     case cmd == "-bn":
3967         gshCtx.Basename(argv)
3968     case cmd == "call":
3969         // _ = gshCtx.excommand(false,argv[1:])
3970     case cmd == "cd" || cmd == "chdir":
3971         gshCtx.xChdir(argv);
3972     case cmd == "-cksum":
3973         gshCtx.xFind(argv)
3974     case cmd == "-sum":
3975         gshCtx.xFind(argv)
3976     case cmd == "-sumtest":
3977         str := ""
3978         if 1 < len(argv) { str = argv[1] }
3979         crc := strCRC32(str,uint64(len(str)))
3980         fmt.Printf(stderr,"%v %v\n",crc,len(str))
3981     case cmd == "close":
3982         gshCtx.xClose(argv)
3983     case cmd == "gcp":
3984         gshCtx.FileCopy(argv)
3985     case cmd == "dec" || cmd == "decode":
3986         gshCtx.Dec(argv)
3987     case cmd == "#define":
3988     case cmd == "dic" || cmd == "d":
3989         xDic(argv)
3990     case cmd == "dump":
3991         gshCtx.Dump(argv)
3992     case cmd == "echo" || cmd == "e":
3993         echo(argv,true)
3994     case cmd == "enc" || cmd == "encode":
3995         gshCtx.Enc(argv)
3996     case cmd == "env":
3997         env(argv)
3998     case cmd == "eval":
3999         xEval(argv[1:],true)
4000     case cmd == "ev" || cmd == "events":
4001         dumpEvents(argv)
4002     case cmd == "exec":
4003         // = gshCtx.excommand(true,argv[1:])
4004         // should not return here
4005     case cmd == "exit" || cmd == "quit":
4006         // write Result code EXIT to 3>
4007         return true
4008     case cmd == "fdls":
4009         // dump the attributes of fds (of other process)
4010     case cmd == "-find" || cmd == "fin" || cmd == "ufind" || cmd == "uf":
4011         gshCtx.xFind(argv[1:])
4012     case cmd == "fu":
4013         gshCtx.xFind(argv[1:])
4014     case cmd == "fork":
4015         // mainly for a server
4016     case cmd == "-gen":
4017         gshCtx.gen(argv)
4018     case cmd == "-g":
4019         gshCtx.xGo(argv)
4020     case cmd == "-grep":
4021         gshCtx.xFind(argv)
4022     case cmd == "gdeq":
4023         gshCtx.Deq(argv)
4024     case cmd == "genq":
4025         gshCtx.Eng(argv)
4026     case cmd == "gpop":
4027         gshCtx.Pop(argv)
4028     case cmd == "gpush":
4029         gshCtx.Push(argv)
4030     case cmd == "history" || cmd == "hi": // hi should be alias
4031         gshCtx.xHistory(argv)
4032     case cmd == "jobs":
4033         gshCtx.xJobs(argv)
4034     case cmd == "lnsp" || cmd == "nlsp":
4035         gshCtx.SplitLine(argv)
4036     case cmd == "lg":
4037         gshCtx.xFind(argv)
4038     case cmd == "nop":
4039         // do nothing
4040     case cmd == "pipe":
4041         gshCtx.xOpen(argv)
4042     case cmd == "plug" || cmd == "plugin" || cmd == "pin":
4043         gshCtx.xPlugin(argv[1:])
4044     case cmd == "print" || cmd == "-pr":
4045         // output internal slice // also sprintf should be
4046         gshCtx.Printv(argv)
4047     case cmd == "ps":
4048         gshCtx.xPs(argv)
4049     case cmd == "pstitle":
4050         // to be gsh.title
4051     case cmd == "rexeed" || cmd == "rexd":
4052         gshCtx.RexeceServer(argv)
4053     case cmd == "rexec" || cmd == "rex":
4054         gshCtx.RexeceClient(argv)
4055     case cmd == "repeat" || cmd == "rep": // repeat cond command
4056         gshCtx.repeat(argv)
4057     case cmd == "replay":
4058         gshCtx.xReplay(argv)
4059     case cmd == "scan":
4060         // scan input (or so in fscanf) to internal slice (like Files or map)
4061         gshCtx.Scanv(argv)
4062     case cmd == "set":
4063         // set name ...
4064     case cmd == "serv":
4065         gshCtx.httpServer(argv)
4066     case cmd == "shift":
4067         gshCtx.Shiftv(argv)
4068     case cmd == "sleep":
4069         gshCtx.sleep(argv)
4070     case cmd == "-sort":

```

```

4071     gshCtx.Sortv(argv)
4072
4073     case cmd == "j" || cmd == "join":
4074         gshCtx.Rjoin(argv)
4075     case cmd == "a" || cmd == "alpa":
4076         gshCtx.Reuse(argv)
4077     case cmd == "jcd" || cmd == "jchdir":
4078         gshCtx.Rchdir(argv)
4079     case cmd == "jget":
4080         gshCtx.Rget(argv)
4081     case cmd == "jle":
4082         gshCtx.Rls(argv)
4083     case cmd == "jput":
4084         gshCtx.Rput(argv)
4085     case cmd == "jpwd":
4086         gshCtx.Rpwd(argv)
4087
4088     case cmd == "time":
4089         fin = gshCtx.xTime(argv)
4090     case cmd == "ungets":
4091         if l < len(argv) {
4092             ungets(argv[1]+"\\n")
4093         } else {
4094             }
4095     case cmd == "pwd":
4096         gshCtx.xPwd(argv);
4097     case cmd == "ver" || cmd == "-ver" || cmd == "version":
4098         gshCtx.showVersion(argv)
4099     case cmd == "where":
4100         // data file or so?
4101         case cmd == "which":
4102             which("PATH", argv);
4103         case cmd == "gj" && l < len(argv) && argv[1] == "listen":
4104             go gj_server(argv[1:]);
4105         case cmd == "gj" && l < len(argv) && argv[1] == "serve":
4106             go gj_server(argv[1:]);
4107         case cmd == "gj" && l < len(argv) && argv[1] == "join":
4108             go gj_client(argv[1:]);
4109         case cmd == "gj":
4110             jsend(argv);
4111         case cmd == "jsend":
4112             jsend(argv);
4113     default:
4114         if gshCtx.whichPlugin(cmd, []string{"-s"}) != nil {
4115             gshCtx.xPlugin(argv)
4116         } else {
4117             notfound, _ := gshCtx.excommand(false, argv)
4118             if notfound {
4119                 fmt.Printf("--E-- command not found (%v)\\n", cmd)
4120             }
4121         }
4122     }
4123     return fin
4124 }
4125
4126 func (gsh*GshContext)gshellv(gline string) (rfin bool) {
4127     argv := strings.Split(string(gline), " ")
4128     fin := gsh.gshellv(argv)
4129     return fin
4130 }
4131 func (gsh*GshContext)tgshell(gline string)(xfn bool){
4132     start := time.Now()
4133     fin := gsh.gshell(gline)
4134     end := time.Now()
4135     elps := end.Sub(start);
4136     if gsh.CmdTime {
4137         fmt.Printf("--T-- " + time.Now().Format(time.Stamp) + " (%d.%09ds)\\n",
4138             elps/1000000000, elps%1000000000)
4139     }
4140     return fin
4141 }
4142
4143 func Ttyid() (int) {
4144     fi, err := os.Stdin.Stat()
4145     if err != nil {
4146         return 0;
4147     }
4148     //fmt.Printf("Stdin: %v Dev=%d\\n",
4149     //fi.Mode()&&fi.Mode(&os.ModeDevice)
4150     if (fi.Mode() && os.ModeDevice) != 0 {
4151         stat := aStat_t{};
4152         err := aStat(0, &stat)
4153         if err != nil {
4154             //fmt.Printf("--I-- Stdin: (%v)\\n", err)
4155         } else {
4156             //fmt.Printf("--I-- Stdin: rdev=%d %d\\n",
4157             // stat.Rdev&0xFF, stat.Rdev);
4158             //fmt.Printf("--I-- Stdin: tty=%d\\n", stat.Rdev&0xFF);
4159             return int(stat.Rdev & 0xFF)
4160         }
4161     }
4162     return 0
4163 }
4164
4165 func (gshCtx *GshContext) ttyfile() string {
4166     //fmt.Printf("--I-- GSH_HOME=%s\\n", gshCtx.GshHomeDir)
4167     ttyfile := gshCtx.GshHomeDir + "/" + "gsh-tty" +
4168         fmt.Sprintf("%02d", gshCtx.TerminalId)
4169     //strconv.Itoa(gshCtx.TerminalId)
4170     //fmt.Printf("--I-- ttyfile=%s\\n", ttyfile)
4171     return ttyfile
4172 }
4173
4174 func (gshCtx *GshContext) ttyline() (*os.File) {
4175     file, err := os.OpenFile(gshCtx.ttyfile(), os.O_RDWR|os.O_CREATE|os.O_TRUNC, 0600)
4176     if err != nil {
4177         fmt.Printf("--F-- cannot open %s (%s)\\n", gshCtx.ttyfile(), err)
4178         return file;
4179     }
4180     return file
4181 }
4182
4183 func (gshCtx *GshContext)getline(hix int, skipping bool, prevline string) (string) {
4184     if skipping {
4185         reader := bufio.NewReaderSize(os.Stdin, LINESIZE)
4186         line, _, _ := reader.ReadLine()
4187         return string(line)
4188     } else {
4189         if true {
4190             return xgetline(hix, prevline, gshCtx)
4191         }
4192     }
4193     /*
4194     if (with_exgetline && gshCtx.GetLine != "") {
4195         //var xhix int64 = int64(hix); // cast
4196         newenv := os.Environ()
4197         newenv = append(newenv, "GSH_LINENO="+strconv.FormatInt(int64(hix), 10) )
4198         tty := gshCtx.ttyline()
4199         tty.WriteString(prevline)
4200         Pa := os.ProcAttr {
4201             "", // start dir
4202             newenv, //os.Environ(),
4203             []os.File{os.Stdin, os.Stdout, os.Stderr, tty},
4204             nil,
4205         }
4206         //fmt.Printf("--I-- getline=%s // %s\\n", gsh_getlinev[0], gshCtx.GetLine)
4207         proc, err := os.StartProcess(gsh_getlinev[0], []string{"getline", "getline"}, &Pa)
4208         if err != nil {
4209             fmt.Printf("--F-- getline process error (%v)\\n", err)
4210             // for ; {
4211             return "exit (getline program failed)"
4212         }
4213         //stat, err := proc.Wait()
4214         proc.Wait()
4215         buff := make([]byte, LINESIZE)
4216         count, err := tty.Read(buff)
4217         //_, err = tty.Read(buff)
4218         //fmt.Printf("--D-- getline (%d)\\n", count)
4219         if err != nil {
4220             if l (count == 0) { // && err.String() == "EOF" } {
4221                 fmt.Printf("--E-- getline error (%s)\\n", err)
4222             }
4223         } else {
4224             //fmt.Printf("--I-- getline OK \"%s\"\\n", buff)
4225         }
4226         tty.Close()
4227         gline := string(buff[0:count])
4228         return gline
4229     } else {
4230         // if isatty {
4231         //     fmt.Printf("%d\\n", hix)
4232         //     fmt.Print(PROMPT)
4233         // }
4234         reader := bufio.NewReaderSize(os.Stdin, LINESIZE)
4235         line, _, _ := reader.ReadLine()
4236         return string(line)
4237     }
4238 }
4239
4240 //== begin ===== getline
4241 /*
4242 * getline.c
4243 * 2020-0819 extracted from dog.c
4244 * getline.go
4245 * 2020-0822 ported to Go
4246 */
4247 package main // getline main

```



```

4248 import (
4249     "fmt" // <a href="https://golang.org/pkg/fmt/">fmt</a>
4250     "strings" // <a href="https://golang.org/pkg/strings/">strings</a>
4251     "os" // <a href="https://golang.org/pkg/os/">os</a>
4252     "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
4253     "bytes" // <a href="https://golang.org/pkg/bytes/">bytes</a>
4254     "os/exec" // <a href="https://golang.org/pkg/os/exec/">os/exec</a>
4255 )
4256 /*
4257 // C language compatibility functions
4258 var errno = 0
4259 var stdin *os.File = os.Stdin
4260 var stdout *os.File = os.Stdout
4261 var stderr *os.File = os.Stderr
4262 var EOF = -1
4263 var NULL = 0
4264 type FILE os.File
4265 type StrBuff []byte
4266 var NULL_FP *os.File = nil
4267 var NULLSP = 0
4268 //var LINESIZE = 1024
4269
4270 func system(cmdstr string)(int){
4271     //PA := syscall.ProcAttr {
4272     PA := os.ProcAttr {
4273         ** // the starting directory
4274         os.Environ(),
4275         //([uintptr(os.Stdin.Fd()),os.Stdout.Fd(),os.Stderr.Fd()),
4276         []*os.File{os.Stdin,os.Stdout,os.Stderr},
4277         nil,
4278     }
4279     argv := strings.Split(cmdstr, " ")
4280     //pid, err := syscall.ForkExec(argv[0], argv, &PA)
4281     proc, err := os.StartProcess(argv[0], argv, &PA);
4282     if( err != nil ){
4283         //fmt.Printf("--Es-- system(%v)\n(%v)\n", cmdstr, err);
4284         return -1;
4285     }
4286     pstat, _ := proc.Wait();
4287     pid := pstat.Pid();
4288     if( err == nil ){
4289         //fmt.Printf("--E-- pid=%v syscall(%v) err(%v)\n", pid, cmdstr, err)
4290     }
4291     //syscall.Wait4(pid, nil, 0, nil)
4292     //fmt.Printf("===E=== pid=%d exit=%v stat=%v\n", pid, pstat.Exited(), pstat.ExitCode());
4293
4294     /*
4295     argv := strings.Split(cmdstr, " ")
4296     fmt.Fprintf(os.Stderr, "--l-- system(%v)\n", argv)
4297     //cmd := exec.Command(argv[0], argv...)
4298     cmd := exec.Command(argv[0], argv[1], argv[2])
4299     cmd.Stdin = strings.NewReader("output of system")
4300     var out bytes.Buffer
4301     cmd.Stdout = &out
4302     var serr bytes.Buffer
4303     cmd.Stderr = &serr
4304     err := cmd.Run()
4305     if err != nil {
4306         //fmt.Printf("ERR:%s\n", serr.String())
4307         //fmt.Printf("ERR:%s\n", serr.String())
4308     }
4309     }else{
4310         //fmt.Printf("%s", out.String())
4311     }
4312     */
4313     return 0
4314 }
4315 func atoi(str string)(ret int){
4316     ret, err := fmt.Sscanf(str, "%d", &ret)
4317     if err == nil {
4318         return ret
4319     }else{
4320         // should set errno
4321         return 0
4322     }
4323 }
4324 func getenv(name string)(string){
4325     val, got := os.LookupEnv(name)
4326     if got {
4327         return val
4328     }else{
4329         return "?"
4330     }
4331 }
4332 func strcpy(dst StrBuff, src string){
4333     var i int
4334     srcb := []byte(src)
4335     for i = 0; i < len(src) && srcb[i] != 0; i++ {
4336         dst[i] = srcb[i]
4337     }
4338     dst[i] = 0
4339 }
4340 func xstrcpy(dst StrBuff, src StrBuff){
4341     dst = src
4342 }
4343 func strcat(dst StrBuff, src StrBuff){
4344     dst = append(dst, src...)
4345 }
4346 func strdup(str StrBuff)(string){
4347     return string(str[:strlen(str)])
4348 }
4349 func strlen(str string)(int){
4350     return len(str)
4351 }
4352 func strlen(str StrBuff)(int){
4353     var i int
4354     for i = 0; i < len(str) && str[i] != 0; i++ {
4355     }
4356     return i
4357 }
4358 func sizeof(data StrBuff)(int){
4359     return len(data)
4360 }
4361 func isatty(fd int)(ret int){
4362     return 1
4363 }
4364
4365 func fopen(file string, mode string)(fp*os.File){
4366     if mode == "r" {
4367         fp, err := os.Open(file)
4368         if( err != nil ){
4369             //fmt.Printf("--E-- fopen(%s,%s)=(%v)\n", file, mode, err)
4370             return NULL_FP;
4371         }
4372         return fp;
4373     }else{
4374         fp, err := os.OpenFile(file, os.O_RDWR|os.O_CREATE|os.O_TRUNC, 0600)
4375         if( err != nil ){
4376             return NULL_FP;
4377         }
4378         return fp;
4379     }
4380 }
4381 func fclose(fp*os.File){
4382     fp.Close()
4383 }
4384 func fflush(fp *os.File)(int){
4385     return 0
4386 }
4387 func fgetc(fp*os.File)(int){
4388     var buf [1]byte
4389     _, err := fp.Read(buf[0:1])
4390     if( err != nil ){
4391         return EOF;
4392     }else{
4393         return int(buf[0])
4394     }
4395 }
4396 func sfgets(str*string, size int, fp*os.File)(int){
4397     buf := make(StrBuff, size)
4398     var ch int
4399     var i int
4400     for i = 0; i < len(buf)-1; i++ {
4401         ch = fgetc(fp)
4402         //fprintf(stderr, "--fgets %d/%d %X\n", i, len(buf), ch)
4403         if( ch == EOF ){
4404             break;
4405         }
4406         buf[i] = byte(ch);
4407         if( ch == '\n' ){
4408             break;
4409         }
4410     }
4411     buf[i] = 0
4412     //fprintf(stderr, "--fgets %d/%d (%s)\n", i, len(buf), buf[0:i])
4413     return i
4414 }
4415 func fgets(buf StrBuff, size int, fp*os.File)(int){
4416     var ch int
4417     var i int
4418     for i = 0; i < len(buf)-1; i++ {
4419         ch = fgetc(fp)
4420         //fprintf(stderr, "--fgets %d/%d %X\n", i, len(buf), ch)
4421         if( ch == EOF ){
4422             break;
4423         }
4424         buf[i] = byte(ch);

```

```

4425     if( ch == '\n' ){
4426         break;
4427     }
4428 }
4429 buf[i] = 0
4430 //fprintf(stderr, "--fgets %d/%d (%s)\n", i, len(buf), buf[0:i])
4431 return i
4432 }
4433 func fputc(ch int , fp*os.File)(int){
4434     var buf []byte
4435     buf[0] = byte(ch)
4436     fp.Write(buf[0:1])
4437     return 0
4438 }
4439 func fputs(buf StrBuff, fp*os.File)(int){
4440     fp.Write(buf)
4441     return 0
4442 }
4443 func xputs(str string, fp*os.File)(int){
4444     return fputs([]byte(str), fp)
4445 }
4446 func sscanf(str StrBuff, fmts string, params ...interface{})(int){
4447     fmt.Sscanf(string(str[0:strlen(str)]), fmts, params...)
4448     return 0
4449 }
4450 func fprintf(fp*os.File, fmts string, params ...interface{})(int){
4451     fmt.Fprintf(fp, fmts, params...)
4452     return 0
4453 }
4454 }
4455 // <a name="IME">Command Line IME</a>
4456 //-----MyIME----- MyIME
4457 var MyIMEVER = "MyIME/0.0.2";
4458 type RomKana struct {
4459     dic string // dictionary ID
4460     pat string // input pattern
4461     out string // output pattern
4462     hit int64 // count of hit and used
4463 }
4464 var dicents = 0
4465 var romkana [1024]RomKana
4466 var Romkana []RomKana
4467 }
4468 func isinDic(str string)(int){
4469     for i, v := range Romkana {
4470         if v.pat == str {
4471             return i
4472         }
4473     }
4474     return -1
4475 }
4476 const (
4477     DIC_COM_LOAD = "im"
4478     DIC_COM_DUMP = "s"
4479     DIC_COM_LIST = "ls"
4480     DIC_COM_ENA = "en"
4481     DIC_COM_DIS = "di"
4482 )
4483 func helpDic(argv []string){
4484     out := stderr
4485     cmd := ""
4486     if 0 < len(argv) { cmd = argv[0] }
4487     fprintf(out, "-- %v Usage\n", cmd)
4488     fprintf(out, "Commands\n")
4489     fprintf(out, "... %v %v %v [dicName] [dicURL] -- Import dictionary\n", cmd, DIC_COM_LOAD)
4490     fprintf(out, "... %v %v %v [pattern] -- Search in dictionary\n", cmd, DIC_COM_DUMP)
4491     fprintf(out, "... %v %v %v [dicName] -- List dictionaries\n", cmd, DIC_COM_LIST)
4492     fprintf(out, "... %v %v %v [dicName] -- Disable dictionaries\n", cmd, DIC_COM_DIS)
4493     fprintf(out, "... %v %v %v [dicName] -- Enable dictionaries\n", cmd, DIC_COM_ENA)
4494     fprintf(out, "... Keys ... %v\n", "ESC can be used for \"\"")
4495     fprintf(out, "... \v -- Reverse the case of the last character\n",)
4496     fprintf(out, "... \i -- Replace input with translated text\n",)
4497     fprintf(out, "... \j -- On/Off translation mode\n",)
4498     fprintf(out, "... \l -- Force Lower Case\n",)
4499     fprintf(out, "... \u -- Force Upper Case (software CapsLock)\n",)
4500     fprintf(out, "... \v -- Show translation actions\n",)
4501     fprintf(out, "... \x -- Replace the last input character with it Hexa-Decimal\n",)
4502 }
4503 func xDic(argv []string){
4504     if len(argv) <= 1 {
4505         helpDic(argv)
4506         return
4507     }
4508     argv = argv[1:]
4509     var debug = false
4510     var info = false
4511     var silent = false
4512     var dump = false
4513     var builtin = false
4514     cmd := argv[0]
4515     argv = argv[1:]
4516     opt := ""
4517     arg := ""
4518 }
4519 if 0 < len(argv) {
4520     arg1 := argv[0]
4521     if arg1[0] == '-' {
4522         switch arg1 {
4523             default: fmt.Printf("--Ed-- Unknown option(%v)\n", arg1)
4524             return
4525             case "-b": builtin = true
4526             case "-d": debug = true
4527             case "-s": silent = true
4528             case "-v": info = true
4529         }
4530     }
4531     opt = arg1
4532     argv = argv[1:]
4533 }
4534 }
4535 dicName := ""
4536 dicURL := ""
4537 if 0 < len(argv) {
4538     arg = argv[0]
4539     dicName = arg
4540     argv = argv[1:]
4541 }
4542 if 0 < len(argv) {
4543     dicURL = argv[0]
4544     argv = argv[1:]
4545 }
4546 if false {
4547     fprintf(stderr, "--Dd-- com(%v) opt(%v) arg(%v)\n", cmd, opt, arg)
4548 }
4549 if cmd == DIC_COM_LOAD {
4550     //dicType := ""
4551     dicBody := ""
4552     if !builtin && dicName != "" && dicURL == "" {
4553         f, err := os.Open(dicName)
4554         if err == nil {
4555             dicURL = dicName
4556         }else{
4557             f, err = os.Open(dicName+".html")
4558             if err == nil {
4559                 dicURL = dicName+".html"
4560             }else{
4561                 f, err = os.Open("gshdic-"+dicName+".html")
4562                 if err == nil {
4563                     dicURL = "gshdic-"+dicName+".html"
4564                 }
4565             }
4566         }
4567     }
4568     if err == nil {
4569         var buf = make([]byte, 128*1024)
4570         count, err := f.Read(buf)
4571         f.Close()
4572         if info {
4573             fprintf(stderr, "--Id-- ReadDic(%v,%v)\n", count, err)
4574         }
4575         dicBody = string(buf[0:count])
4576     }
4577     if dicBody == "" {
4578         switch arg {
4579             default:
4580                 dicName = "WorldDic"
4581                 dicURL = "WorldDic"
4582                 if info {
4583                     fprintf(stderr, "--Id-- default dictionary \"%v\"\n",
4584                         dicName);
4585                 }
4586             case "wnn":
4587                 dicName = "WnnDic"
4588                 dicURL = "WnnDic"
4589             case "sumomo":
4590                 dicName = "SumomoDic"
4591                 dicURL = "SumomoDic"
4592             case "sijimi":
4593                 dicName = "SijimiDic"
4594                 dicURL = "SijimiDic"
4595             case "jkl":
4596                 dicName = "JKLJaDic"
4597                 dicURL = "JA_JKLJaDic"
4598         }
4599     }
4600     if debug {
4601         fprintf(stderr, "--Id-- %v URL=%v\n\n", dicName, dicURL);
4602     }
4603 }

```

```

4602     dicv := strings.Split(dicURL, ",")
4603     if debug {
4604         fmt.Fprintf(stderr, "--Id-- %v encoded data...\n", dicName)
4605         fmt.Fprintf(stderr, "Type: %v\n", dicv[0])
4606         fmt.Fprintf(stderr, "Body: %v\n", dicv[1])
4607         fmt.Fprintf(stderr, "\n")
4608     }
4609     body, _ := base64.StdEncoding.DecodeString(dicv[1])
4610     dicBody = string(body)
4611 }
4612 if info {
4613     fmt.Printf("--Id-- %v %v\n", dicName, dicURL)
4614     fmt.Printf("%s\n", dicBody)
4615 }
4616 if debug {
4617     fmt.Fprintf(stderr, "--Id-- dicName %v text...\n", dicName)
4618     fmt.Fprintf(stderr, "%v\n", string(dicBody))
4619 }
4620 envv := strings.Split(dicBody, "\n")
4621 if info {
4622     fmt.Fprintf(stderr, "--Id-- %v scan...\n", dicName);
4623 }
4624 var added int = 0
4625 var dup int = 0
4626 for i, v := range envv {
4627     var pat string
4628     var out string
4629     fmt.Sscanf(v, "%s %s", &pat, &out)
4630     if len(pat) <= 0 {
4631     } else {
4632         if 0 <= isinDic(pat) {
4633             dup += 1
4634             continue
4635         }
4636         romkana[dicents] = RomKana(dicName, pat, out, 0)
4637         dicents += 1
4638         added += 1
4639         Romkan = append(Romkan, RomKana(dicName, pat, out, 0))
4640         if debug {
4641             fmt.Printf("%3v: [%2v] %8v [%2v] %v\n",
4642                 i, len(pat), pat, len(out), out)
4643         }
4644     }
4645 }
4646 if isilent {
4647     url := dicURL
4648     if strBegins(url, "data:") {
4649         url = "builtin"
4650     }
4651     fmt.Fprintf(stderr, "--Id-- %v scan... %v added, %v dup. / %v total (%v)\n",
4652         dicName, added, dup, len(Romkan), url);
4653 }
4654 // should sort by pattern length for complete match, for performance
4655 if debug {
4656     arg = "" // search pattern
4657     dump = true
4658 }
4659 }
4660 if cmd == DIC_COM_DUMP || dump {
4661     fmt.Fprintf(stderr, "--Id-- %v dump... %v entries:\n", dicName, len(Romkan));
4662     var match = 0
4663     for i := 0; i < len(Romkan); i++ {
4664         dic := Romkan[i].dic
4665         pat := Romkan[i].pat
4666         out := Romkan[i].out
4667         if arg == "" || 0 <= strings.Index(pat, arg) || 0 <= strings.Index(out, arg) {
4668             fmt.Printf("%3v: [%2v] %8v [%2v] %v [%2v] %v\n",
4669                 i, dic, len(pat), pat, len(out), out)
4670             match += 1
4671         }
4672     }
4673     fmt.Fprintf(stderr, "--Id-- %v matched %v / %v entries:\n", arg, match, len(Romkan));
4674 }
4675 }
4676 func loadDefaultDic(dic int) {
4677     if (0 < len(Romkan)) {
4678         return
4679     }
4680     //fmt.Fprintf(stderr, "\n")
4681     xdic([]string{"dic", DIC_COM_LOAD});
4682 }
4683 var info = false
4684 if info {
4685     fmt.Fprintf(stderr, "--Id-- Conguraturations!! WorldDic is now activated.\n")
4686     fmt.Fprintf(stderr, "--Id-- enter \"dic\" command for help.\n")
4687 }
4688 }
4689 func readDic()(int) {
4690     /*
4691     var rk *os.File;
4692     var dic = "MYIME-dic.txt";
4693     //rk = fopen("romkana.txt", "r");
4694     //rk = fopen("JK-JB-morse-dic.txt", "r");
4695     rk = fopen(dic, "r");
4696     if( rk == NULL_FP ) {
4697         if true {
4698             fmt.Fprintf(stderr, "--s-- Could not load %s\n", MYIMEVER, dic);
4699         }
4700         return -1;
4701     }
4702     if( true ) {
4703         var di int;
4704         var line = make(StrBuff, 1024);
4705         var pat string
4706         var out string
4707         for di = 0; di < 1024; di++ {
4708             if( fgets(line, sizeof(line), rk) == NULLSP ) {
4709                 break;
4710             }
4711             fmt.Sscanf(string(line[0:strlen(line)]), "%s %s", &pat, &out);
4712             //sscanf(line, "%s %[\r\n]", &pat, &out);
4713             romkana[di].pat = pat;
4714             romkana[di].out = out;
4715             //fmt.Fprintf(stderr, "--bd- %8s\n", pat, out)
4716         }
4717         dicents += di
4718         if false {
4719             fmt.Fprintf(stderr, "--s-- loaded romkana.txt [%d]\n", MYIMEVER, di);
4720             for di = 0; di < dicents; di++ {
4721                 fmt.Fprintf(stderr,
4722                     "%s %v\n", romkana[di].pat, romkana[di].out);
4723             }
4724         }
4725     }
4726     fclose(rk);
4727 }
4728 //romkana[dicents].pat = "//ddump"
4729 //romkana[dicents].pat = "//ddump" // dump the dic. and clean the command input
4730 /*
4731 return 0;
4732 }
4733 func matchlen(stri string, pati string)(int) {
4734     if strBegins(stri, pati) {
4735         return len(pati)
4736     } else {
4737         return 0
4738     }
4739 }
4740 func convs(src string)(string) {
4741     var si int;
4742     var sx = len(src);
4743     var di int;
4744     var mi int;
4745     var dstb []byte
4746 }
4747 for si = 0; si < sx; { // search max. match from the position
4748     if strBegins(src[si:], "x") {
4749         // \x/integer/ // s/a/b/
4750         ix := strings.Index(src[si+3:], "/")
4751         if 0 < ix {
4752             var iv int = 0
4753             //fmt.Sscanf(src[si+3:si+3+ix], "%d", &iv)
4754             sval := fmt.Sprintf("%d", iv)
4755             bval := []byte(sval)
4756             dstb = append(dstb, bval...)
4757             si = si+3+ix+1
4758             continue
4759         }
4760     }
4761     if strBegins(src[si:], "d") {
4762         // \d/integer/ // s/a/b/
4763         ix := strings.Index(src[si+3:], "/")
4764         if 0 < ix {
4765             var iv int = 0
4766             //fmt.Sscanf(src[si+3:si+3+ix], "%d", &iv)
4767             sval := fmt.Sprintf("%d", iv)
4768             bval := []byte(sval)
4769             dstb = append(dstb, bval...)
4770             si = si+3+ix+1
4771             continue
4772         }
4773     }
4774     if strBegins(src[si:], "t") {
4775         now := time.Now()
4776         if true {
4777             date := now.Format(time.Stamp)
4778         }

```

```

4779     dstb = append(dstb, []byte(date)...)
4780     si = si+3
4781     }
4782     continue
4783     }
4784     var maxlen int = 0;
4785     var len int;
4786     mi = -1;
4787     for di = 0; di < dicents; di++ {
4788         len = matchlen(src[si:], romkana[di].pat);
4789         if (maxlen < len) {
4790             maxlen = len;
4791             mi = di;
4792         }
4793     }
4794     if (0 < maxlen) {
4795         out := romkana[mi].out;
4796         dstb = append(dstb, []byte(out)...);
4797         si += maxlen;
4798     } else {
4799         dstb = append(dstb, src[si])
4800         si += 1;
4801     }
4802     }
4803     return string(dstb)
4804 }
4805 func trans(src string)(int){
4806     dst := conv(src);
4807     xfprintf(dst, stderr);
4808     return 0;
4809 }
4810 }
4811 //----- LINEEDIT
4812 // "?" at the top of the line means searching history
4813 }
4814 // should be compatible with Telnet
4815 const (
4816     EV_MODE     = 255
4817     EV_IDLE    = 254
4818     EV_TIMEOUT  = 253
4819
4820     GO_UP      = 252 // k
4821     GO_DOWN   = 251 // j
4822     GO_RIGHT  = 250 // l
4823     GO_LEFT   = 249 // h
4824     DEL_RIGHT = 248 // x
4825     GO_TOPL   = 'A'-0x40 // 0
4826     GO_ENDL   = 'E'-0x40 // $
4827
4828     GO_TOPW   = 239 // b
4829     GO_ENDW   = 238 // e
4830     GO_NEXTW  = 237 // w
4831
4832     GO_FORWCH = 229 // f
4833     GO_PAIRCH = 228 // %
4834
4835     GO_DEL    = 219 // d
4836
4837     HI_SRCH_FW = 209 // /
4838     HI_SRCH_BK = 208 // ?
4839     HI_SRCH_RFW = 207 // n
4840     HI_SRCH_RBK = 206 // N
4841 )
4842 }
4843 // should return number of octets ready to be read immediately
4844 //fprintf(stderr, "\n--Select(%v %v)\n", err, r.Bits[0])
4845 }
4846 }
4847 var EventRecvFd = -1 // file descriptor
4848 var EventSendFd = -1
4849 const EventFdOffset = 1000000
4850 const NormalFdOffset = 100
4851 }
4852 /* 2020-1021 replaced poll() with channel/select
4853 func putKeyinEvent(event int, evarg int){
4854     if true {
4855         if EventRecvFd < 0 {
4856             var pv = [int32]-1
4857             syscall.Pipe(pv)
4858             EventRecvFd = pv[0]
4859             EventSendFd = pv[1]
4860             //fmt.Printf("--De-- EventPipe created(%v,%v)\n", EventRecvFd, EventSendFd)
4861         }
4862     } else {
4863         if EventRecvFd < 0 {
4864             // the document differs from this spec
4865             // https://golang.org/src/syscall_unix.go?s=8096:8158#L340
4866             sv, err := syscall.Socketpair(syscall.AF_UNIX, syscall.SOCK_STREAM, 0)
4867             EventRecvFd = sv[0]
4868             EventSendFd = sv[1]
4869             if err != nil {
4870                 //fmt.Printf("--De-- EventSock created(%v,%v)\n",
4871                     EventRecvFd, EventSendFd, err)
4872             }
4873         }
4874     }
4875     var buf = []byte{ byte(event) }
4876     n, err := syscall.Write(EventSendFd, buf)
4877     if err != nil {
4878         //fmt.Printf("--De-- putEvent(%v) (%v) (%v %v)\n", EventSendFd, event, n, err)
4879     }
4880 }
4881 */
4882 func ungets(str string){
4883     for _, ch := range str {
4884         putKeyinEvent(int(ch), 0)
4885     }
4886 }
4887 func (gsh *GshContext)xReplay(argv []string){
4888     hix := 0
4889     tempo := 1.0
4890     xtempo := 1.0
4891     repeat := 1
4892 }
4893 for _, a := range argv { // tempo
4894     if strBegins(a, "x") {
4895         fmt.Sscanf(a[1:], "%f", &xtempo)
4896         tempo = 1 / xtempo
4897         //fprintf(stderr, "--De-- tempo={%v}\n", a[2:], tempo);
4898     } else {
4899         if strBegins(a, "c") { // repeat
4900             fmt.Sscanf(a[1:], "%v", &repeat)
4901         } else {
4902             if strBegins(a, "l") {
4903                 fmt.Sscanf(a[1:], "%d", &hix)
4904             } else {
4905                 fmt.Sscanf(a, "%d", &hix)
4906             }
4907         }
4908     }
4909     if hix == 0 || len(argv) <= 1 {
4910         hix = len(gsh.CommandHistory)-1
4911     }
4912     //fmt.Printf("--Ir-- Replay(!%v x%v r%v)\n", hix, xtempo, repeat)
4913     //dumpEvents(hix)
4914     //gsh.xScanReplay(hix, false, repeat, tempo, argv)
4915     go gsh.xScanReplay(hix, true, repeat, tempo, argv)
4916     runtime.Gosched(); // wait xScanReplay is launched
4917     //fmt.Printf("--Ir-- Replay set\n");
4918 }
4919 }
4920 // <a href="https://golang.org/pkg/syscall/#FdSet">syscall.Select</a>
4921 // 2020-0827 GShell-0.2.3
4922 /*
4923 func FpollIn(fp *os.File, usec int)(uintptr){
4924     nfd := 1
4925
4926     rdv := syscall.FdSet {}
4927     fd1 := fp.Fd()
4928     bank1 = fd1/32
4929     mask1 := int32(1 << fd1)
4930     rdv.Bits[bank1] = mask1
4931
4932     fd2 := -1
4933     bank2 := -1
4934     var mask2 int32 = 0
4935
4936     if 0 <= EventRecvFd {
4937         fd2 = EventRecvFd
4938         nfd = fd2 + 1
4939         bank2 = fd2/32
4940         mask2 = int32(1 << fd2)
4941         rdv.Bits[bank2] |= mask2
4942         //fmt.Printf("--De-- EventPoll mask added [%d][%v]\n", fd2, bank2, mask2)
4943     }
4944     tout := syscall.NsecToTimeval(int64(usec*1000))
4945     //n, err := syscall.Select(nfd, rdv, nil, nil, &tout) // spec. mismatch
4946     err := syscall.Select(nfd, rdv, nil, nil, &tout)
4947     if err != nil {
4948         //fmt.Printf("--De-- select() err(%v)\n", err)
4949     }
4950     if err == nil {
4951         if 0 <= fd2 && (rdv.Bits[bank2] & mask2) != 0 {
4952             if false {
4953                 //fmt.Printf("--De-- got Event\n")
4954             }
4955         }

```

```

4956         return uintptr(EventFdOffset + fd2)
4957     }else
4958     if (rdv.Bits[bank1] & mask1) != 0 {
4959         return uintptr(NormalFdOffset + fd1)
4960     }else{
4961         return 1
4962     }
4963 }else{
4964     return 0
4965 }
4966 }
4967 */
4968 /*
4969 func fgetcTimeout1(fp *os.File,usec int)(int){
4970     READ:
4971     //readyFd := FpollIn1(fp,usec)
4972     readyFd := CpollIn1(fp,usec)
4973     if readyFd < 100 {
4974         return EV_TIMEOUT
4975     }
4976     var buf [1]byte
4977     if EventFdOffset <= readyFd {
4978         fd := int(readyFd-EventFdOffset)
4979         _,err := syscall.Read(fd,buf[0:1])
4980         if (err != nil){
4981             return EOF;
4982         }else{
4983             if buf[0] == EV_MODE {
4984                 recvKeyEvent(fd)
4985                 goto READ
4986             }
4987             return int(buf[0])
4988         }
4989     }
4990     _,err := fp.Read(buf[0:1])
4991     if (err != nil){
4992         return EOF;
4993     }else{
4994         return int(buf[0])
4995     }
4996 }
4997 */
5000 */
5001 func visibleChar(ch int)(string){
5002     switch {
5003     case '!' <= ch && ch <= '-':
5004         return string(ch)
5005     }
5006     switch ch {
5007     case '\a': return "\\a"
5008     case '\n': return "\\n"
5009     case '\r': return "\\r"
5010     case '\t': return "\\t"
5011     }
5012     switch ch {
5013     case 0x00: return "NUL"
5014     case 0x07: return "BEL"
5015     case 0x08: return "BS"
5016     case 0x0E: return "SO"
5017     case 0x0F: return "SI"
5018     case 0x1B: return "ESC"
5019     case 0x7F: return "DEL"
5020     }
5021     switch ch {
5022     case EV_IDLE: return fmt.Sprintf("IDLE")
5023     case EV_MODE: return fmt.Sprintf("MODE")
5024     }
5025     return fmt.Sprintf("%X",ch)
5026 }
5027 */
5028 func recvKeyEvent(fd int){
5029     var buf = make([]byte,1)
5030     _,_ = syscall.Read(fd,buf[0:1])
5031     if( buf[0] != 0 ){
5032         romkanmode = true
5033     }else{
5034         romkanmode = false
5035     }
5036 }
5037 */
5038 func (gsh*GshContext)xScanReplay(hix int,replay bool,repeat int,tempo float64,argv[]string){
5039     var Start time.Time
5040     var events = []Event{}
5041     for _,e := range Events {
5042         if hix == 0 || e.CmdIndex == hix {
5043             events = append(events,e)
5044         }
5045     }
5046     elen := len(events)
5047     if 0 < elen {
5048         if events[elen-1].event == EV_IDLE {
5049             events = events[0:elen-1]
5050         }
5051     }
5052     for r := 0; r < repeat; r++ {
5053         for i,e := range events {
5054             nano := e.when.Nanosecond()
5055             micro := nano / 1000
5056             if Start.Second() == 0 {
5057                 Start = time.Now()
5058             }
5059             diff := time.Now().Sub(Start)
5060             if replay {
5061                 if e.event != EV_IDLE {
5062                     //fmt.Printf("--replay %v / %v event=%X\n",i,len(events),e.event);
5063                     putKeyEvent(e.event,0)
5064                     if e.event == EV_MODE { // event with arg
5065                         putKeyEvent(int(e.evarg),0)
5066                     }
5067                 }else{
5068                     //fmt.Printf("--replay %v / %v idle=%X\n",i,len(events),e.event);
5069                 }
5070             }else{
5071                 fmt.Printf("#7.3fms #8-3v %8-3v [%v.%06d] %3v %02X %4v %10.3fms\n",
5072                     i,
5073                     e.CmdIndex,
5074                     e.when.Format(time.Stamp),micro,
5075                     e.event,e.event,visibleChar(e.event),
5076                     float64(e.evarg)/1000000.0)
5077             }
5078             if e.event == EV_IDLE {
5079                 //fmt.Printf("--replay %v / %v delay\n",i,len(events));
5080                 d := time.Duration(float64(time.Duration(e.evarg)) * tempo)
5081                 //nsleep(time.Duration(e.evarg))
5082                 nsleep(d)
5083             }
5084         }
5085     }
5086 }
5087 */
5088 func dumpEvents(argv[]string){
5089     hix := 0
5090     if 1 < len(argv) {
5091         hix = atoi(argv[1])
5092     }
5093     for i,e := range Events {
5094         nano := e.when.Nanosecond()
5095         micro := nano / 1000
5096         //if e.event != EV_TIMEOUT {
5097             if hix == 0 || e.CmdIndex == hix {
5098                 fmt.Printf("#8-3v %8-3v [%v.%06d] %3v %02X %4v %10.3fms\n",i,
5099                     e.CmdIndex,
5100                     e.when.Format(time.Stamp),micro,
5101                     e.event,e.event,visibleChar(e.event),float64(e.evarg)/1000000.0)
5102             }
5103         }
5104     }
5105 }
5106 */
5107 func fgetcTimeout(fp *os.File,usec int)(int){
5108     ch := fgetcTimeout1(fp,usec)
5109     if ch != EV_TIMEOUT {
5110         now := time.Now()
5111         if 0 < len(Events) {
5112             last := Events[len(Events)-1]
5113             dura := int64(now.Sub(last.when))
5114             Events = append(Events,Event{last.when,EV_IDLE,dura,last.CmdIndex})
5115         }
5116         Events = append(Events,Event{time.Now(),ch,0,CmdIndex})
5117     }
5118     return ch
5119 }
5120 */
5121 */
5122 // 2020-1021 replaced poll() with channel/select
5123 var Kbd = make(chan int);
5124 var Kbinet = false;
5125 var evQ = make(chan int);
5126 */
5127 func keyInput(kbd chan int, fp *os.File){
5128     for {
5129         ch := C.getc(C.stdin);
5130         if( ch == C.EOF ){
5131             break;
5132         }
5133     }
5134 }

```

```

5133     }
5134     kbd <- int(ch);
5135     }
5136     }
5137     */
5138     // https://godoc.org/golang.org/x/crypto/ssh/terminal
5139     // https://stackoverflow.com/questions/14094190/function-similar-to-getchar
5140     func keyInput(kbd chan int, tty *os.File) {
5141         tmode := C.setTermRaw();
5142         defer func() { C.setTermMode(tmode); }();
5143         if(!OnWindows) {
5144             system("/bin/stty -echo -icanon");
5145             defer func() { system("/bin/stty echo sane"); }();
5146         }
5147         for {
5148             var rbuf []byte = make([]byte,1);
5149             if(OnWindows) {
5150                 C.setTermRaw();
5151             }
5152             _,rerr := tty.Read(rbuf);
5153             if(rerr != nil) {
5154                 break;
5155             }
5156             //fmt.Printf("++KBD[%X]\n",rbuf[0]);
5157             kbd <- int(rbuf[0]);
5158         }
5159     }
5160     if(!OnWindows) { system("/bin/stty echo sane"); }
5161     func fgetcTimeout(fp *os.File,usec int)(int){
5162         if(!Kbinit) {
5163             Kbinit = true;
5164             go keyInput(Kbd,fp);
5165         }
5166         for {
5167             select {
5168                 case <- time.After(time.Duration(usec*1000)):
5169                     //fmt.Printf("--Timeout %v us\n",usec);
5170                     return EV_TIMEOUT;
5171                 case ch := <- Kbd:
5172                     //fmt.Printf("--KBD[%X]\n",ch);
5173                     // record a KeyIn(ch) Event
5174                     {
5175                         now := time.Now()
5176                         if 0 < len(Events) {
5177                             last := Events[len(Events)-1]
5178                             dura := int64(now.Sub(last.when))
5179                             Events = append(Events,Event{last.when,EV_IDLE,dura,last.CmdIndex})
5180                         }
5181                         Events = append(Events,Event{time.Now(),ch,0,CmdIndex})
5182                     }
5183                     return ch;
5184                 case ch := <- evQ:
5185                     if(ch == EV_MODE) {
5186                         recvKeyEvent()
5187                     }else{
5188                         return ch;
5189                     }
5190             }
5191         }
5192     }
5193     func putKeyInEvent(event int, evarg int){
5194         evQ <- event;
5195     }
5196     func recvKeyEvent(){
5197         ch := <- evQ;
5198         if(ch != 0) {
5199             romkanmode = true
5200         }else{
5201             romkanmode = false
5202         }
5203     }
5204
5205     var AtConsoleLineTop = true
5206     var TtyMaxCol = 72 // to be obtained by ioctl?
5207     var EscTimeout = (100*1000)
5208     var (
5209         MODE_VicMode    bool // vi compatible command mode
5210         MODE_ShowMode   bool // shown translation mode, the mode to be retained
5211         romkanmode      bool // shown translation mode, the mode to be retained
5212         MODE_Recursive  bool // recursive translation
5213         MODE_CapsLock   bool // software CapsLock
5214         MODE_Lowerlock  bool // force lower-case character lock
5215         MODE_VInsert    int // visible insert mode, should be like "I" icon in X Window
5216         MODE_VItrace    bool // output newline before translation
5217     )
5218     type IInput struct {
5219         lno int
5220         lastlno int
5221         pch []int // input queue
5222         prompt string
5223         line string
5224         right string
5225         inmode bool
5226         pinmode bool
5227         waitingMeta string // waiting meta character
5228         LastCmd string
5229     }
5230     func (iin*IInput)Getc(timeoutUs int)(int){
5231         ch1 := EOF
5232         ch2 := EOF
5233         ch3 := EOF
5234         if( 0 < len(iin.pch) ) { // deQ
5235             ch1 = iin.pch[0]
5236             iin.pch = iin.pch[1:]
5237         }else{
5238             ch1 = fgetcTimeout(stdin,timeoutUs);
5239         }
5240         if(ch1 == 033) { // escape sequence
5241             ch2 = fgetcTimeout(stdin,EscTimeout);
5242             if(ch2 == EV_TIMEOUT) {
5243                 }else{
5244                     ch3 = fgetcTimeout(stdin,EscTimeout);
5245                     if(ch3 == EV_TIMEOUT) {
5246                         iin.pch = append(iin.pch,ch2) // enQ
5247                     }else{
5248                         switch(ch2) {
5249                             default:
5250                                 iin.pch = append(iin.pch,ch2) // enQ
5251                                 iin.pch = append(iin.pch,ch3) // enQ
5252                         }
5253                         case '[':
5254                             switch(ch3) {
5255                                 case 'A': ch1 = GO_UP; // ^
5256                                 case 'B': ch1 = GO_DOWN; // v
5257                                 case 'C': ch1 = GO_RIGHT; // >
5258                                 case 'D': ch1 = GO_LEFT; // <
5259                                 case '3':
5260                                     ch4 := fgetcTimeout(stdin,EscTimeout);
5261                                     if(ch4 == '-') {
5262                                         //fprintf(stderr,"X%02X %02X %02X %02X\n",ch1,ch2,ch3,ch4);
5263                                         ch1 = DEL_RIGHT
5264                                     }
5265                                 case '\\':
5266                                     //ch4 := fgetcTimeout(stdin,EscTimeout);
5267                                     //fprintf(stderr,"Y%02X %02X %02X %02X\n",ch1,ch2,ch3,ch4);
5268                                     switch(ch3) {
5269                                         case '-': ch1 = DEL_RIGHT
5270                                     }
5271                                 }
5272                             }
5273                         }
5274                     }
5275                 }
5276                 return ch1
5277             }
5278             func (iin*IInput)clearline(){
5279                 var i int
5280                 fprintf(stderr,"r");
5281                 // should be ANSI ESC sequence
5282                 for i = 0; i < TtyMaxCol; i++ { // to the max. position in this input action
5283                     fputc(' ',os.Stderr);
5284                 }
5285                 fprintf(stderr,"r");
5286             }
5287             func (iin*IInput)Redraw(){
5288                 redraw(iin,iin.lno,iin.line,iin.right)
5289             }
5290             func redraw(iin *IInput,lno int,line string,right string){
5291                 inMeta := false
5292                 showMode := "" // visible Meta mode on the cursor position
5293                 showLno := fmt.Sprintf("%d",lno)
5294                 InsertMark := "" // in visible insert mode
5295                 if MODE_VicMode {
5296                     }else
5297                 if 0 < len(iin.right) {
5298                     InsertMark = " "
5299                 }
5300             }
5301             if( 0 < len(iin.waitingMeta) ) {
5302                 inMeta = true
5303                 if iin.waitingMeta[0] != 033 {
5304                     showMeta = iin.waitingMeta
5305                 }
5306             }
5307             if( romkanmode ) {
5308                 //romkanmark = " *";

```

```

5310 }else{
5311 //romkanmark = "";
5312 }
5313 if MODE_ShowMode {
5314 romkan := "--"
5315 inmeta := "-"
5316 inveri := ""
5317 if MODE_CapsLock {
5318 inmeta = "A"
5319 }
5320 if MODE_LowerLock {
5321 inmeta = "a"
5322 }
5323 if MODE_ViTrace {
5324 inveri = "v"
5325 }
5326 if MODE_VicMode {
5327 inveri = ":"
5328 }
5329 if romkanmode {
5330 romkan = "343201202"
5331 if MODE_CapsLock {
5332 inmeta = "R"
5333 }else{
5334 inmeta = "r"
5335 }
5336 }
5337 if inMeta {
5338 inmeta = ""
5339 }
5340 showMode = "["+romkan+inmeta+inveri+"]";
5341 }
5342 Pre := "\r" + showMode + showLineo
5343 Output := ""
5344 Left := ""
5345 Right := ""
5346 if romkanmode {
5347 Left = convs(line)
5348 Right = InsertMark+convs(right)
5349 }else{
5350 Left = line
5351 Right = InsertMark+right
5352 }
5353 Output = Pre+Left
5354 if MODE_ViTrace {
5355 Output += iin.LastCmd
5356 }
5357 Output += showMeta+Right
5358 for len(Output) < TtyMaxCol { // to the max. position that may be dirty
5359 Output += " "
5360 // should be ANSI ESC sequence
5361 // not necessary just after newline
5362 }
5363 Output += Pre+Left+showMeta // to set the cursor to the current input position
5364 fprintf(stderr,"%s",Output)
5365 }
5366 if MODE_ViTrace {
5367 if 0 < len(iin.LastCmd) {
5368 iin.LastCmd = ""
5369 fprintf(stderr,"\r\n")
5370 }
5371 }
5372 AtConsoleLineTop = false
5373 //fmt.Printf("(Redraw(%v))\n",len(line),len(right));
5374 }
5375 // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
5376 func delHeadChar(str string)(rline string,head string){
5377 clen := utf8.DecodeRune([]byte(str))
5378 head = string(str[0:clen])
5379 return str[clen:],head
5380 }
5381 func delTailChar(str string)(rline string, last string){
5382 var i = 0
5383 var clen = 0
5384 for {
5385 clen := utf8.DecodeRune([]byte(str)[i:])
5386 if clen == 0 { break }
5387 clen = siz
5388 i += siz
5389 }
5390 last = str[len(str)-clen:]
5391 return str[0:len(str)-clen:],last
5392 }
5393 }
5394 // 3> for output and history
5395 // 4> for keylog?
5396 // <a name="getline">Command Line Editor</a>
5397 func xgetline(lno int, prevline string, gsh*GshContext)(string){
5398 var iin Input
5399 iin.lastlno = lno
5400 iin.lno = lno
5401 }
5402 CmdIndex = len(gsh.CommandHistory)
5403 if( isatty(0) == 0 ){
5404 if( sfgets(&iin.line,LINESIZE,stdin) == NULL ){
5405 iin.line = "exit\n";
5406 }else{
5407 return iin.line
5408 }
5409 }
5410 if( true ){
5411 //var pts string;
5412 //pts = ptsname(0);
5413 //pts = ttyname(0);
5414 //fprintf(stderr,"--pts[0] = %s\n",pts?pts:"");
5415 }
5416 if( false ){
5417 fprintf(stderr, "! ");
5418 fflush(stderr);
5419 sfgets(&iin.line,LINESIZE,stdin);
5420 return iin.line
5421 }
5422 if( !OnWindows ){system("/bin/stty -echo -icanon"); }
5423 xline = iin.xgetline(prevline,gsh)
5424 if( !OnWindows ){system("/bin/stty echo sane"); }
5425 return xline
5426 }
5427 func (iin*Input)Translate(cmdch int){
5428 romkanmode = iromkanmode;
5429 if MODE_ViTrace {
5430 fprintf(stderr,"%v\r\n",string(cmdch));
5431 }else
5432 if( cmdch == 'J' ){
5433 fprintf(stderr,"J\r\n");
5434 iin.inMode = true
5435 }
5436 iin.Redraw();
5437 loadDefaultDic(cmdch);
5438 iin.Redraw();
5439 }
5440 func (iin*Input)Replace(cmdch int){
5441 iin.LastCmd = fmt.Sprintf("%v",string(cmdch))
5442 iin.Redraw();
5443 loadDefaultDic(cmdch);
5444 dst := convs(iin.line+iin.right);
5445 iin.line = dst
5446 iin.right = ""
5447 if( cmdch == 'I' ){
5448 fprintf(stderr,"I\r\n");
5449 iin.inMode = true
5450 }
5451 iin.Redraw();
5452 }
5453 // aa 12 alal
5454 func isAlpha(ch rune)(bool){
5455 if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
5456 return true
5457 }
5458 return false
5459 }
5460 func isAlnum(ch rune)(bool){
5461 if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
5462 return true
5463 }
5464 if '0' <= ch && ch <= '9' {
5465 return true
5466 }
5467 return false
5468 }
5469 }
5470 // 0.2.8 2020-0901 created
5471 // <a href="https://golang.org/pkg/unicode/utf8/#DecodeRuneInString">DecodeRuneInString</a>
5472 func (iin*Input)GotoTOPW(){
5473 str = iin.line
5474 i := len(str)
5475 if i <= 0 {
5476 return
5477 }
5478 //io := i
5479 i -= 1
5480 lastSize := 0
5481 var lastRune rune
5482 var found = -1
5483 for 0 < i { // skip preamble spaces
5484 lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
5485 if !isAlnum(lastRune) { // character, type, or string to be searched
5486 i -= lastSize

```

```

5487         continue
5488     }
5489     break
5490 }
5491 for 0 < i {
5492     lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
5493     if lastSize <= 0 { continue } // not the character top
5494     if !isAlnum(lastRune) { // character, type, or string to be searched
5495         found = i
5496         break
5497     }
5498     i += lastSize
5499 }
5500 if found < 0 && i == 0 {
5501     found = 0
5502 }
5503 if 0 <= found {
5504     if isAlnum(lastRune) { // or non-kana character
5505     }else{ // when positioning to the top o the word
5506         i += lastSize
5507     }
5508     iin.right = str[i:] + iin.right
5509     if 0 < i {
5510         iin.line = str[0:i]
5511     }else{
5512         iin.line = ""
5513     }
5514 }
5515 //fmt.Printf("\n(%d,%d,%d)[%s][%s]\n",i0,i,found,iin.line,iin.right)
5516 //fmt.Printf("") // set debug messae at the end of line
5517 }
5518 // 0.2.8 2020-0901 created
5519 func (iin*Input)GotoENDW(){
5520     str := iin.right
5521     if len(str) <= 0 {
5522         return
5523     }
5524     lastSize := 0
5525     var lastRune rune
5526     var lastW = 0
5527     i := 0
5528     inWord := false
5529
5530     lastRune, lastSize = utf8.DecodeRuneInString(str[0:])
5531     if isAlnum(lastRune) {
5532     }, z := utf8.DecodeRuneInString(str[lastSize:])
5533     if 0 < z && isAlnum(r) {
5534         inWord = true
5535     }
5536 }
5537 for i < len(str) {
5538     lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
5539     if lastSize <= 0 { break } // broken data?
5540     if !isAlnum(lastRune) { // character, type, or string to be searched
5541         break
5542     }
5543     lastW = i // the last alnum if in alnum word
5544     i += lastSize
5545 }
5546 if inWord {
5547     goto DISP
5548 }
5549 for i < len(str) {
5550     lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
5551     if lastSize <= 0 { break } // broken data?
5552     if !isAlnum(lastRune) { // character, type, or string to be searched
5553         break
5554     }
5555     i += lastSize
5556 }
5557 for i < len(str) {
5558     lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
5559     if lastSize <= 0 { break } // broken data?
5560     if !isAlnum(lastRune) { // character, type, or string to be searched
5561         break
5562     }
5563     lastW = i
5564     i += lastSize
5565 }
5566 DISP:
5567 if 0 < lastW {
5568     iin.line = iin.line + str[0:lastW]
5569     iin.right = str[lastW:]
5570 }
5571 //fmt.Printf("\n(%d)[%s][%s]\n",i,iin.line,iin.right)
5572 //fmt.Printf("") // set debug messae at the end of line
5573 }
5574 // 0.2.8 2020-0901 created
5575 func (iin*Input)GotoNEXTW(){
5576     str := iin.right
5577     if len(str) <= 0 {
5578         return
5579     }
5580     lastSize := 0
5581     var lastRune rune
5582     var found = -1
5583     i := 1
5584     for i < len(str) {
5585     lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
5586     if lastSize <= 0 { break } // broken data?
5587     if !isAlnum(lastRune) { // character, type, or string to be searched
5588         found = i
5589         break
5590     }
5591     i += lastSize
5592 }
5593 if 0 < found {
5594     if isAlnum(lastRune) { // or non-kana character
5595     }else{ // when positioning to the top o the word
5596         found += lastSize
5597     }
5598     iin.line = iin.line + str[0:found]
5599     if 0 < found {
5600         iin.right = str[found:]
5601     }else{
5602         iin.right = ""
5603     }
5604 }
5605 //fmt.Printf("\n(%d)[%s][%s]\n",i,iin.line,iin.right)
5606 //fmt.Printf("") // set debug messae at the end of line
5607 }
5608 // 0.2.8 2020-0902 created
5609 func (iin*Input)GotoPAIRCH(){
5610     str := iin.right
5611     if len(str) <= 0 {
5612         return
5613     }
5614     lastRune, lastSize := utf8.DecodeRuneInString(str[0:])
5615     if lastSize <= 0 {
5616         return
5617     }
5618     forw := false
5619     back := false
5620     pair := ""
5621     switch string(lastRune){
5622     case "(": pair = ")"; forw = true
5623     case ")": pair = "("; back = true
5624     case "[": pair = "]"; forw = true
5625     case "]": pair = "["; back = true
5626     case "{": pair = "}"; forw = true
5627     case "}": pair = "{"; back = true
5628     case "<": pair = ">"; forw = true
5629     case ">": pair = "<"; back = true
5630     case "\"": pair = "\""; // context depednet, can be f" or back-double quote
5631     case "`": pair = "`"; // context depednet, can be f' or back-quote
5632     // case Japanese Kakkos
5633 }
5634 if forw {
5635     iin.SearchForward(pair)
5636 }
5637 if back {
5638     iin.SearchBackward(pair)
5639 }
5640 }
5641 // 0.2.8 2020-0902 created
5642 func (iin*Input)SearchForward(pat string)(bool){
5643     right := iin.right
5644     found := -1
5645     i := 0
5646     if strBegins(right,pat) {
5647     }, z := utf8.DecodeRuneInString(right[i:])
5648     if 0 < z {
5649         i += z
5650     }
5651 }
5652 for i < len(right) {
5653     if strBegins(right[i:],pat) {
5654         found = i
5655         break
5656     }
5657     }, z := utf8.DecodeRuneInString(right[i:])
5658     if z <= 0 { break }
5659     i += z
5660 }
5661 if 0 <= found {
5662     iin.line = iin.line + right[0:found]
5663     iin.right = iin.right[found:]

```



```

5664         return true
5665     }else{
5666         return false
5667     }
5668 }
5669 // 0.2.8 2020-0902 created
5670 func (iin*IInput)SearchBackward(pat string)(bool){
5671     line := iin.line
5672     found = -1
5673     i := len(line)-1
5674     for i = i; 0 <= i; i-- {
5675         z := utf8.DecodeRuneInString(line[i:])
5676         if z <= 0 {
5677             continue
5678         }
5679         //fprintf(stderr, "-- %v %v\n", pat, line[i:])
5680         if strBegins(line[i:], pat) {
5681             found = i
5682             break
5683         }
5684     }
5685     //fprintf(stderr, "--%d\n", found)
5686     if 0 <= found {
5687         iin.right = line[found:] + iin.right
5688         iin.line = line[0:found]
5689         return true
5690     }else{
5691         return false
5692     }
5693 }
5694 // 0.2.8 2020-0902 created
5695 // search from top, end, or current position
5696 func (gsh*GshContext)SearchHistory(pat string, forw bool)(bool, string){
5697     if forw {
5698         for v := range gsh.CommandHistory {
5699             if 0 <= strings.Index(v.CmdLine, pat) {
5700                 //fprintf(stderr, "\n--De-- found !%v [%v]\n", i, pat, v.CmdLine)
5701                 return true, v.CmdLine
5702             }
5703         }
5704     }else{
5705         hlen := len(gsh.CommandHistory)
5706         for i := hlen-1; 0 < i; i-- {
5707             v := gsh.CommandHistory[i]
5708             if 0 <= strings.Index(v.CmdLine, pat) {
5709                 //fprintf(stderr, "\n--De-- found !%v [%v]\n", i, pat, v.CmdLine)
5710                 return true, v.CmdLine
5711             }
5712         }
5713     }
5714     //fprintf(stderr, "\n--De-- not-found(%v)\n", pat)
5715     return false, "(Not Found in History)"
5716 }
5717 // 0.2.8 2020-0902 created
5718 func (iin*IInput)GotoFORWSTR(pat string, gsh*GshContext){
5719     found = false
5720     if 0 < len(iin.right) {
5721         found = iin.SearchForward(pat)
5722     }
5723     if !found {
5724         found, line := gsh.SearchHistory(pat, true)
5725         if found {
5726             iin.line = line
5727             iin.right = ""
5728         }
5729     }
5730 }
5731 func (iin*IInput)GotoBACKSTR(pat string, gsh*GshContext){
5732     found = false
5733     if 0 < len(iin.line) {
5734         found = iin.SearchBackward(pat)
5735     }
5736     if !found {
5737         found, line := gsh.SearchHistory(pat, false)
5738         if found {
5739             iin.line = line
5740             iin.right = ""
5741         }
5742     }
5743 }
5744 func (iin*IInput)getString(prompt string)(string){ // should be editable
5745     iin.clearline();
5746     fprintf(stderr, "\r?v", prompt)
5747     str := ""
5748     for {
5749         ch := iin.Getc(10*1000*1000)
5750         if ch == '\n' || ch == '\r' {
5751             break
5752         }
5753         sch := string(ch)
5754         str += sch
5755         fprintf(stderr, "%s", sch)
5756     }
5757     return str
5758 }
5759
5760 // search pattern must be an array and selectable with 'N/P'
5761 var SearchPat = ""
5762 var SearchForw = true
5763
5764 func (iin*IInput)xgetline(prevline string, gsh*GshContext)(string){
5765     var ch int;
5766     MODE_ShowMode = false
5767     MODE_VicMode = false
5768     iin.Redraw();
5769     first = true
5770
5771     for cix := 0; ; cix++ {
5772         iin.pinJmode = iin.inJmode
5773         iin.inJmode = false
5774
5775         ch = iin.Getc(1000*1000)
5776
5777         if ch != EV_TIMEOUT && first {
5778             first = false
5779             mode = 0
5780             if romkanmode {
5781                 mode = 1
5782             }
5783             now := time.Now()
5784             Events = append(Events, Event(now, EV_MODE, int64(mode), CmdIndex))
5785         }
5786         if ch == 033 {
5787             MODE_ShowMode = true
5788             MODE_VicMode = !MODE_VicMode
5789             iin.Redraw();
5790             continue
5791         }
5792         if MODE_VicMode {
5793             switch ch {
5794                 case 'o': ch = GO_TOPL
5795                 case '$': ch = GO_ENDL
5796                 case 'b': ch = GO_TOPW
5797                 case 'e': ch = GO_ENDW
5798                 case 'w': ch = GO_NEXTW
5799                 case '%': ch = GO_PAIRCH
5800
5801                 case 'j': ch = GO_DOWN
5802                 case 'k': ch = GO_UP
5803                 case 'h': ch = GO_LEFT
5804                 case 'l': ch = GO_RIGHT
5805                 case 'x': ch = DEL_RIGHT
5806                 case 'a': MODE_VicMode = !MODE_VicMode
5807                 case 'r': ch = GO_RIGHT
5808                 case 'i': MODE_VicMode = !MODE_VicMode
5809                 iin.Redraw();
5810                 continue
5811                 case '-':
5812                     right, head := delHeadChar(iin.right)
5813                     if len([]byte(head)) == 1 {
5814                         ch = int(head[0])
5815                         if 'a' <= ch && ch <= 'z' {
5816                             ch = ch + 'A' - 'a'
5817                         }else{
5818                             if 'A' <= ch && ch <= 'Z' {
5819                                 ch = ch + 'a' - 'A'
5820                             }
5821                         }
5822                     }
5823                     iin.right = string(ch) + right
5824                 }
5825             iin.Redraw();
5826             continue
5827             case 'f': // GO_FORWCH
5828                 iin.Redraw();
5829                 ch = iin.Getc(3*1000*1000)
5830                 if ch == EV_TIMEOUT {
5831                     iin.Redraw();
5832                     continue
5833                 }
5834                 SearchPat = string(ch)
5835                 SearchForw = true
5836                 iin.GotoFORWSTR(SearchPat, gsh)
5837                 iin.Redraw();
5838                 continue
5839             case '/':
5840                 SearchPat = iin.getString("/") // should be editable
5841                 SearchForw = true

```

```

5841         iin.GotoFORWSTR(SearchPat,gsh)
5842         iin.Redraw();
5843         continue
5844     case '?':
5845         SearchPat = iin.getStringl("?") // should be editable
5846         SearchForw = false
5847         iin.GotoBACKSTR(SearchPat,gsh)
5848         iin.Redraw();
5849         continue
5850     case 'n':
5851         if SearchForw {
5852             iin.GotoFORWSTR(SearchPat,gsh)
5853         }else{
5854             iin.GotoBACKSTR(SearchPat,gsh)
5855         }
5856         iin.Redraw();
5857         continue
5858     case 'N':
5859         if !SearchForw {
5860             iin.GotoFORWSTR(SearchPat,gsh)
5861         }else{
5862             iin.GotoBACKSTR(SearchPat,gsh)
5863         }
5864         iin.Redraw();
5865         continue
5866     }
5867 }
5868 switch ch {
5869 case GO_TOPW:
5870     iin.GotoTOPW()
5871     iin.Redraw();
5872     continue
5873 case GO_ENDW:
5874     iin.GotoENDW()
5875     iin.Redraw();
5876     continue
5877 case GO_NEXTW:
5878     // To next space then
5879     iin.GotoNEXTW()
5880     iin.Redraw();
5881     continue
5882 case GO_PAIRCH:
5883     iin.GotoPAIRCH()
5884     iin.Redraw();
5885     continue
5886 }
5887 //fprintf(stderr,"A[%02X]\n",ch);
5888 if( ch == '\\ ' || ch == 033 ){
5889     MODE_ShowMode = true
5890     metach := ch
5891     iin.waitingMeta = string(ch)
5892     iin.Redraw();
5893     // set cursor //fprintf(stderr,"???\b\b\b")
5894     ch = fgetcTimeout(stdin,2000*1000)
5895     // reset cursor
5896     iin.waitingMeta = ""
5897     cmdch := ch
5898     if( ch == EV_TIMEOUT ){
5899         if metach == 033 {
5900             continue
5901         }
5902         ch = metach
5903     }else
5904     /*
5905     if( ch == 'm' || ch == 'M' ){
5906         mch := fgetcTimeout(stdin,1000*1000)
5907         if mch == 'r' {
5908             romkanmode = true
5909         }else{
5910             romkanmode = false
5911         }
5912         continue
5913     }else
5914     /*
5915     if( ch == 'k' || ch == 'K' ){
5916         MODE_Recursive = !MODE_Recursive
5917         iin.Translate(cmdch);
5918         continue
5919     }else
5920     if( ch == 'j' || ch == 'J' ){
5921         iin.Translate(cmdch);
5922         continue
5923     }else
5924     if( ch == 'i' || ch == 'I' ){
5925         iin.Replace(cmdch);
5926         continue
5927     }else
5928     if( ch == 'l' || ch == 'L' ){
5929         MODE_LowerLock = !MODE_LowerLock
5930         MODE_CapsLock = !MODE_CapsLock
5931         if MODE_ViTrace {
5932             fprintf(stderr,"%v\n",string(cmdch));
5933         }
5934         iin.Redraw();
5935         continue
5936     }else
5937     if( ch == 'u' || ch == 'U' ){
5938         MODE_CapsLock = !MODE_CapsLock
5939         MODE_LowerLock = false
5940         if MODE_ViTrace {
5941             fprintf(stderr,"%v\n",string(cmdch));
5942         }
5943         iin.Redraw();
5944         continue
5945     }else
5946     if( ch == 'v' || ch == 'V' ){
5947         MODE_ViTrace = !MODE_ViTrace
5948         if MODE_ViTrace {
5949             fprintf(stderr,"%v\n",string(cmdch));
5950         }
5951         iin.Redraw();
5952         continue
5953     }else
5954     if( ch == 'c' || ch == 'C' ){
5955         if 0 < len(iin.line) {
5956             xline,tail := delTailChar(iin.line)
5957             if len([]byte(tail)) == 1 {
5958                 ch = int(tail[0])
5959                 if 'a' <= ch && ch <= 'z' {
5960                     ch = ch + 'A'-'a'
5961                 }else
5962                 if 'A' <= ch && ch <= 'Z' {
5963                     ch = ch + 'a'-'A'
5964                 }
5965             }
5966             iin.line = xline + string(ch)
5967         }
5968         if MODE_ViTrace {
5969             fprintf(stderr,"%v\n",string(cmdch));
5970         }
5971         iin.Redraw();
5972         continue
5973     }else{
5974         iin.pch = append(iin.pch,ch) // push
5975         ch = '\\'
5976     }
5977 }
5978 }
5979 switch( ch ){
5980 case 'P'-0x40: ch = GO_UP
5981 case 'N'-0x40: ch = GO_DOWN
5982 case 'B'-0x40: ch = GO_LEFT
5983 case 'F'-0x40: ch = GO_RIGHT
5984 }
5985 //fprintf(stderr,"B[%02X]\n",ch);
5986 switch( ch ){
5987 case 0:
5988     continue;
5989 case '\t':
5990     iin.Replace('j');
5991     continue
5992 case 'X'-0x40:
5993     iin.Replace('j');
5994     continue
5995 case EV_TIMEOUT:
5996     iin.Redraw();
5997     if iin.pinmode {
5998         fprintf(stderr,"\\J\n")
5999         iin.inmode = true
6000     }
6001     continue
6002 case GO_UP:
6003     if iin.lno == 1 {
6004         continue
6005     }
6006     cmd,ok := gsh.cmdStringInHistory(iin.lno-1)
6007     if ok {
6008         iin.line = cmd
6009         iin.right = ""
6010         iin.lno = iin.lno - 1
6011     }
6012     iin.Redraw();
6013     continue
6014 case GO_DOWN:

```

```

6018 cmd,ok := gsh.cmdStringInHistory(iin.lno+1)
6019 if ok {
6020     iin.line = cmd
6021     iin.right = ""
6022     iin.lno = iin.lno + 1
6023 }else{
6024     iin.line = ""
6025     iin.right = ""
6026     if iin.lno == iin.lastlno-1 {
6027         iin.lno = iin.lno + 1
6028     }
6029 }
6030 iin.Redraw();
6031 continue
6032 case GO LEFT:
6033     if 0 < len(iin.line) {
6034         xline,tail := delTailChar(iin.line)
6035         iin.line = xline
6036         iin.right = tail + iin.right
6037     }
6038     iin.Redraw();
6039     continue;
6040 case GO RIGHT:
6041     if( 0 < len(iin.right) && iin.right[0] != 0 ){
6042         xright,head := delHeadChar(iin.right)
6043         iin.right = xright
6044         iin.line += head
6045     }
6046     iin.Redraw();
6047     continue;
6048 case EOF:
6049     goto EXIT;
6050 case 'R'-0x40: // replace
6051     dst := convs(iin.line+iin.right);
6052     iin.line = dst
6053     iin.right = ""
6054     iin.Redraw();
6055     continue;
6056 case 'T'-0x40: // just show the result
6057     readDic();
6058     romkanmode = !romkanmode;
6059     iin.Redraw();
6060     continue;
6061 case 'L'-0x40:
6062     iin.Redraw();
6063     continue
6064 case 'K'-0x40:
6065     iin.right = ""
6066     iin.Redraw();
6067     continue
6068 case 'E'-0x40:
6069     iin.line += iin.right
6070     iin.right = ""
6071     iin.Redraw();
6072     continue
6073 case 'A'-0x40:
6074     iin.right = iin.line + iin.right
6075     iin.line = ""
6076     iin.Redraw();
6077     continue
6078 case 'U'-0x40:
6079     iin.line = ""
6080     iin.right = ""
6081     iin.clearline();
6082     iin.Redraw();
6083     continue;
6084 case DEL RIGHT:
6085     if( 0 < len(iin.right) ){
6086         iin.right,_ = delHeadChar(iin.right)
6087         iin.Redraw();
6088     }
6089     continue;
6090 case 0x7F: // BS? not DEL
6091     if( 0 < len(iin.line) ){
6092         iin.line,_ = delTailChar(iin.line)
6093         iin.Redraw();
6094     }
6095     /*
6096     else
6097     if( 0 < len(iin.right) ){
6098         iin.right,_ = delHeadChar(iin.right)
6099         iin.Redraw();
6100     }
6101     */
6102     continue;
6103 case 'H'-0x40:
6104     if( 0 < len(iin.line) ){
6105         iin.line,_ = delTailChar(iin.line)
6106         iin.Redraw();
6107     }
6108     continue;
6109 }
6110 if( OnWindows && ch == '\n' ){
6111     continue;
6112 }
6113 if( ch == '\n' || ch == '\r' ){
6114     iin.line += iin.right;
6115     iin.right = ""
6116     iin.Redraw();
6117     /*putc(ch,stderr);
6118     fprintf(stderr, "\r\n"); // NL on Unix, CR on Windows
6119     AtConsoleLineTop = true
6120     break;
6121 */
6122 if MODE_CapsLock {
6123     if 'a' <= ch && ch <= 'z' {
6124         ch = ch+'A'-'a'
6125     }
6126 }
6127 if MODE_LowerLock {
6128     if 'A' <= ch && ch <= 'Z' {
6129         ch = ch+'a'-'A'
6130     }
6131 }
6132 iin.line += string(ch);
6133 iin.Redraw();
6134 }
6135 EXIT:
6136 return iin.line + iin.right;
6137 }
6138
6139 func getline_main(){
6140     line := xgetline(0,"",nil)
6141     fprintf(stderr, "%s\n",line);
6142     /*
6143     dp = strpbk(line, "\r\n");
6144     if( dp != NULL ){
6145         *dp = 0;
6146     }
6147     if( 0 ){
6148         fprintf(stderr, "\n(%d)\n",int(strlen(line)));
6149     }
6150     if( lseek(3,0,0) == 0 ){
6151         if( romkanmode ){
6152             var buf [8*1024]byte;
6153             convs(line,buf);
6154             strcpy(line,buf);
6155         }
6156         write(3,line,strlen(line));
6157         truncate(3,lseek(3,0,SEEK_CUR));
6158         //fprintf(stderr, "outsize=%d\n", (int)lseek(3,0,SEEK_END));
6159         lseek(3,0,SEEK_SET);
6160         close(3);
6161     }else{
6162         fprintf(stderr, "\r\n gotline: ");
6163         trans(line);
6164         //printf("%s\n",line);
6165         printf("\n");
6166     }
6167 */
6168 }
6169 }
6170 }
6171 }
6172 //
6173 // $USERHOME/.gsh/
6174 // gsh-rc.txt, or gsh-configure.txt
6175 // gsh-history.txt
6176 // gsh-aliases.txt // should be conditional?
6177 //
6178 func (gshCtx *GshContext)gshSetupHomedir()(bool) {
6179     homedir,found := userHomeDir()
6180     if !found {
6181         fmt.Printf("--E-- You have no UserHomeDir\n")
6182         return true
6183     }
6184     gshhome := homedir + "/" + GSH_HOME
6185     err2 := os.Stat(gshhome)
6186     if err2 != nil {
6187         err3 := os.Mkdir(gshhome,0700)
6188         if err3 != nil {
6189             fmt.Printf("--E-- Could not Create %s (%s)\n",
6190                 gshhome,err3)
6191             return true
6192         }
6193     }
6194     fmt.Printf("--I-- Created %s\n",gshhome)
6195 }

```

```

6195     gshCtx.GshHomeDir = gshhome
6196     return false
6197 }
6198 func setupGshContext()(GshContext,bool){
6199 //gshPA := syscall.ProcAttr {
6200 gshPA := os.ProcAttr {
6201     // the starting directory
6202     os.Environ(), // environ[]
6203     //[]uintptr(os.Stdin.Fd()),os.Stdout.Fd(),os.Stderr.Fd()),
6204     //[]os.File(os.Stdin,os.Stdout,os.Stderr),
6205     nil, // OS specific
6206 }
6207 cwd, _ := os.Getwd()
6208 gshCtx := GshContext {
6209     cwd, // StartDir
6210     // GetLine
6211     []GshChdirHistory { {cwd,time.Now(),0} }, // ChdirHistory
6212     gshPA,
6213     []GCommandHistory{}, //something for invokation?
6214     GCommandHistory{}, // CmdCurrent
6215     false,
6216     []os.ProcessState{}, //[]int{,
6217     aRusage{},
6218     // GshHomeDir
6219     Ttyid{
6220         false,
6221         false,
6222     }PluginInfo{,
6223     []string{,
6224         "g",
6225     }ValueStack{,
6226     GServer{"","}, // LastServer
6227     // RSERV
6228     cwd, // RMD
6229     CheckSum{,
6230     }
6231 }
6232 err := gshCtx.gshSetupHomedir()
6233 return gshCtx, err
6234 }
6235 func (gsh*GshContext)gshellllh(gline string)(bool){
6236 ghist := gsh.CmdCurrent
6237 ghist.WorKDir = os.Getwd()
6238 ghist.WorKDirX = len(gsh.ChdirHistory)-1
6239 //fmt.Printf("--D-ChdirHistory(%#d)\n",len(gsh.ChdirHistory))
6240 ghist.StartAt = time.Now()
6241 rusagev1 := Getrusagev()
6242 gsh.CmdCurrent.FoundFile = []string{
6243     fin := gsh.tgshellll(gline)
6244     rusagev2 := Getrusagev()
6245     ghist.Rusagev = RusageSubv(rusagev2,rusagev1)
6246     ghist.EndAt = time.Now()
6247     ghist.Cmdline = gline
6248     ghist.FoundFile = gsh.CmdCurrent.FoundFile
6249 }
6250 /* record it but not show in list by default
6251 if len(gline) == 0 {
6252     continue
6253 }
6254 if gline == "hi" || gline == "history" { // don't record it
6255     continue
6256 }
6257 */
6258 gsh.CommandHistory = append(gsh.CommandHistory, ghist)
6259 return fin
6260 }
6261 // <a name="main">Main loop</a>
6262 func script(gshCtxGiven *GshContext) (_ GshContext) {
6263     gshCtxBuf,err0 := setupGshContext()
6264     if err0 != nil {
6265         return gshCtxBuf;
6266     }
6267     gshCtx := *gshCtxBuf
6268     //fmt.Printf("--I-- GSH_HOME=%s\n",gshCtx.GshHomeDir)
6269     //resmap()
6270 }
6271 /*
6272 if false {
6273     gsh_getlinev, with exgetline :=
6274     which("PATH",[]string{"which","gsh-getline","-s"})
6275     if with exgetline {
6276         gsh_getlinev[0] = toFullpath(gsh_getlinev[0])
6277         gshCtx.GetLine = toFullpath(gsh_getlinev[0])
6278     }else{
6279         fmt.Printf("--W-- No gsh-getline found. Using internal getline.\n");
6280     }
6281 }
6282 */
6283 }
6284 ghist0 := gshCtx.CmdCurrent // something special, or gshrc script, or permanent history
6285 gshCtx.CommandHistory = append(gshCtx.CommandHistory,ghist0)
6286 }
6287 }
6288 prevline := ""
6289 skipping := false
6290 for hix := len(gshCtx.CommandHistory); ; {
6291     gline := gshCtx.getline(hix,skipping,prevline)
6292     if skipping {
6293         if strings.Index(gline,"fi") == 0 {
6294             fmt.Printf("fi\n");
6295             skipping = false;
6296         }else{
6297             //fmt.Printf("%s\n",gline);
6298         }
6299         continue
6300     }
6301     if strings.Index(gline,"if") == 0 {
6302         //fmt.Printf("--D-- if start: %s\n",gline);
6303         skipping = true;
6304         continue
6305     }
6306     if false {
6307         os.Stdout.Write([]byte("gotline:"))
6308         os.Stdout.Write([]byte(gline))
6309         os.Stdout.Write([]byte("\n"))
6310     }
6311     gline = strsubst(gshCtx,gline,true)
6312     if false {
6313         fmt.Printf("fmt.Printf %v - %v\n",gline)
6314         fmt.Printf("fmt.Printf %s - %s\n",gline)
6315         fmt.Printf("fmt.Printf %x - %x\n",gline)
6316         fmt.Printf("fmt.Printf %U - %U\n",gline)
6317         fmt.Printf("Stoutt.Write -")
6318         os.Stdout.Write([]byte(gline))
6319         fmt.Printf("\n")
6320     }
6321     /*
6322     // should be cared in substitution ?
6323     if 0 < len(gline) && gline[0] == '!' {
6324         xgline, set, err := searchHistory(gshCtx,gline)
6325         if err != nil {
6326             continue
6327         }
6328         if set {
6329             // set the line in command line editor
6330         }
6331         gline = xgline
6332     }
6333     */
6334     fin := gshCtx.gshellllh(gline)
6335     if fin {
6336         break;
6337     }
6338     prevline = gline;
6339     hix++;
6340 }
6341 return *gshCtx
6342 }
6343 }
6344 func ftest(wher, path string){
6345 //fi,err := os.Stat(path);
6346 //fmt.Printf("-- %v os.Stat(%v)=(%v)%v\n",wher,path,err,fi);
6347 }
6348 }
6349 func main() {
6350     initGshEnv();
6351     ftest("gsh-main",".");
6352     ftest("gsh-main","gsh.go");
6353     ftest("gsh-main","gsh.exe");
6354     ftest("gsh-main","gsh");
6355 }
6356 }
6357 gshCtxBuf := GshContext{
6358     gsh := *gshCtxBuf
6359     argv := os.Args
6360 }
6361 if( isin("wss",argv) ){
6362     gj_server(argv[1]);
6363     return;
6364 }
6365 if( isin("wsc",argv) ){
6366     gj_client(argv[1]);
6367     return;
6368 }
6369 if 1 < len(argv) {
6370     if isin("version",argv){
6371         gsh.showVersion(argv)
6372         return
6373     }
6374     if argv[1] == "gj" {

```

```

6372 if argv[2] == "listen" { go gj_server(argv[2]); }
6373 if argv[2] == "server" { go gj_server(argv[2]); }
6374 if argv[2] == "serve" { go gj_server(argv[2]); }
6375 if argv[2] == "client" { go gj_client(argv[2]); }
6376 if argv[2] == "join" { go gj_client(argv[2]); }
6377 }
6378 comx := isinX("-c",argv)
6379 if 0 < comx {
6380 gshCtxBuf,err := setupGshContext()
6381 gsh := gshCtxBuf
6382 if lerr {
6383 gsh.gshshell(argv[comx+1:])
6384 }
6385 return
6386 }
6387 if 1 < len(argv) && isin("-s",argv) {
6388 }else{
6389 gsh.showVersion(append(argv,[]string{"-l","-a"}...))
6390 }
6391 script(nil)
6392 //gshCtx := script(nil)
6393 //gshshell(gshCtx,"time")
6394 }
6395 }
6396 }
6397 }
6398 }
6399 }
6400 }
6401 }
6402 }
6403 }
6404 }
6405 }
6406 }
6407 }
6408 }
6409 }
6410 }
6411 }
6412 }
6413 }
6414 }
6415 }
6416 }
6417 }
6418 }
6419 }
6420 }
6421 }
6422 }
6423 }
6424 }
6425 }
6426 }
6427 }
6428 }
6429 }
6430 }
6431 }
6432 }
6433 }
6434 }
6435 }
6436 }
6437 }
6438 }
6439 }
6440 }
6441 }
6442 }
6443 }
6444 }
6445 }
6446 }
6447 }
6448 }
6449 }
6450 }
6451 }
6452 }
6453 }
6454 }
6455 }
6456 }
6457 }
6458 }
6459 }
6460 }
6461 }
6462 }
6463 }
6464 }
6465 }
6466 }
6467 }
6468 }
6469 }
6470 }
6471 }
6472 }
6473 }
6474 }
6475 }
6476 }
6477 }
6478 }
6479 }
6480 }
6481 }
6482 }
6483 }
6484 }
6485 }
6486 }
6487 }
6488 }
6489 }
6489 }
6490 }
6491 }
6492 }
6493 }
6494 }
6495 }
6496 }
6497 }
6498 }
6499 }
6500 }
6501 }
6502 }
6503 }
6504 }
6505 }
6506 }
6507 }
6508 }
6509 }
6510 }
6511 }
6512 }
6513 }
6514 }
6515 }
6516 }
6517 }
6518 }
6519 }
6520 }
6521 }
6522 }
6523 }
6524 }
6525 }
6526 }
6527 }
6528 }
6529 }
6530 }
6531 }
6532 }
6533 }
6534 }
6535 }
6536 }
6537 }
6538 }
6539 }
6540 }
6541 }
6542 }
6543 }
6544 }
6545 }
6546 }
6547 }
6548 }

```

```

6549 <span id="gsh-script-view"></span>
6550 </details>
6551
6552 <details id="gsh-data-frame" onclick="fill_DataView()"><summary>Builtin data part</summary>
6553 <a name="gsh-data-frame"></a>
6554 <span id="gsh-data-view"></span>
6555 </details>
6556
6557 </div></details>
6558 */
6559
6560 /*
6561 <div id="GshFooter0"></div>
6562 <!-- 2020-09-17 SatoxITS, visible script { -->
6563 <details><summary>GJScript</summary>
6564 <style>gjscript { font-family:Georgia; }</style>
6565 <pre id="gjscript_1" class="gjscript"> function gjtest1(){ alert('Hello GJScript!'); }
6566 gjtest1()
6567 </pre>
6568 <script>
6569 gjs = document.getElementById('gjscript_1');
6570 //eval(gjs.innerHTML);
6571 //gjs.outerHTML = ""
6572 </script>
6573 </details><!-- ----- END-OF-VISIBLE-PART ----- -->
6574
6575 <!--
6576 // 2020-0906 added,
6577 https://developer.mozilla.org/en-US/docs/Web/CSS/z-index
6578 https://developer.mozilla.org/en-US/docs/Web/CSS/position
6579 -->
6580 <span id="GshGrid">(^_^)</small><small><small></span>
6581
6582 <span id="GStat"><br>
6583 </span>
6584 <span id="GMenu" onclick="GShellMenu(this)" draggable="true"></span>
6585 <span id="GTop"></span>
6586 <div id="GShellPlane" onclick="showGShellPlane();" ></div>
6587 <div id="RawTextViewer"></div>
6588 <div id="RawTextViewerClose" onclick="hideRawTextViewer()"> CLOSE </div>
6589
6590 <style id="GshStyleDef">
6591 #LineNumbered table,tr,td {
6592   margin:0;
6593   padding:4px;
6594   spacing:0;
6595   border:1px;
6596 }
6597 textarea.LineNumbered {
6598   font-size:12px;
6599   font-family:monospace,Courier New;
6600   color:#282;
6601   padding:4px;
6602   text-align:right;
6603 }
6604 textarea.LineNumbered {
6605   font-size:12px;
6606   font-family:monospace,Courier New;
6607   padding:4px;
6608   wrap:off;
6609 }
6610 #RawTextViewer{
6611   z-index:0;
6612   position:fixed; top:0px; left:0px;
6613   width:100%; xxxheight:50px; xheight:0px;
6614   overflow:auto;
6615   color:#fff; background-color:rgba(128,128,256,0.2);
6616   font-size:12px;
6617   spellcheck:false;
6618 }
6619 #RawTextViewerClose{
6620   z-index:0;
6621   position:fixed; top:-100px; left:-100px;
6622   color:#fff; background-color:rgba(128,128,256,0.2);
6623   font-size:20px; font-family:Georgia;
6624   white-space:pre;
6625 }
6626 #xxxGShellPlane{
6627   z-index:0;
6628   position:fixed; top:0px; left:0px;
6629   width:100%; height:50px;
6630   overflow:auto;
6631   color:#fff; background-color:rgba(128,128,256,0.3);
6632   font-size:12px;
6633 }
6634 #xxxGTop{
6635   z-index:9;
6636   opacity:1.0;
6637   position:fixed; top:0px; left:0px;
6638   width:320px; height:20px;
6639   color:#fff; background-color:rgba(32,32,160,0.15);
6640   color:#fff; font-size:12px;
6641 }
6642 #xxxGPos{
6643   z-index:12;
6644   position:fixed; top:0px; left:0px;
6645   opacity:1.0;
6646   width:640px; height:30px;
6647   color:#fff; background-color:rgba(0,0,0,0.2);
6648   color:#fff; font-size:12px;
6649 }
6650 #GMenu{
6651   z-index:100000000;
6652   position:fixed; top:250px; left:0px;
6653   opacity:1.0;
6654   width:100px; height:100px;
6655   color:#fff;
6656   color:#fff; background-color:rgba(0,0,0,0.0);
6657   color:#fff; font-size:16px; font-family:Georgia;
6658   background-repeat:no-repeat;
6659 }
6660 #xxxGStat{
6661   z-index:8;
6662   opacity:0.0;
6663   position:fixed; top:20px; left:0px;
6664   xwidth:640px;
6665   width:100%; height:90px;
6666   color:#fff; background-color:rgba(0,0,128,0.04);
6667   font-size:20px; font-family:Georgia;
6668 }
6669 #GLog{
6670   z-index:10;
6671   position:fixed; top:50px; left:0px;
6672   opacity:1.0;
6673   width:640px; height:60px;
6674   color:#fff; background-color:rgba(0,0,128,0.10);
6675   font-size:12px;
6676 }
6677 #GshGrid {
6678   z-index:11;
6679   xopacity:0.0;
6680   position:fixed; top:0px; left:0px;
6681   width:320px; height:30px;
6682   color:#9f9; font-size:16px;
6683 }
6684 xbody {display:none;}
6685 .gsh-link{color:green;}
6686 #gsh {border-width:1;margin:0;padding:0;}
6687 #gsh {font-family:monospace,Courier New;color:#ddf;font-size:8px;}
6688 #gsh header{height:100px;}
6689 #xgsh header{height:100px;background-image:url(GShell-Logo00.png);}
6690 #GshMenu{font-size:14pt;color:#c44;}
6691 .GshMenu{font-size:14pt;color:#c44;}
6692 .GshMenu1{
6693   font-size:14pt;color:#2a2;padding:4px;text-align:right;
6694 }
6695 .GshMenu1:hover{
6696   font-size:14pt;color:#fff;font-weight:bold;background-color:#2a2;
6697 }
6698 #GshFooter{height:100px;background-size:80px;background-repeat:no-repeat;}
6699 #gsh note{color:#000;font-size:10pt;}
6700 #gsh h2{color:#24a;font-family:Georgia;font-size:18pt;}
6701 #gsh h3{color:#24a;font-family:Georgia;font-size:16pt;}
6702 #gsh details{color:#888;background-color:#fff;font-family:monospace;}
6703 #gsh summary{font-size:16pt;color:#fff;background-color:#8af;xxxheight:30px;}
6704 #gsh summary{font-size:16pt;color:#fff;
6705   padding:0px;
6706   line-height:1.0;
6707   vertical-align:middle;
6708   xxx-background-color:#8af;
6709   background-color:#6881AD;xxx-Pblue;
6710   xxxheight:30px;
6711 }
6712 #gsh pre{font-size:11pt;color:#223;background-color:#fafff;}
6713 #gsh a{color:#24a;}
6714 #gsh a[name]{color:#24a;font-size:16pt;}
6715 #gsh .gsh-src{white-space:pre;font-family:monospace,Courier New;font-size:11pt;}
6716 #gsh .gsh-src{background-color:#fafff;color:#223;}
6717 #gsh-src{spellcheck:false}
6718 #gsh .gsh-src{white-space:pre;font-family:Courier New;font-size:10pt;}
6719 #gsh .gsh-code {white-space:pre;font-family:Courier New !important;}
6720 #gsh .gsh-code {color:#024;font-size:11pt; background-color:#fafff;}
6721 #gsh .gsh-golang-data {display:none;}
6722 #gsh .winid {color:#000;font-size:14pt;}
6723
6724
6725 .gsh-document {font-size:11pt;background-color:#fff;font-family:Georgia;}

```



```

7257 position:relative;
7258 top:0px; left:0px;
7259 border:0px solid #000; margin:0px; padding:0px;
7260 width:280px; height:160px;
7261 border:0px;
7262 font-family:Courier New,monospace !important;
7263 font-size:9pt;
7264 line-height:1.0;
7265 white-space:pre;
7266 color:#fff; background-color:rgba(0,0,64,0.5);
7267 background-color:rgba(32,32,128,0.8) !important;
7268 }
7269 .GJNode{
7270 display:inline;
7271 position:relative;
7272 top:0px; left:0px;
7273 border:0px solid #000; border-radius:0px;
7274 margin:0px; padding:0px;
7275 width:280px; height:20px;
7276 font-size:9pt;
7277 line-height:1.0;
7278 white-space:nowrap;
7279 color:#fff; background-color:rgba(0,0,64,0.7);
7280 text-align:left;
7281 vertical-align:middle;
7282 }
7283 </style>
7284
7285 <script id="gsh-script">
7286 // 2020-0909 added, permanet local storage
7287 // https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage
7288 var MyHistory
7289 Permanent = localStorage;
7290 MyHistory = Permanent.getItem('MyHistory')
7291 if( MyHistory == null ){ MyHistory = "" }
7292 d = new Date()
7293 MyHistory = d.getTime()/1000+ " " + document.URL + "\n" + MyHistory
7294 Permanent.setItem('MyHistory',MyHistory)
7295 //Permanent.setItem('MyWindow',window)
7296
7297 var GJLog_Win = null
7298 var GJLog_Tab = null
7299 var GJLog_Stat = null
7300 var GJLog_Text = null
7301 var GJWin_Mode = null
7302 var FProductInterval = 0
7303
7304 var GJ_FactoryID = -1
7305 var GJFactory = null
7306 if( e = document.getElementById('GJFactory_0') ){
7307   GJFactory_l.height = 0
7308   GJFactory = e
7309   e.setAttribute('class','GJFactory')
7310   var GJ_FactoryID = 0
7311 }else{
7312   GJFactory = GJFactory_l
7313   var GJ_FactoryID = 1
7314 }
7315
7316 function GJFactory_Destroy(){
7317   gjf = GJFactory
7318   //gjf = document.getElementById('GJFactory')
7319   //alert('gjf='+gjf)
7320   if( gjf != null ){
7321     if( gjf.childNodes != null ){
7322       for( i = 0; i < gjf.childNodes.length; i++ ){
7323         gjf.removeChild(gjf.childNodes[i])
7324       }
7325     }
7326     gjf.innerHTML = ''
7327     gjf.style.width = 0
7328     gjf.style.height = 0
7329     gjf.removeAttribute('style')
7330     GJLog_Win = GJLog_Tab = GJLog_Stat = GJLog_Text = GJWin_Mode = null
7331     window.clearInterval(FProductInterval)
7332     return '-- Destroy: work product destroyed'
7333   }else{
7334     return '-- Destroy: work product not exist'
7335   }
7336 }
7337
7338 var TransMode = false
7339 var onKeyControl = false
7340 var OnKeyShift = false
7341 var OnKeyAlt = false
7342 var OnKeyJ = false
7343 var OnKeyK = false
7344 var OnKeyL = false
7345
7346 function GJWin_OnKeyUp(ev){
7347   keycode = ev.code;
7348   if( keycode == 'ShiftLeft' ){
7349     OnKeyShift = false
7350   }else
7351   if( keycode == 'ControlLeft' ){
7352     onKeyControl = false
7353   }else
7354   if( keycode == 'AltLeft' ){
7355     OnKeyAlt = false
7356   }else
7357   if( keycode == 'KeyJ' ){ OnKeyJ = false }else
7358   if( keycode == 'KeyK' ){ OnKeyK = false }else
7359   if( keycode == 'KeyL' ){ OnKeyL = false }else
7360   {
7361     ev.preventDefault()
7362   }
7363 }
7364 function and(a,b){ if(a){ if(b){ return true; } return false; } }
7365 function GJWin_OnKeyDown(ev){
7366   keycode = ev.code;
7367   mode = ''
7368   key = ''
7369   if( keycode == 'ControlLeft' ){
7370     onKeyControl = true
7371     ev.preventDefault()
7372     return;
7373   }else
7374   if( keycode == 'ShiftLeft' ){
7375     OnKeyShift = true
7376     ev.preventDefault()
7377     return;
7378   }else
7379   if( keycode == 'AltLeft' ){
7380     ev.preventDefault()
7381     OnKeyAlt = true
7382     return;
7383   }else
7384   if( keycode == 'Backquote' ){
7385     TransMode = !TransMode
7386     ev.preventDefault()
7387   }else
7388   if( and(keycode == 'Space', OnKeyShift) ){
7389     TransMode = !TransMode
7390     ev.preventDefault()
7391   }else
7392   if( keycode == 'ShiftRight' ){
7393     TransMode = !TransMode
7394   }else
7395   if( keycode == 'Escape' ){
7396     TransMode = true
7397     ev.preventDefault()
7398   }else
7399   if( keycode == 'Enter' ){
7400     TransMode = false
7401     //ev.preventDefault()
7402   }
7403   if( keycode == 'KeyJ' ){ OnKeyJ = true }else
7404   if( keycode == 'KeyK' ){ OnKeyK = true }else
7405   if( keycode == 'KeyL' ){ OnKeyL = true }else
7406   {
7407     }
7408   if( ev.altKey ){ key += 'Alt+' }
7409   if( onKeyControl ){ key += 'Ctrl+' }
7410   if( OnKeyShift ){ key += 'Shift+' }
7411   if( and(keycode == 'KeyJ', OnKeyJ) ){ key += 'J+' }
7412   if( and(keycode == 'KeyK', OnKeyK) ){ key += 'K+' }
7413   if( and(keycode == 'KeyL', OnKeyL) ){ key += 'L+' }
7414   key += keycode
7415
7416   if( TransMode ){
7417     //mode = "[\343\201\202r]"
7418     JaAutF8 = new Uint8Array([0343,0201,0202]);
7419     utf8dec = new TextDecoder();
7420     JaA = utf8dec.decode(JaAutF8);
7421     mode = "[" + JaA + "r]";
7422   }
7423   }else{
7424     mode = '[---]'
7425   }
7426 }
7427 // gmode.innerHTML = "[---]"
7428 GJWin_Mode.innerHTML = mode + ' ' + key
7429 //alert('Key: '+keycode)
7430 ev.stopPropagation()
7431 //ev.preventDefault()
7432 }
7433 function GJWin_OnScroll(ev){

```

```

7434 x = DragStartX = gsh.getBoundingClientRect().left.toFixed(0)
7435 y = DragStartY = gsh.getBoundingClientRect().top.toFixed(0)
7436 GJLog.append('OnScroll: x='+x+',y='+y)
7437 }
7438 document.addEventListener('scroll',GJWin_OnScroll)
7439 function GJWin_OnResize(ev){
7440 w = window.innerWidth
7441 h = window.innerHeight
7442 GJLog.append('OnResize: w='+w+',h='+h)
7443 }
7444 window.addEventListener('resize',GJWin_OnResize)
7445
7446 var DragStartX = 0
7447 var DragStartY = 0
7448 function GJWin_DragStart(ev){
7449 // maybe this is the grabbing position
7450 this.style.position = 'fixed'
7451 x = DragStartX = this.getBoundingClientRect().left.toFixed(0)
7452 y = DragStartY = this.getBoundingClientRect().top.toFixed(0)
7453 GJLog.Stat.value = 'DragStart: x='+x+',y='+y
7454 }
7455 function GJWin_Drag(ev){
7456 x = ev.clientX; y = ev.clientY // x = ev.pageX; y = ev.pageY
7457 this.style.left = x - DragStartX
7458 this.style.top = y - DragStartY
7459 this.style.zIndex = '30000'
7460 this.style.position = 'fixed'
7461 x = this.getBoundingClientRect().left.toFixed(0)
7462 y = this.getBoundingClientRect().top.toFixed(0)
7463 GJLog.Stat.value = 'x='+x+',y'+y
7464 ev.preventDefault()
7465 ev.stopPropagation()
7466 }
7467 function GJWin_DragEnd(ev){
7468 x = ev.clientX; y = ev.clientY
7469 //x = ev.pageX; y = ev.pageY
7470 this.style.left = x - DragStartX
7471 this.style.top = y - DragStartY
7472 this.style.zIndex = '30000'
7473 this.style.position = 'fixed'
7474 if( true ){
7475 console.log("Dropped: "+this.nodeName+'#'+this.id+' x'+x+' y'+y
7476 +', parent'+this.parentNode.id)
7477 }
7478 x = this.getBoundingClientRect().left.toFixed(0)
7479 y = this.getBoundingClientRect().top.toFixed(0)
7480 GJLog.Stat.value = 'x='+x+',y'+y
7481 ev.preventDefault()
7482 ev.stopPropagation()
7483 }
7484 function GJWin_DragIgnore(ev){
7485 ev.preventDefault()
7486 ev.stopPropagation()
7487 }
7488 // 2020-09-15 let every object have console view!
7489 var GJ_ConsoleID = 0
7490 var PrevReport = new Date()
7491 function GJLog_StatUpdate(){
7492 txa = GJLog_Stat;
7493 if( txa == null ){
7494 return;
7495 }
7496 tmlap0 = new Date();
7497 p = txa.parentNode;
7498 pw = txa.getBoundingClientRect().width;
7499 ph = txa.getBoundingClientRect().height;
7500 //txa.value += '#'+p.id+' pw='+pw+', ph='+ph+'\n';
7501 txl = '#'+p.id+' pw='+pw+', ph='+ph+'\n';
7502 }
7503 w = txa.getBoundingClientRect().width;
7504 h = txa.getBoundingClientRect().height;
7505 //txa.value += 'w='+w+', h='+h+'\n';
7506 txl += 'w='+w+', h='+h+'\n';
7507 }
7508 //txa.value += '\n';
7509 //txa.value += DateShort() + '\n';
7510 txl += '\n';
7511 txl += DateShort() + '\n';
7512 tmlap1 = new Date();
7513 }
7514 txa.value += txl;
7515 tmlap2 = new Date();
7516 }
7517 // vertical centering of the last line
7518 sHeight = txa.scrollHeight - 30; // depends on the font-size
7519 tmlap3 = new Date();
7520 }
7521 txa.scrollTop = sHeight; // depends on the font-size
7522 tmlap4 = new Date();
7523 }
7524 now = tmlap0.getTime();
7525 if( PrevReport == 0 || 10000 <= now-PrevReport ){
7526 PrevReport = now;
7527 console.log('StatBarUpdate: '
7528 + 'leng=' + txa.value.length + ' byte, '
7529 + 'times=' + (tmlap4 - tmlap0) + ' ms {
7530 + 'tadd=' + (tmlap2 - tmlap1) + ', '
7531 + 'hcal=' + (tmlap3 - tmlap2) + ', '
7532 + 'sctl=' + (tmlap4 - tmlap3) + '}'
7533 );
7534 }
7535 }
7536 GJWin_StatUpdate = GJLog_StatUpdate;
7537 function GJ_showTime(wid){
7538 //e = document.getElementById(wid);
7539 //console.log(wid.id+' value.length='+wid.value.length)
7540 if( e == null ){
7541 //e.value = DateShort();
7542 }else{
7543 // should remove the Listener
7544 }
7545 }
7546 function GJWin_OnResizeTextarea(ev){
7547 this.value += 'resized: ' + '\n'
7548 }
7549 function GJ_NewConsole(wname){
7550 wid = wname + ' ' + GJ_ConsoleID
7551 GJ_ConsoleID += 1
7552 }
7553 GJFactory.style.setProperty('width',360+'px'); //GJfsize
7554 GJFactory.style.setProperty('height',320+'px')
7555 e = GJFactory;
7556 console.log('GJFA #' + e.id + ' from w=' + e.style.width + ', h=' + e.style.height)
7557 }
7558 if( GJFactory.innerHTML == "" ){
7559 GJFactory.innerHTML = '<' + 'H3>GJ Factory_' + GJ_FactoryID + '<' + '/H3><' + 'hr>\n'
7560 }else{
7561 GJFactory.innerHTML += '<' + 'hr>\n'
7562 }
7563 }
7564 gjwin = GJLog_Min = document.createElement('span')
7565 gjwin.id = wid
7566 gjwin.setAttribute('class','GJWin')
7567 gjwin.setAttribute('draggable','true')
7568 gjwin.addEventListener('dragstart',GJWin_DragStart)
7569 gjwin.addEventListener('drag',GJWin_Drag)
7570 gjwin.addEventListener('dragend',GJWin_Drag)
7571 gjwin.addEventListener('dragover',GJWin_DragIgnore)
7572 gjwin.addEventListener('dragenter',GJWin_DragIgnore)
7573 gjwin.addEventListener('dragleave',GJWin_DragIgnore)
7574 gjwin.addEventListener('dragexit',GJWin_DragIgnore)
7575 gjwin.addEventListener('drop',GJWin_DragIgnore)
7576 gjwin.addEventListener('keydown',GJWin_OnKeyDown)
7577 }
7578 gjtab = GJLog_Tab = document.createElement('textarea')
7579 gjtab.addEventListener('keydown',GJWin_OnKeyDown)
7580 gjtab.style.readonly = true
7581 gjtab.contentEditable = false
7582 gjtab.value = wid
7583 gjtab.id = wid + ' Tab'
7584 gjtab.setAttribute('class','GJTab')
7585 gjtab.setAttribute('spellcheck','false')
7586 gjwin.appendChild(gjtab)
7587 }
7588 gjstat = GJLog_Stat = document.createElement('textarea')
7589 gjstat.addEventListener('keydown',GJWin_OnKeyDown)
7590 gjstat.id = wid + ' Stat'
7591 gjstat.value = DateShort()
7592 gjstat.setAttribute('class','GJStat')
7593 gjstat.setAttribute('spellcheck','false')
7594 gjwin.appendChild(gjstat)
7595 }
7596 gjicon = document.createElement('span')
7597 gjicon.addEventListener('keydown',GJWin_OnKeyDown)
7598 gjicon.id = wid + ' Icon'
7599 gjicon.innerHTML = '< font color=#f44>J</font>'
7600 gjicon.setAttribute('class','GJIcon')
7601 gjicon.setAttribute('spellcheck','false')
7602 gjwin.appendChild(gjicon)
7603 }
7604 gjtext = GJLog_Text = document.createElement('textarea')
7605 gjtext.addEventListener('keydown',GJWin_OnKeyDown)
7606 gjtext.addEventListener('keyup',GJWin_OnKeyUp)
7607 gjtext.addEventListener('resize',GJWin_OnResizeTextarea)
7608 gjtext.id = wid + ' Text'
7609 gjtext.setAttribute('class','GJText')
7610 gjtext.setAttribute('spellcheck','false')

```

```

7611     gJwin.appendChild(gjtext)
7612
7613
7614     // user's mode as of IME
7615     gJmode = GJwin_Mode = document.createElement('textarea')
7616     gJmode.addEventListener('keydown', GJwin_OnKeyDown)
7617     gJmode.addEventListener('keydown', GJwin_OnKeyDown)
7618     gJmode.id = wid + '_Mode'
7619     gJmode.setAttribute('class', 'GJMode')
7620     gJmode.setAttribute('spellcheck', 'false')
7621     gJmode.innerHTML = `---`
7622     gJwin.appendChild(gJmode)
7623
7624     gJwin.zIndex = 30000
7625     GJFactory.appendChild(gJwin)
7626
7627     gjtab.scrollTop = 0
7628     gjstat.scrollTop = 0
7629
7630     //x = gJwin.getBoundingClientRect().left.toFixed(0)
7631     //y = gJwin.getBoundingClientRect().top.toFixed(0)
7632     //gJwin.style.position = 'static'
7633     //gJwin.style.left = 0
7634     //gJwin.style.top = 0
7635
7636     //update = `{"wid+`.value=DateShort()`,
7637     update = `(GJ_showTime1("wid+`);`,
7638     // 2020-09-19 this causes memory leaks
7639     //ProductInterval = window.setInterval(update,200)
7640     //ProductInterval = window.setInterval(GJWin_StatUpdate,200)
7641     //ProductInterval = window.setInterval(GJ_showTime1,200,wid);
7642     ProductInterval = window.setInterval(GJ_showTime1,200,gjstat);
7643     return update
7644 }
7645
7646 function xxxGJF_StripeClass(){
7647     GJLog_Win.style.removeProperty('width')
7648     GJLog_Tab.style.removeProperty('width')
7649     GJLog_Stat.style.removeProperty('width')
7650     GJLog_Text.style.removeProperty('width')
7651     return "Stripped classes"
7652 }
7653
7654 function isElem(id){
7655     return document.getElementById(id) != null
7656 }
7657
7658 function GJLog_append(...args){
7659     txt = GJLog_Text;
7660     if( txt == null ){
7661         return; // maybe GJLog element is removed
7662     }
7663     logs = args.join(' ');
7664     txt.value += logs + '\n'
7665     txt.scrollTop = txt.scrollHeight
7666     //GJLog_Stat.value = DateShort()
7667 }
7668 //window.addEventListener('time',GJLog_StatUpdate)
7669
7670 function test_GJ_Console(){
7671     window.setInterval(GJLog_StatUpdate,1000);
7672     GJ_NewConsole('GJ_Console')
7673     e = GJFactory;
7674     console.log("GJF0 #'+e.id+' from w='+e.style.width+', h='+e.style.height)
7675     e.style.width = 360; //GJFsize
7676     e.style.height = 320;
7677     console.log("GJF0 #'+e.id+' to w='+e.style.width+', h='+e.style.height)
7678 }
7679
7680 // test_GJ_Console();
7681
7682 var StopConsoleLog = true
7683 // 2020-09-15 added,
7684 // log should be saved to permanent memory
7685 // const px = new Proxy(console.log,{ alert() })
7686
7687 console_log = console.log
7688 console_info = console.info
7689 console_warn = console.warn
7690 console_error = console.error
7691 console_exception = console.exception
7692 // should pop callstack info.
7693 console_exception = function(...args){
7694     console_exception(...args)
7695     alert('-- got console.exception("+args+")')
7696 }
7697 console.error = function(...args){
7698     console_error(...args)
7699     alert('-- got console.error("+args+")')
7700 }
7701 console.warn = function(...args){
7702     console_warn(...args)
7703     alert('-- got console.warn("+args+")')
7704 }
7705 console.info = function(...args){
7706     alert('-- got console.info("+args+")')
7707     console_info(...args)
7708 }
7709
7710 console.xxxlog = function(...args){ // rewrite xxxlog to log to enable it
7711     console_log(...args)
7712     if( StopConsoleLog ){
7713         return;
7714     }
7715     if( 0 <= args[0].indexOf('!') ){
7716         //alert('-- got console.log("+args+")')
7717     }
7718     GJLog_append(...args)
7719 }
7720
7721 //document.getElementById('GshFaviconURL').href = GShellFavicon
7722 //document.getElementById('GshFaviconURL').href = GShellInsideIcon
7723 //document.getElementById('GshFaviconURL').href = ITSmoreQR
7724 //document.getElementById('GshFaviconURL').href = GShellLogo
7725
7726 // id of GShell HTML elements
7727 var E_BANNER = "GshBanner" // banner element in HTML
7728 var E_FOOTER = "GshFooter" // footer element in HTML
7729 var E_GINDEX = "gsh-gindex" // index of Golang code of GShell
7730 var E_GOCODE = "gsh-gocode" // Golang code of GShell
7731 var E_TODO = "gsh-todo" // TODO of GShell
7732 var E_DICT = "gsh-dict" // Dictionary of GShell
7733
7734 function bannerElem(){ return document.getElementById(E_BANNER); }
7735 function bannerStyleFunc(){ return bannerElem().style; }
7736 var bannerStyle = bannerStyleFunc()
7737
7738 function GshSetImages(){
7739     document.getElementById('GshFaviconURL').href = GShellInsideIcon
7740     bannerStyle.backgroundImage = "url("+GShellLogo+")";
7741     //bannerStyle.backgroundImage = "url("+GShellInsideIcon+")";
7742     //bannerStyle.backgroundImage = "url("+GShellFavicon+")";
7743     //GMenu.style.backgroundImage = "url("+GShellInsideIcon+")";
7744     //showFooter();
7745 }
7746
7747 function GshInsideIconSetup(){
7748     GMenu.style.backgroundImage = "url("+GShellInsideIcon+")";
7749     GMenu.style.zIndex = 10000000;
7750     //GMenu.style.left = window.innerWidth - 100
7751     GMenu.style.left = 0;
7752     GMenu.style.top = window.innerHeight - 90; // - 200
7753     window.addEventListener('resize',GshInsideIconSetup);
7754 }
7755
7756 function footerElem(){ return document.getElementById(E_FOOTER); }
7757 function footerStyle(){ return footerElem().style; }
7758 //footerElem().style.backgroundImage="url("+ITSmoreQR+")";
7759 //footerStyle().backgroundImage = "url("+ITSmoreQR+")";
7760
7761 function html_fold(e){
7762     if( e.innerHTML == "Fold" ){
7763         e.innerHTML = "Unfold"
7764         document.getElementById('gsh-menu-exit').innerHTML=""
7765         document.getElementById('GshStatement').open=false
7766         GshFeatures.open = false
7767         document.getElementById('html-src').open=false
7768         document.getElementById(E_GINDEX).open=false
7769         document.getElementById(E_GOCODE).open=false
7770         document.getElementById(E_TODO).open=false
7771         document.getElementById('References').open=false
7772     }else{
7773         e.innerHTML = "Fold"
7774         document.getElementById('GshStatement').open=true
7775         GshFeatures.open = true
7776         document.getElementById(E_GINDEX).open=true
7777         document.getElementById(E_GOCODE).open=true
7778         document.getElementById(E_TODO).open=true
7779         document.getElementById('References').open=true
7780     }
7781 }
7782
7783 function html_pure(e){
7784     if( e.innerHTML == "Pure" ){
7785         document.getElementById('gsh').style.display=true
7786         //document.style.display = false
7787         e.innerHTML = "Unpure"
7788     }else{
7789         document.getElementById('gsh').style.display=false
7790         //document.style.display = true
7791         e.innerHTML = "Pure"
7792     }
7793 }
7794
7795 var bannerIsStopping = false
7796 //NOTE: .com/JSREF/prop_style_backgroundposition.asp
7797 function shiftBG(){

```

```

7788 bannerIsStopping = !bannerIsStopping
7789 bannerStyle.backgroundColor = "0 0";
7790 }
7791 // status should be inherited on Window Fork(), so use the status in DOM
7792 function html_stop(e,toggle){
7793     if( toggle ){
7794         if( e.innerHTML == "Stop" ){
7795             bannerIsStopping = true
7796             e.innerHTML = "Start"
7797         }else{
7798             bannerIsStopping = false
7799             e.innerHTML = "Stop"
7800         }
7801     }else{
7802         // update JavaScript variable from DOM status
7803         if( e.innerHTML == "Stop" ){ // shown if it's running
7804             bannerIsStopping = false
7805         }else{
7806             bannerIsStopping = true
7807         }
7808     }
7809 }
7810 html_stop(document.getElementById('GshMenuStop'),false) // onInit.
7811 //html_stop(bannerElem(),false) // onInit.
7812 //https://www.w3schools.com/jsref/met_win_setinterval.asp
7813 var banNshft = 0;
7814 function consoleLog(str){
7815     //console.log(str);
7816 }
7817 function shiftBanner(){
7818     var now = new Date().getTime();
7819     bpos = ((now/10%10000).toFixed(0)+'px' + " 0px";
7820     if( !bannerIsStopping ){
7821         bannerStyle.backgroundColor = bpos;
7822         //GshBanner.style.setProperty('background-position',bpos,'important');
7823         banNshft += 1;
7824         console.log("shiftBanner <"+GshBanner.nodeName+"> "+banNshft
7825             + " now="+now+";
7826             //+ ' stop='+bannerIsStopping
7827             + ' pos'+bpos
7828             + ' -> '+bannerStyle.backgroundColor);
7829     }
7830 }
7831 }
7832 function Banner_init(){
7833     console.log("-- Banner Shift init.");
7834     window.setInterval(shiftBanner,10); // onInit.
7835 }
7836 }
7837 // <a href="https://developer.mozilla.org/ja/docs/Web/API/Window/open">window.open()</a>
7838 // from embedded html to standalone page
7839 var MyChildren = 0
7840 function html_fork(){
7841     ResetPerfMon();
7842     ResetAffView();
7843     Reset_ShadingCanvas();
7844     GJFactory_Destroy();
7845     MyChildren += 1
7846     WinId = document.getElementById('gsh-WinId').innerHTML + "-" + MyChildren;
7847     newwin = window.open("",WinId,"");
7848     src = document.getElementById("gsh");
7849     srctml = src.outerHTML
7850     newwin.document.write("<"+srctml>");
7851     newwin.document.write(srctml);
7852     newwin.document.write("<"+"/html>");
7853     newwin.document.getElementById('gsh-menu-exit').innerHTML = "Close";
7854     newwin.document.getElementById('gsh-WinId').innerHTML = WinId;
7855     newwin.document.close();
7856     newwin.focus();
7857 }
7858 function html_close(){
7859     window.close()
7860 }
7861 function win_jump(win){
7862     //win = window.top;
7863     win = window.opener; // https://developer.mozilla.org/ja/docs/Web/API/window/opener
7864     if( win == null ){
7865         console.log("jump to window.opener("+win+") (Error)\n");
7866     }else{
7867         console.log("jump to window.opener("+win+")\n");
7868         win.focus();
7869     }
7870 }
7871 }
7872 // 0.2.9 2020-0902 created checksum of HTML
7873 CRC32UNIX = 0x04c11db7 // Unix cksum
7874 function byteCRC32add(bigcrc,octstr,octlen){
7875     var crc = new Uint32Array(1)
7876     crc[0] = bigcrc
7877     let oi = 0
7878     for( ; oi < octlen; oi++){
7879         var oct = new Uint8Array(1)
7880         oct[0] = octstr[oi]
7881         for( bi = 0; bi < 8; bi++){
7882             //console.log("--CRC32 "+crc[0]+" "+oct[0].toString(16)+" ["+oi+"."+bi+"]\n")
7883             ovf1 = crc[0] < 0 ? 1 : 0
7884             ovf2 = oct[0] < 0 ? 1 : 0
7885             ovf = ovf1 ^ ovf2
7886             oct[0] <<= 1
7887             crc[0] <<= 1
7888             if( ovf ){ crc[0] ^= CRC32UNIX }
7889         }
7890     }
7891     //console.log("--CRC32 byteAdd return crc="+crc[0]+" "+oi+" "+octlen+"\n")
7892     return crc[0];
7893 }
7894 }
7895 function strCRC32add(bigcrc,stri,striLen){
7896     var crc = new Uint32Array(1)
7897     crc[0] = bigcrc
7898     var code = new Uint8Array(striLen);
7899     for( i = 0; i < striLen; i++){
7900         code[i] = stri.charCodeAtAt(i) // not charAt() !!!!
7901         //console.log("==="+code[i].toString(16)+" <<="+stri[i]+"")
7902     }
7903     crc[0] = byteCRC32add(crc,code,striLen)
7904     //console.log("--CRC32 strAdd return crc="+crc[0]+"")
7905     return crc[0];
7906 }
7907 }
7908 function byteCRC32end(bigcrc,len){
7909     var crc = new Uint32Array(1)
7910     crc[0] = bigcrc
7911     var slen = new Uint8Array(4)
7912     let li = 0
7913     for( ; li < 4; ){
7914         slen[li] = len
7915         li += 1
7916         len >>= 8
7917         if( len == 0 ){
7918             break
7919         }
7920     }
7921     crc[0] = byteCRC32add(crc[0],slen,li)
7922     crc[0] = 0xFFFFFFFF
7923     return crc[0];
7924 }
7925 }
7926 function strCRC32(stri,len){
7927     var crc = new Uint32Array(1)
7928     crc[0] = 0
7929     crc[0] = strCRC32add(0,stri,len)
7930     crc[0] = byteCRC32end(crc[0],len)
7931     //console.log("--CRC32 "+crc[0]+" "+len+"\n")
7932     return crc[0];
7933 }
7934 }
7935 DestroyGJLink = null; // to be replaced
7936 DestroyFooter = null; // to be defined
7937 DestroyEventSharingCodeview = function dummy(){
7938 Destroy_VirtualDesktop = function(){
7939 DestroyNavButtons = function(){
7940 }
7941 function getSourceText(){
7942     if( DestroyFooter != null ) DestroyFooter();
7943     version = document.getElementById('GshVersion').innerHTML
7944     sfavico = document.getElementById('GshFaviconURL').href;
7945     sbanner = document.getElementById('GshBanner').style.backgroundColor;
7946     spositi = document.getElementById('GshBanner').style.backgroundColor;
7947     if( document.getElementById('GJC_1') != null ){ GJC_1.remove() }
7948     if( DestroyGJLink != null ) DestroyGJLink();
7949     DestroyEventSharingCodeview();
7950     Destroy_VirtualDesktop();
7951     GshTopbar.innerHTML = "";
7952     DestroyIndexBar();
7953     DestroyNavButtons();
7954     ResetPerfMon();
7955     ResetAffView();
7956     Reset_ShadingCanvas();
7957     // these should be removed by CSS selector or class, after seaved to non-printed attribute
7958     GshBanner.removeAttribute("style");
7959     document.getElementById('GshMenuSign').removeAttribute("style");
7960     styleGMenu = GMenu.getAttribute("style")
7961     GMenu.removeAttribute("style");
7962     styleGStat = GStat.getAttribute("style")
7963     GStat.removeAttribute("style");
7964     styleGTop = GTop.getAttribute("style")

```

```

7965 GTop.removeAttribute("style");
7966 styleGshGrid = GshGrid.getAttribute("style")
7967 GshGrid.removeAttribute("style");
7968 //styleGPos = GPos.getAttribute("style");
7969 //GPos.removeAttribute("style");
7970 //GPos.innereHTML = "";
7971 //styleGLog = GLog.getAttribute("style");
7972 //GLog.removeAttribute("style");
7973 //GLog.innereHTML = "";
7974 styleGShellPlane = GShellPlane.getAttribute("style")
7975 GShellPlane.removeAttribute("style")
7976 styleRawTextViewer = RawTextViewer.getAttribute("style")
7977 RawTextViewer.removeAttribute("style")
7978 styleRawTextViewerClose = RawTextViewerClose.getAttribute("style")
7979 RawTextViewerClose.removeAttribute("style")
7980
7981 GshFaviconURL.href = "";
7982 if( iselem('ConfigIcon') ) ConfigIcon.src = "";
7983
7984 //it seems that interHTML and outerHTML generate style="" for these (??)
7985 //GshBanner.removeAttribute("style");
7986 //GshFooter.removeAttribute("style");
7987 //GshMenuSign.removeAttribute("style");
7988 GshBanner.style="";
7989 GshMenuSign.style="";
7990
7991 textarea = document.createElement("textarea")
7992 srchtml = document.getElementById("gsh").outerHTML;
7993 //textarea = document.createElement("textarea")
7994 // 2020-0910 ?? ... this causes inserting style="" to Banner and Footer,
7995 // with Chromium/ after reloading from file:///
7996 textarea.innereHTML = srchtml;
7997 // <a href="https://stackoverflow.com/questions/5796718/html-entity-decode">Thanks</a>
7998 var rawtext = textarea.value
7999 //textarea.destory()
8000 //rawtext = gsh.textContent // this removes #include <FILENAME> too
8001 var orgtext = ""
8002 + "/<<+>html>\n" // lost preamble text
8003 + rawtext
8004 + "<+>/html>\n" // lost trail text
8005 ;
8006
8007 tlen = orgtext.length
8008 //console.log("getSourceText: length="+tlen+"\n")
8009 document.getElementById("GshFaviconURL").href = sfavico;
8010
8011 document.getElementById("GshBanner").style.backgroundImage = sbanner;
8012 document.getElementById("GshBanner").style.backgroundPosition = spositii;
8013
8014 GStat.setAttribute("style",styleGStat)
8015 GMenu.setAttribute("style",styleGMenu)
8016 GTop.setAttribute("style",styleGTop)
8017 //GLog.setAttribute("style",styleGLog)
8018 //GPos.setAttribute("style",styleGPos)
8019 GshGrid.setAttribute("style",styleGshGrid)
8020 GShellPlane.setAttribute("style",styleGShellPlane)
8021 RawTextViewer.setAttribute("style",styleRawTextViewer)
8022 RawTextViewerClose.setAttribute("style",styleRawTextViewerClose)
8023 canontext = orgtext.replace(' style=""','')
8024 // gsetat too
8025 return canontext
8026 }
8027
8028 function getDigest(){
8029 var text = ""
8030 text = getSourceText()
8031 var digest = ""
8032 tlen = text.length
8033 digest = strCRC32(text,tlen) + " " + tlen
8034 return { text, digest }
8035 }
8036
8037 function html_digest(){
8038 version = document.getElementById("GshVersion").innerHTML
8039 let {text, digest} = getDigest()
8040 alert("cksum: " + digest + " " + version)
8041 }
8042
8043 function charsin(str, char){
8044 ln = 0;
8045 for( i = 0; i < str.length; i++ ){
8046 if( str.charCodeAt(i) == char.charCodeAt(0) )
8047 ln++;
8048 }
8049 return ln;
8050 }
8051
8052 //class digestElement extends HTMLElement {
8053 //< script>customElements.define('digest',digestElement)< /script>
8054 function showDigest(e){
8055 result = 'version=' + GshVersion.innerHTML + '\n'
8056 result = 'lines=' + e.dataset.lines + '\n'
8057 result = 'length=' + e.dataset.length + '\n'
8058 result = 'crc32u=' + e.dataset.crc32u + '\n'
8059 result = 'time=' + e.dataset.time + '\n';
8060 alert(result)
8061 }
8062
8063 function html_sign(e){
8064 if( RawTextViewer.style.zIndex == 1000 ){
8065 hideRawTextViewer()
8066 return
8067 }
8068 GshTopbar.innereHTML = "";
8069 ResetParMon();
8070 ResetAffView();
8071 ResetShadingCanvas();
8072 DestroyIndexBar();
8073 DestroyViviButtons();
8074 DestroyEventSharingCodeview();
8075 DestroyWirTualDesktop();
8076 GJFactory_Destroy();
8077 if( DestroyGJLink != null ) DestroyGJLink();
8078 //gsh digest .innereHTML = "";
8079 text = getSourceText() // the original text
8080 tlen = text.length
8081 digest = strCRC32(text,tlen)
8082 //gsh digest .innereHTML = digest + " " + tlen
8083 //text = getSourceText(), // the text with its digest
8084 Lines = charsin(text, '\n')
8085
8086 name = "gsh"
8087 sid = name + "-digest"
8088 d = new Date()
8089 signedAt = d.getTime()
8090
8091 sign = '/' + '<' + 'span'\n'
8092 + ' id="" + sid + "\n'
8093 + ' class="" digest '\n'
8094 + ' data-target-id="" name=""\n'
8095 + ' data-crc32u="" + digest + '\n'
8096 + ' data-length="" + tlen + '\n'
8097 + ' data-lines="" + Lines + '\n'
8098 + ' data-time="" + signedAt + '\n'
8099 + '>' + '/span>\n'+ '\n'
8100
8101 text = sign + text
8102
8103 txthtml = '<' + 'table id="LineNumbered"><' + 'tr<' + 'td'
8104 + '<' + 'textarea cols=5 rows= ' + Lines + ' class="LineNumber">'
8105 for( i = 1; i <= Lines; i++ ){
8106 txthtml += i.toString() + '\n'
8107 }
8108 txthtml += ""
8109 + '<' + '/textarea'
8110 + '<' + '/td<' + '>'
8111 + '<' + 'textarea cols=150 rows= ' + Lines + ' spellcheck="false"
8112 + ' class="LineNumbered">'
8113 + text + '<' + '/textarea'
8114 + '<' + '/td<' + '>' + '/tr<' + '/table>'
8115
8116 for( i = 1; i <= 30; i++ ){
8117 txthtml += '<br>\n'
8118 }
8119 RawTextViewer.innereHTML = txthtml
8120 RawTextViewer.spellcheck = false // (spellcheck above seems ineffective)
8121
8122 btn = e
8123 e.style.color = "rgba(128,128,255,0.9)";
8124 y = e.getBoudingClientRect().top.toFixed(0)
8125 //h = e.getBoudingClientRect().height.toFixed(0)
8126 RawTextViewer.style.top = Number(y) + 30
8127 RawTextViewer.style.left = 100;
8128 RawTextViewer.style.height = window.innerHeight - 20;
8129 //RawTextViewer.style.Opacity = 1.0;
8130 //RawTextViewer.style.backgroundColor = "rgba(0,0,0,0.0)";
8131 RawTextViewer.style.backgroundColor = "rgba(255,255,255,0.8)";
8132 RawTextViewer.style.zIndex = 1000;
8133 RawTextViewer.style.display = true;
8134
8135 if( RawTextViewerClose.style == null ){
8136 RawTextViewerClose.style = "";
8137 }
8138 RawTextViewerClose.style.top = Number(y) + 10
8139 RawTextViewerClose.style.left = 100;
8140 RawTextViewerClose.style.zIndex = 1001;
8141 ScrollToElement(CurElement,RawTextViewerClose)
8142 }

```

```

8142 function hideRawTextViewer(){
8143 RawTextViewer.style.left = 10000;
8144 RawTextViewer.style.zIndex = 1000;
8145 RawTextViewer.style.Opacity = 0.0;
8146 RawTextViewer.style = null;
8147 RawTextViewer.innerHTML = "";
8148
8149 GshMenuSign.style.color = "rgba(255,128,128,1.0)";
8150 RawTextViewerClose.style.top = 0;
8151 RawTextViewerClose.style = null;
8152 }
8153
8154 // source code view
8155 function frame_close(){
8156 srcframe = document.getElementById("src-frame");
8157 srcframe.innerHTML = "";
8158 //srcframe.style.cols = 1;
8159 srcframe.style.rows = 1;
8160 srcframe.style.height = 0;
8161 srcframe.style.display = false;
8162 src = document.getElementById("SrcTextarea");
8163 src.innerHTML = "";
8164 //src.cols = 0
8165 src.rows = 0
8166 src.display = false
8167 //alert("--closed--")
8168 }
8169 //<!-- | <span onclick="html_view();">Source</span> -->
8170 //<!-- | <span onclick="frame_close();">SourceClose</span> -->
8171 //<!-- | <span>Download</span> -->
8172 function frame_open(){
8173 GshNavbar.innerHTML = "";
8174 ResetPerfMon();
8175 ResetAfvView();
8176 Reset_ShadingCanvas();
8177 DestroyIndexBar();
8178 DestroyNavButtons();
8179 if( DestroyFooter != null ) DestroyFooter();
8180 document.getElementById("GshFavIconURL").href = "";
8181 oldsrc = document.getElementById("GENSRC");
8182 if( oldsrc != null ){
8183 //alert("--I--(excasing old text)")
8184 oldsrc.innerHTML = "";
8185 return
8186 }else{
8187 //alert("--I--(no old text)")
8188 }
8189 styleBanner = GshBanner.getAttribute("style")
8190 GshBanner.removeAttribute("style")
8191 if( document.getElementById("GJC_1") ){ GJC_1.remove() }
8192
8193 GshFavIconURL.href = "";
8194 if( isIcon("ConfigIcon") ) ConfigIcon.src = "";
8195 GStat.removeAttribute("style")
8196 GshGrid.removeAttribute("style")
8197 GshMenuSign.removeAttribute("style")
8198 //GPos.removeAttribute("style")
8199 //GPos.innerHTML = "";
8200 //GLog.removeAttribute("style")
8201 //GLog.innerHTML = "";
8202 GMenu.removeAttribute("style")
8203 GTop.removeAttribute("style")
8204 GShellPlane.removeAttribute("style")
8205 RawTextViewer.removeAttribute("style");
8206 RawTextViewerClose.removeAttribute("style")
8207
8208 if( DestroyGJLink != null ) DestroyGJLink();
8209 GJFactory_Destroy()
8210 Destroy_VirtualDesktop();
8211 DestroyEventSharingCodeview();
8212
8213 src = document.getElementById("gsh");
8214 srchtml = src.outerHTML;
8215 srcframe = document.getElementById("src-frame");
8216 srcframe.innerHTML =
8217 + "<+>cite id=\"GENSRC\">\n"
8218 + "<+>style\n"
8219 + "#GENSRC textarea{tab-size:4;}\n"
8220 + "#GENSRC textarea{-o-tab-size:4;}\n"
8221 + "#GENSRC textarea{-moz-tab-size:4;}\n"
8222 + "#GENSRC textarea{spellcheck:false;}\n"
8223 + "</+>\n"
8224 + "<+>textarea id=\"SrcTextarea\" cols=100 rows=20 class=\"gsh-code\" spellcheck=\"false\">
8225 + "<+>html\n" // lost preamble text
8226 + srchtml
8227 + "<+>/html\n" // lost trail text
8228 + "<+>textarea\n"
8229 + "</+>cite<!-- GENSRC -->\n";
8230
8231 //srcframe.style.cols = 80;
8232 //srcframe.style.rows = 80;
8233
8234 GshBanner.setAttribute("style",styleBanner)
8235 }
8236 function fill_CSSView(){
8237 part = document.getElementById("GshStyleDef")
8238 view = document.getElementById("gsh-style-view")
8239 view.innerHTML = ""
8240 + "<+>textarea cols=100 rows=20 class=\"gsh-code\">
8241 + part.innerHTML
8242 + "<+>/textarea"
8243 }
8244 function fill_JavaScriptView(){
8245 jspart = document.getElementById("gsh-script")
8246 view = document.getElementById("gsh-script-view")
8247 view.innerHTML = ""
8248 + "<+>textarea cols=100 rows=20 class=\"gsh-code\">
8249 + jspart.innerHTML
8250 + "<+>/textarea"
8251 }
8252 function fill_DataView(){
8253 part = document.getElementById("gsh-data")
8254 view = document.getElementById("gsh-data-view")
8255 view.innerHTML = ""
8256 + "<+>textarea cols=100 rows=20 class=\"gsh-code\">
8257 + part.innerHTML
8258 + "<+>/textarea"
8259 }
8260 function jumpto_StyleView(){
8261 jview = document.getElementById("html-src")
8262 jview.open = true
8263 jview = document.getElementById("gsh-style-frame")
8264 jview.open = true
8265 fill_CSSView()
8266 }
8267 function jumpto_JavaScriptView(){
8268 jview = document.getElementById("html-src")
8269 jview.open = true
8270 jview = document.getElementById("gsh-script-frame")
8271 jview.open = true
8272 fill_JavaScriptView()
8273 }
8274 function jumpto_DataView(){
8275 jview = document.getElementById("html-src")
8276 jview.open = true
8277 jview = document.getElementById("gsh-data-frame")
8278 jview.open = true
8279 fill_DataView()
8280 }
8281 function jumpto_WholeView(){
8282 jview = document.getElementById("html-src")
8283 jview.open = true
8284 jview = document.getElementById("gsh-whole-view")
8285 jview.open = true
8286 frame_open()
8287 }
8288 function html_view(){
8289 html_stop();
8290
8291 banner = document.getElementById("GshBanner").style.backgroundImage;
8292 footer = document.getElementById("GshFooter").style.backgroundImage;
8293 document.getElementById("GshBanner").style.backgroundImage = "";
8294 document.getElementById("GshBanner").style.backgroundPosition = "";
8295 document.getElementById("GshFooter").style.backgroundImage = "";
8296
8297 //srcwin = window.open("", "CodeView2", "");
8298 srcwin = window.open("", "");
8299 srcwin.document.write("<span id=\"gsh\">\n");
8300
8301 src = document.getElementById("gsh");
8302 srcwin.document.write("<+>style\n");
8303 srcwin.document.write("<+>textarea{tab-size:4;}\n");
8304 srcwin.document.write("<+>textarea{-o-tab-size:4;}\n");
8305 srcwin.document.write("<+>textarea{-moz-tab-size:4;}\n");
8306 srcwin.document.write("</+>\n");
8307 srcwin.document.write("<h2>\n");
8308 srcwin.document.write("<+>span onclick=\"window_close();\">Close</span> | \n");
8309 //srcwin.document.write("<+>span onclick=\"html_stop();\">Run</span>\n");
8310 srcwin.document.write("</h2>\n");
8311 srcwin.document.write("<+>textarea id=\"gsh-src-src\" cols=100 rows=60>");
8312 srcwin.document.write("<+>html\n");
8313 srcwin.document.write("<+>span id=\"gsh\">");
8314 srcwin.document.write(src.innerHTML);
8315 srcwin.document.write("</+>/span<+>/html\n");
8316 srcwin.document.write("<+>textarea\n");
8317
8318 document.getElementById("GshBanner").style.backgroundImage = banner;

```

```

8319 document.getElementById('GshFooter').style.backgroundImage = footer
8320
8321 sty = document.getElementById("GshStyleDef");
8322 srcwin.document.write("<"+style>\n");
8323 srcwin.document.write(sty.innerHTML);
8324 srcwin.document.write("<"+style>\n");
8325
8326 run = document.getElementById("gsh-script");
8327 srcwin.document.write("<"+script>\n");
8328 srcwin.document.write(run.innerHTML);
8329 srcwin.document.write("<"+script>\n");
8330
8331 srcwin.document.write("<"+span>"+html>\n"); // gsh span
8332 srcwin.document.close();
8333 srcwin.focus();
8334
8335 GSH = document.getElementById("gsh")
8336
8337 //GSH.onclick = "alert('Ouch!')"
8338 //GSH.css = "{background-color:#eef;}"
8339 //GSH.style = "background-color:#eef;"
8340 //GSH.style.display = false;
8341 //alert('Ouch0!')
8342 //GSH.style.display = true;
8343
8344 // 2020-0904 created, tentative
8345 //document.addEventListener('keydown', jgshCommand);
8346 //CurElement = GshStatement
8347 CurElement = GshMenu
8348 MemElement = GshMenu
8349
8350 function nextSib(e){
8351   n = e.nextSibling;
8352   for( i = 0; i < 100; i++ ){
8353     if( n == null ){
8354       break;
8355     }
8356     if( n.nodeName == "DETAILS" ){
8357       return n;
8358     }
8359     n = n.nextSibling;
8360   }
8361   return null;
8362 }
8363 function prevSib(e){
8364   n = e.previousSibling;
8365   for( i = 0; i < 100; i++ ){
8366     if( n == null ){
8367       break;
8368     }
8369     if( n.nodeName == "DETAILS" ){
8370       return n;
8371     }
8372     n = n.previousSibling;
8373   }
8374   return null;
8375 }
8376 function setColor(e,eName,eColor){
8377   if( e.hasChildNodes() ){
8378     e = e.childNodes;
8379     if( s != null ){
8380       for( ci = 0; ci < s.length; ci++ ){
8381         if( s[ci].nodeName == eName ){
8382           s[ci].style.color = eColor;
8383           //s[ci].style.backgroundColor = eColor;
8384           break;
8385         }
8386       }
8387     }
8388   }
8389 }
8390
8391 // https://docs.microsoft.com/en-us/previous-versions//hh781509(v=vs.85)
8392 function showCurElementPosition(ev){
8393   // if( document.getElementById("GPos") == null ){
8394     return;
8395   // }
8396   // if( GPos == null ){
8397     return;
8398   // }
8399   e = CurElement
8400   y = e.getBoundingClientRect().top.toFixed(0)
8401   x = e.getBoundingClientRect().left.toFixed(0)
8402
8403   h = ev + " "
8404   h += 'y'+y+" "+ 'x'+x+" -- "
8405   h += "w" + window.innerWidth + ", h" + window.innerHeight + " -- "
8406   //GPos.test = h
8407   //GPos.innerHTML = h
8408   // GPos.innerHTML = h
8409 }
8410
8411 function zero2(n){
8412   if( n < 10 ){
8413     return '0' + n;
8414   }else{
8415     return n;
8416   }
8417 }
8418 function DateHourMin(){
8419   d = new Date();
8420   //return '%02d:%02d'.sprintf(d.getHours(),d.getMinutes())
8421   return zero2(d.getHours()) + ":" + zero2(d.getMinutes());
8422 }
8423 function DateShort0(d){
8424   return d.getFullYear()
8425     + '/' + zero2(d.getMonth())
8426     + '/' + zero2(d.getDate())
8427     + ' ' + zero2(d.getHours())
8428     + ':' + zero2(d.getMinutes())
8429     + ':' + zero2(d.getSeconds())
8430 }
8431 function DateShort(){
8432   return DateShort0(new Date());
8433 }
8434 function DateLong0(ms){
8435   d = new Date();
8436   d.setTime(ms);
8437   return DateShort0(d)
8438     + '.' + d.getMilliseconds()
8439     + '+' + d.getTimezoneOffset()/60
8440     + ':' + d.getTime() + '.' + d.getMilliseconds()
8441 }
8442 function DateLong(){
8443   return DateLong0(new Date());
8444 }
8445 function GShellMenu(e){
8446   //GLog.innerHTML = "Hello, World! (" + DateLong() + ")"
8447   //showGShellPlane()
8448   ConfigClick();
8449 }
8450 // placements of planes
8451 function GShellResizeX(ev){
8452   //if( document.getElementById("GMenu") != null ){
8453     //GShellIconSetup();
8454     //GMenu.style.left = window.innerWidth - 100
8455     //GMenu.style.top = window.innerHeight - 90 - 200
8456     //console.log("place GMENU "+GMenu.style.left+" "+GMenu.style.top)
8457   //}
8458   //}
8459   GStat.style.width = window.innerWidth
8460   //if( document.getElementById("GPos") != null ){
8461     //GPos.style.width = window.innerWidth
8462     //GPos.style.top = window.innerHeight - 30; //GPos.style.height
8463   //}
8464   //if( document.getElementById("GLog") != null ){
8465     //GLog.style.width = window.innerWidth
8466     //GLog.innerHTML = ""
8467   //}
8468   //if( document.getElementById("GLog") != null ){
8469     //GLog.innerHTML = "Resize: w=" + window.innerWidth +
8470     //"/", h=" + window.innerHeight
8471   //}
8472   showCurElementPosition(ev)
8473 }
8474 function GShellResize(){
8475   GShellResizeX("RESIZE")
8476 }
8477 window.onresize = GShellResize
8478 var prevNode = null
8479 var LogMouseMoveOverElement = false;
8480 function GJSH_OnMouseMove(ev){
8481   if( LogMouseMoveOverElement == false ){
8482     return;
8483   }
8484   x = ev.clientX
8485   y = ev.clientY
8486   d = new Date()
8487   t = d.getTime() / 1000
8488   if( document.elementFromPoint ){
8489     e = document.elementFromPoint(x,y)
8490     if( e != null ){
8491       if( e == prevNode ){
8492         }else{
8493           console.log('Mo-'+t+'('+x+', '+y+') '
8494             +e.nodeType+' '+e.tagName+'#'+e.id)
8495           prevNode = e

```



```

8496     }
8497     }else{
8498         console.log(t+'('+x+', '+y+') no element')
8499     }
8500 }else{
8501     console.log(t+'('+x+', '+y+') no elementFromPoint')
8502 }
8503 }
8504 window.addEventListener('mousemove',GJSH_OnMouseMove);
8505
8506 function GJSH_OnMouseMoveScreen(ev){
8507     x = ev.screenX
8508     y = ev.screenY
8509     d = new Date()
8510     t = d.getTime() / 1000
8511     console.log(t+'('+x+', '+y+') no elementFromPoint')
8512 }
8513 //screen.addEventListener('mousemove',GJSH_OnMouseMoveScreen);
8514
8515 function ScrollToElement(oe,ne){
8516     ne.scrollIntoView()
8517     ny = ne.getBoundingClientRect().top.toFixed(0)
8518     nx = ne.getBoundingClientRect().left.toFixed(0)
8519     //GLog.innerHTML = "["+ny+", "+nx+"]"
8520     //window.scrollTo(0,0)
8521
8522     GTop.style.backgroundColor = "rgba(0,0,0,0)"
8523     GshGrid.style.left = "250px";
8524     GshGrid.style.zindex = 0
8525     if( false ){
8526         oy = oe.getBoundingClientRect().top.toFixed(0)
8527         ox = oe.getBoundingClientRect().left.toFixed(0)
8528         y = e.getBoundingClientRect().top.toFixed(0)
8529         x = e.getBoundingClientRect().left.toFixed(0)
8530         window.scrollTo(x,y)
8531         ny = e.getBoundingClientRect().top.toFixed(0)
8532         nx = e.getBoundingClientRect().left.toFixed(0)
8533         //GLog.innerHTML = "["+oy+", "+ox+"]->["+y+", "+x+"]->["+ny+", "+nx+"]"
8534     }
8535 }
8536 function showGShellPlane(){
8537     if( GShellPlane.style.zindex == 0 ){
8538         GShellPlane.style.zindex = 1000;
8539         GShellPlane.style.left = 30;
8540         GShellPlane.style.height = 320;
8541         GShellPlane.innerHTML = DataLong() + "<br>" +
8542             "-- History --<br>" + MyHistory;
8543     }else{
8544         GShellPlane.style.zindex = 0;
8545         GShellPlane.style.left = 0;
8546         GShellPlane.style.height = 50;
8547         GShellPlane.innerHTML = "";
8548     }
8549 }
8550 var SuppressGJShell = false
8551 function JgshCommand(kevent){
8552     if( SuppressGJShell ){
8553         return
8554     }
8555     key = kevent
8556     keycode = key.code
8557     //GStat.style.width = window.innerWidth
8558     GStat.style.backgroundColor = "rgba(0,0,0,0.4)"
8559
8560     console.log("JSGsh-Key:"+keycode+"(^-^)//")
8561     if( keycode == "Slash" ){
8562         console.log('('+x+', '+y+') ')
8563         e = document.elementFromPoint(x,y)
8564         console.log('('+x+', '+y+') '+e.nodeType+' '+e.tagName+'#'+e.id)
8565     }else
8566     if( keycode == "Digit0" ){ // fold side-bar
8567         // "Zero page"
8568         showGShellPlane();
8569     }else
8570     if( keycode == "Digit1" ){ // fold side-bar
8571         primary.style.width = "94%"
8572         secondary.style.width = "0%"
8573         secondary.style.opacity = 0
8574         GStat.innerHTML = "[Single Column View]"
8575     }else
8576     if( keycode == "Digit2" ){ // unfold side-bar
8577         primary.style.width = "58%"
8578         secondary.style.width = "36%"
8579         secondary.style.opacity = 1
8580         GStat.innerHTML = "[Double Column View]"
8581     }else
8582     if( keycode == "KeyU" ){ // fold/unfold all
8583         html_fold(GshMenuFold);
8584         location.href = "#"+CurElement.id;
8585     }else
8586     if( keycode == "KeyO" || keycode == "ArrowRight" ){ // fold the element
8587         CurElement.open = !CurElement.open;
8588     }else
8589     if( keycode == "ArrowRight" ){ // unfold the element
8590         CurElement.open = true
8591     }else
8592     if( keycode == "ArrowLeft" ){ // unfold the element
8593         CurElement.open = false
8594     }else
8595     if( keycode == "KeyI" ){ // inspect the element
8596         e = CurElement
8597         //GLog.innerHTML =
8598         GJLog.append("Current Element: " + e + "<br>"
8599             + "name="+e.nodeName + ", "
8600             + "id="+e.id + ", "
8601             + "children="+e.childNodes.length + ", "
8602             + "parent="+e.parentNode.id + "<br>" + "
8603             + "text="+e.textContent)
8604         GStat.style.backgroundColor = "rgba(0,0,0,0.8)"
8605         return
8606     }else
8607     if( keycode == "KeyM" ){ // memory the position
8608         MemElement = CurElement
8609     }else
8610     if( keycode == "KeyN" || keycode == "ArrowDown" ){ // next element
8611         e = nextSib(CurElement)
8612         if( e != null ){
8613             setColor(CurElement,"SUMMARY","#fff")
8614             setColor(e,"SUMMARY","#f8") // should be complement ?
8615             oe = CurElement
8616             CurElement = e
8617             //location.href = "#"+e.id;
8618             ScrollToElement(oe,e)
8619         }
8620     }else
8621     if( keycode == "KeyP" || keycode == "ArrowUp" ){ // previous element
8622         oe = CurElement
8623         e = prevSib(CurElement)
8624         if( e != null ){
8625             setColor(CurElement,"SUMMARY","#fff")
8626             setColor(e,"SUMMARY","#f8") // should be complement ?
8627             CurElement = e
8628             //location.href = "#"+e.id;
8629             ScrollToElement(oe,e)
8630         }
8631     }else{
8632         e = document.getElementById("GshBanner")
8633         if( e != null ){
8634             setColor(CurElement,"SUMMARY","#fff")
8635             CurElement = e
8636             ScrollToElement(oe,e)
8637         }else{
8638             e = document.getElementById("primary")
8639             if( e != null ){
8640                 setColor(CurElement,"SUMMARY","#fff")
8641                 CurElement = e
8642                 ScrollToElement(oe,e)
8643             }
8644         }
8645     }else
8646     if( keycode == "KeyR" ){
8647         location.reload()
8648     }else
8649     if( keycode == "KeyJ" ){
8650         GshGrid.style.top = '120px';
8651         GshGrid.innerHTML = '<_>{Down}';
8652     }else
8653     if( keycode == "KeyK" ){
8654         GshGrid.style.top = '0px';
8655         GshGrid.innerHTML = '{^~^}{Up}';
8656     }else
8657     if( keycode == "KeyH" ){
8658         GshGrid.style.left = '0px';
8659         GshGrid.innerHTML = '{^_}{Left}';
8660     }else
8661     if( keycode == "KeyL" ){
8662         //GLog.innerHTML +=
8663         GJLog.append(
8664             'screen'+screen.width+'px'+<br>'+
8665             'window'+window.innerWidth+'px'+<br>'+
8666         )
8667         GshGrid.style.left = (document.documentElement.clientWidth-160).toString(10)+'px';
8668         GshGrid.innerHTML = '{@}{Right}';
8669     }else
8670     if( keycode == "KeyS" ){
8671         html_stop(GshMenuStop,true)
8672     }else

```

```

8673 if( keycode == "KeyF" ){
8674     html_fork()
8675 }else
8676 if( keycode == "KeyC" ){
8677     window.close()
8678 }else
8679 if( keycode == "KeyD" ){
8680     html_digest()
8681 }else
8682 if( keycode == "KeyU" ){
8683     e = document.getElementById('gsh-digest')
8684     if( e != null ){
8685         showDigest(e)
8686     }
8687 }
8688
8689 showCurElementPosition("[+key.codet+] --");
8690 //if( document.getElementById("GPos") != null ){
8691 //GPos.innerHTML += "[+key.codet+] --"
8692 //}
8693 //GShellResizeX("[+key.codet+] --");
8694 }
8695 var initGSKC = false;
8696 function GShell_initKeyCommands(){
8697     if( initGSKC ){ return; } initGSKC = true;
8698
8699     GShellResizeX("[INIT]");
8700     DisplaySize = -- Display;
8701     + 'screen'+screen.width+'px, '+window+'window.innerWidth+'px';
8702
8703     let {text, digest} = getDigest()
8704     //GLog.innerHTML +=
8705     GJLog_append(
8706         "-- GShell: ' + GshVersion.innerHTML + '\n' +
8707         "-- Digest: ' + digest + '\n' +
8708         DisplaySize
8709         //+ "<br>" + "-- LastVisit:<br>" + MyHistory
8710     )
8711     GShellResizeX(null);
8712 }
8713 //GShell_initKeyCommands();
8714
8715 // <a href="https://www.w3.org/TR/WebCryptoAPI/">Web Cryptography API</a>
8716 //Convert a string into an ArrayBuffer
8717 //from https://developers.google.com/web/updates/2012/06/How-to-convert-ArrayBuffer-to-and-from-String
8718 function str2buf(str) {
8719     const buf = new ArrayBuffer(str.length);
8720     const bufView = new Uint8Array(buf);
8721     for (let i = 0, strLen = str.length; i < strLen; i++) {
8722         bufView[i] = str.charCodeAt(i);
8723     }
8724     return buf;
8725 }
8726
8727 function importPrivateKey(pem) {
8728     const binaryDerString = window.atob(pemContents);
8729     const binaryDer = str2ab(binaryDerString);
8730     return window.crypto.subtle.importKey(
8731         "pkcs8",
8732         binaryDer,
8733         {
8734             name: "RSA-PSS",
8735             modulusLength: 2048,
8736             publicExponent: new Uint8Array([1, 0, 1]),
8737             hash: "SHA-256",
8738         },
8739         true,
8740         ["sign"]
8741     );
8742 }
8743 //importPrivateKey(ppem)
8744
8745 //key = {}
8746 //buf = "abc"
8747 //enc = "xyzxxxxx"; //crypto.publicEncrypt(key,buf)
8748 //b64 = btoa(enc)
8749 //dec = atob(b64)
8750 //GLog.innerHTML = "enc:" + b64 + ", dec:" + dec
8751 </script>
8752
8753 //<!-- ===== Work { ===== -->
8754 <span id="PackmonGo_WorkCodeSpan">
8755 /*
8756 <details id="PackmonGo_Section"><summary id="PackmonGo_Summary">PackmonGo</summary>
8757 <!-- ===== PackmonGo // 2020-1025 SatoxITS { -->
8758 <h2>PackmonGo</h2>
8759 <div id="PackmonGo_1" class="PackmonGo">
8760 <canvas id="PackmonGo_1_Canvas" class="PackmonGo_Canvas" width="150px" height="150px" draggable="true"></canvas>
8761 </div>
8762 <div id="PackmonGo_2" class="PackmonGo">
8763 <canvas id="PackmonGo_2_Canvas" class="PackmonGo_Canvas" width="150px" height="150px" draggable="true"></canvas>
8764 </div>
8765 <div id="PackmonGo_3" class="PackmonGo">
8766 <canvas id="PackmonGo_3_Canvas" class="PackmonGo_Canvas" width="150px" height="150px" draggable="true"></canvas>
8767 </div>
8768 <div id="PackmonGo_4" class="PackmonGo">
8769 <canvas id="PackmonGo_4_Canvas" class="PackmonGo_Canvas" width="400px" height="400px" draggable="true"></canvas>
8770 </div>
8771 <style>
8772 .PackmonGo {
8773     position:fixed !important;
8774     background-color:rgba(0,0,0,0.0);
8775 }
8776 .PackmonGo_Canvas {
8777     background-color:rgba(0,0,0,0.0);
8778 }
8779 </style>
8780 <script>
8781 var stopPackmonFlag = false;
8782 var PackmonInit = false;
8783 function stopPackmon(){
8784     stopPackmonFlag = !stopPackmonFlag;
8785 }
8786 function spawnPackmonGo(){
8787     if( PackmonInit == false ){
8788         pgw = 100;
8789         pgh = 100;
8790         pgo = 0.5;
8791
8792         ctx = PackmonGo_1_Canvas.getContext('2d');
8793         ctx.fillStyle = "rgba(255,255,0,'+pgov')";
8794         ctx.fillRect(0,0,pgw,pgh);
8795         base = PackmonGo_1;
8796         gsh.appendChild(base);
8797         base.style.zIndex = 1000;
8798         base.style.position = "fixed";
8799         base.style.left = 0 + 'px';
8800         base.style.top = 0 + 'px';
8801         base.xinc = 4;
8802         base.yinc = 4;
8803         base.scrx = 0;
8804         base.scry = 0;
8805         base.pgw = 100;
8806         base.pgh = 100;
8807         movePWindow(PackmonGo_1);
8808
8809         ctx = PackmonGo_2_Canvas.getContext('2d');
8810         ctx.fillStyle = "rgba(127,255,127,'+pgov')";
8811         ctx.fillRect(0,0,pgw,pgh);
8812         base = PackmonGo_2;
8813         gsh.appendChild(base);
8814         base.style.zIndex = 1001;
8815         base.style.position = "fixed";
8816         base.style.left = 200 + 'px';
8817         base.style.top = 0 + 'px';
8818         base.xinc = 4;
8819         base.yinc = 4;
8820         base.scrx = 0;
8821         base.scry = 0;
8822         base.pgw = 100;
8823         base.pgh = 100;
8824         movePWindow(PackmonGo_2);
8825
8826         ctx = PackmonGo_3_Canvas.getContext('2d');
8827         pgo = 0.3;
8828         ctx.fillStyle = "rgba(64,64,255,'+pgov')";
8829         ctx.fillRect(0,0,pgw,pgh);
8830         base = PackmonGo_3;
8831         gsh.appendChild(base);
8832         base.style.zIndex = 1002;
8833         base.style.position = "fixed";
8834         base.style.left = 200 + 'px';
8835         base.style.top = 0 + 'px';
8836         base.xinc = 4;
8837         base.yinc = 4;
8838         base.scrx = 0;
8839         base.scry = 0;
8840         base.soff = 0;
8841         base.pgw = 100;
8842         base.pgh = 100;
8843         movePWindow(PackmonGo_3);
8844
8845         ctx = PackmonGo_4_Canvas.getContext('2d');
8846         pgo = 400;
8847         ctx.beginPath();
8848         ctx.strokeStyle = "rgba(0,0,0,0)";
8849         ctx.arc(200,200,200,0,Math.PI*2,true);

```

```

8850     ctx.stroke();
8851     pgo = 0.4;
8852     ctx.fillStyle = 'rgba(255,31,32, 'pgo+')';
8853     ctx.fill();
8854     base = PackmonGo_4;
8855     gsh.appendChild(base);
8856     base.style.zindex = 1002;
8857     base.style.position = "fixed";
8858     base.style.left = 200 + 'px';
8859     base.style.top = 0 + 'px';
8860     base.xinc = 4;
8861     base.yinc = 4;
8862     base.scrx = 0;
8863     base.scry = 0;
8864     base.soft = 5000;
8865     base.pgw = pgw;
8866     base.pgh = pgh;
8867     movePGscreen(PackmonGo_4);
8868 }
8869 function movePGwindow(base){
8870     if( stopPackmonFlag ){
8871         return;
8872     }
8873     x = parseInt(base.style.left);
8874     y = parseInt(base.style.top);
8875     w = window.innerWidth;
8876     h = window.innerHeight;
8877     if( x < 0 || w-base.pgw < x ){
8878         base.xinc = -base.xinc;
8879     }
8880     x += base.xinc;
8881     if( y < 0 || h-base.pgh < y ){
8882         //yinc = -yinc;
8883         base.yinc = -base.yinc;
8884     }
8885     y += base.yinc;
8886     base.style.left = x + 'px';
8887     base.style.top = y + 'px';
8888     //console.log('PG x'+x+',y'+y);
8889     //window.setTimeout(movePG,10);
8890 }
8891 function movePGscreen(base){
8892     if( stopPackmonFlag ){
8893         return;
8894     }
8895     sw = screen.width;
8896     sh = screen.height;
8897     d = new Date();
8898     s = d.getTime(); // milli-seconds
8899     sx = ((s+base.soft) % 10000) * (screen.width / 10000);
8900     sy = ((s+base.soft) % 5000) * (screen.height / 5000);
8901     x = sx - window.screenX;
8902     y = sy - window.screenY;
8903     base.style.left = x + 'px';
8904     base.style.top = y + 'px';
8905 }
8906 function movePGscreenCircle(base){
8907     if( stopPackmonFlag ){
8908         return;
8909     }
8910     sw = screen.width;
8911     sh = screen.height;
8912     pgw = base.pgw;
8913     pgh = base.pgh;
8914     d = new Date();
8915     s = d.getTime(); // milli-seconds
8916     ds = s + base.soft; // delay
8917     vsw = pgw + sw + pgw;
8918     vsh = pgh + sh + pgh;
8919     sx = -pgw + vsw * ((ds % 10000)/10000);
8920     sy = -pgh + vsh * ((ds % 10000)/10000);
8921     x = sx - window.screenX;
8922     y = sy - window.screenY;
8923     base.style.left = x + 'px';
8924     base.style.top = y + 'px';
8925 }
8926 if( PackmonInit == false ){
8927     //window.setTimeout(movePG,mm300);
8928     window.setInterval(movePGwindow,10,PackmonGo_1);
8929     window.setInterval(movePGwindow,10,PackmonGo_2);
8930     window.setInterval(movePGscreen,10,PackmonGo_3);
8931     window.setInterval(movePGscreen,10,PackmonGo_4);
8932     window.setInterval(movePGscreenCircle,10,PackmonGo_4);
8933     window.addEventListener('click',stopPackMon);
8934     PackmonInit = true;
8935     stopPackmonFlag = false;
8936 }
8937 }
8938 function PackmonGo_Setup(e){
8939     stopPackMon();
8940     spawnPackmonGo();
8941     if( e != null ){
8942         e.stopPropagation();
8943     }
8944 }
8945 PackmonGo_Summary.addEventListener('click',PackmonGo_Setup);
8946 if( PackmonGo_Section.open == true ){
8947     PackmonGo_Setup();
8948 }
8949 </script>
8950
8951 <input id="PackmonGo_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
8952 <input id="PackmonGo_WorkCodeViewSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
8953 <input id="PackmonGo_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
8954 <span id="PackmonGo_WorkCodeView"></span>
8955 <script id="PackmonGo_WorkScript">
8956 function PackmonGo_openWorkCodeView(){
8957     function PackmonGo_showWorkCode(){
8958         showHtmlCode(PackmonGo_WorkCodeView,PackmonGo_WorkCodeSpan);
8959     }
8960     PackmonGo_WorkCodeViewOpen.addEventListener('click',PackmonGo_showWorkCode);
8961 }
8962 PackmonGo_openWorkCodeView(); // should be invoked by an event
8963 </script>
8964 </details>
8965 <!-- Template_WorkCodeSpan -->
8966 *//</span>
8967 //<!-- ===== Work } ===== -->
8968
8969
8970
8971 //<!-- ===== Work { ===== -->
8972 //<span id="SightGlass_WorkCodeSpan">
8973 /*
8974 <details id="SightGlass"><summary>SightGlass</summary>
8975 <!-- ===== SightGlass // 2020-1023 SatoxITS { -->
8976 <h2>SightGlass</h2>
8977 <details><summary>meter</summary>
8978 <div id="SightGlass_1_BPosition" class="SightGlass_TextData">(0000, 0000)(0000 x 0000) </div>
8979 <div id="SightGlass_1_BGScroffset" class="SightGlass_TextData">(0000, 0000) </div>
8980 <div id="SightGlass_1_GPosition" class="SightGlass_TextData">(0000, 0000)(0000 x 0000) </div>
8981 <div id="SightGlass_1_BPosition" class="SightGlass_TextData">(0000, 0000)(0000 x 0000) </div>
8982 <div id="SightGlass_1_Wheel" class="SightGlass_TextData">Wheel</div>
8983 </details>
8984 <div id="SightGlass_1_Window" class="SightGlass_Window" draggable="true"></div>
8985 <style>
8986 .SightGlass_TextData {
8987     color:#000;
8988     font-size:10pt;
8989 }
8990 .SightGlass_Window {
8991     xxzoom:2;
8992     resize:both;
8993     width:600px;
8994     height:320px;
8995     background-color:rgba(200,200,200,0.5);
8996     background-size:200%;
8997     xbackground-size:1280px 720px;
8998     background-image:url(WD-WallPaper03.png);
8999 }
9000 xbody {
9001     scroll-behavior:smooth;
9002 }
9003 </style>
9004 </script>
9005 //
9006 // https://stackoverflow.com/questions/4319487/detecting-if-the-browser-window-is-moved-with-javascript
9007 var SGScrInitX = 0;
9008 var SGScrInitY = 0;
9009 var SG_initX = 0;
9010 var SG_initY = 0;
9011 function SG_adjust(){
9012     xy = window.screenX + ' ' + window.screenY;
9013     wh = window.innerWidth + ' ' + window.innerHeight;
9014     xywh = 'xy(' + xy + ') wh(' + wh + ')';
9015     if( SightGlass.open) SightGlass_1_BPosition.innerHTML = 'Browser: ' + xywh;
9016 }
9017 sg = SightGlass_1_Window;
9018 sgr = sg.getBoundingClientRect();
9019 xy = sgr.left.toFixed(0) + ' ' + sgr.top.toFixed(0);
9020 wh = sgr.width + ' ' + sgr.height;
9021 xywh = 'xy(' + xy + ') wh(' + wh + ')';
9022 if( SightGlass.open) SightGlass_1_GPosition.innerHTML = 'SiGlass: ' + xywh;
9023
9024 //SightGlass_1_Window.style.backgroundColor = sgr.left+'px ' + sgr.top+'px';
9025 if( SG_initX == 0 ){
9026     SGScrInitX = window.screenX;

```

```

9027     SGScrInitY = window.screenY;
9028     //SightClass_1_Window.style.backgroundColor = '200%';
9029     //SightClass_1_Window.style.backgroundImage = 'url("WD-WallPaper03.png")';
9030     SG_initX = sgr.left;
9031     SG_initY = sgr.top;
9032     }
9033     dx = SG_initX - sgr.left;
9034     dy = SG_initY - sgr.top;
9035     }
9036     dsx = SGScrInitX - window.screenX;
9037     dsy = SGScrInitY - window.screenY;
9038     scroff = 'Screen: ' + 'sx'+dsx + ',sy'+dsy;
9039     if( SightClass.open) SightClass_1_BGScroffset.innerHTML = scroff;
9040     dx += dsx;
9041     dy += dsy;
9042     }
9043     bpos = dx+'px ' + dy+'px';
9044     SightClass_1_Window.style.backgroundColor = bpos;
9045     if( SightClass.open) SightClass_1_BGPosition.innerHTML = 'BGround:'
9046     + SightClass_1_Window.style.backgroundColor;
9047     }
9048     var wheels = 0;
9049     function SG_wheel(e){
9050     wheels += 1;
9051     SightClass_1_Wheel.innerHTML = 'Mouse Wheel: ' + wheels + ' #' + e.target.id;
9052     if( e.target.id == 'SightClass_1_Window' ){
9053     e.preventDefault();
9054     }
9055     }
9056     function SG_adjust1(){
9057     SG_adjust();
9058     //sampling = 0;
9059     sampling = 20;
9060     //sampling = 200;
9061     window.setTimeout(SG_adjust1,sampling);
9062     }
9063     function SightClass_Setup(){
9064     SG_adjust();
9065     window.addEventListener('resize',SG_adjust);
9066     window.addEventListener('mousemove',SG_adjust);
9067     window.addEventListener('scroll',SG_adjust);
9068     window.addEventListener('wheel',SG_wheel,{passive:false});
9069     //window.moveTo(0,0);
9070     }
9071     // if focused
9072     //window.setInterval(SG_adjust,1);
9073     window.setTimeout(SG_adjust1,1);
9074     }
9075     // https://stackoverflow.com/questions/45225798/how-do-i-subscribe-to-a-window-move-event-in-the-atom-editor
9076     function Electron_Setup(){
9077     const { BrowserWindow } = require('electron');
9078     const currentWindow = BrowserWindow.getFocusedWindow();
9079     //currentWindow.on('move',function(){ SG_adjust(); });
9080     currentWindow.addEventListener('move',SG_adjust);
9081     }
9082     </script>
9083     <input id="SightClass_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
9084     <input id="SightClass_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
9085     <input id="SightClass_WorkCodesSignature" class="HtmlCodeViewButton" type="button" value="Signature">
9086     <span id="SightClass_WorkCodeView"></span>
9087     <script id="SightClass_WorkScript">
9088     function SightClass_openWorkCodeView(){
9089     function SightClass_showWorkCode(){
9090     showHtmlCode(SightClass_WorkCodeView,SightClass_WorkCodeSpan);
9091     }
9092     SightClass_WorkCodeViewOpen.addEventListener('click',SightClass_showWorkCode);
9093     }
9094     SightClass_openWorkCodeView(); // should be invoked by an event
9095     </script>
9096     </details>
9097     <!-- SightClass_WorkCodeSpan -->
9098     *! /</span>
9099     /*!-- ===== Work } ===== -->
9100     }
9101     }
9102     }
9103     }
9104     /*
9105     <!-- ----- GJConsole BEGIN { ----- -->
9106     <span id="gjc" data-title="GJConsole" data-author="sato@its-more.jp">
9107     <details><summary>GJ Console</summary>
9108     <p>
9109     <span id="GJE_RootNode0"></span>
9110     <span id="GJCI_Container"></span>
9111     </p>
9112     <style id="GJConsoleStyle">
9113     .GJConsole {
9114     z-index:1000;
9115     width:400; height:200px;
9116     margin:2px;
9117     color:#fff; background-color:#66a;
9118     font-size:12px; font-family:monospace,Courier New;
9119     }
9120     </style>
9121     }
9122     <script id="GJConsoleScript" class="GJConsole">
9123     var PS1 = "% "
9124     function GJC_KeyDown(keyevent){
9125     key = keyevent.code
9126     if( key == "Enter" ){
9127     GJC_Command(this)
9128     this.value += "\n" + PS1 // prompt
9129     }else
9130     if( key == "Escape"){
9131     SuppressGJShell = false
9132     GshMenu.focus() // should be previous focus
9133     }
9134     }
9135     var GJC_SessionId
9136     function GJC_SetSessionId(){
9137     var xd = new Date()
9138     GJC_SessionId = xd.getTime() / 1000
9139     }
9140     GJC_SetSessionId()
9141     function GJC_Memory(mem,args,text){
9142     argv = args.split(' ')
9143     cmd = argv[0]
9144     argv.shift()
9145     args = argv.join(' ')
9146     ret = ""
9147     }
9148     if( cmd == 'clear' ){
9149     Permanent.setItem(mem,'')
9150     }else
9151     if( cmd == 'read' ){
9152     ret = Permanent.getItem(mem)
9153     }else
9154     if( cmd == 'save' ){
9155     val = Permanent.getItem(mem)
9156     if( val == null ){ val = "" }
9157     d = new Date()
9158     val += d.getTime()/1000+ " +GJC_SessionId+ " +document.URL+ " +args+"\n"
9159     val += text.value
9160     Permanent.setItem(mem,val)
9161     }else
9162     if( cmd == 'write' ){
9163     val = Permanent.getItem(mem)
9164     if( val == null ){ val = "" }
9165     d = new Date()
9166     val += d.getTime()/1000+ " +GJC_SessionId+ " +document.URL+ " +args+"\n"
9167     Permanent.setItem(mem,val)
9168     }else{
9169     ret = "Commands: write | read | save | clear"
9170     }
9171     return ret
9172     }
9173     // -- 2020-09-14 added TableEditor
9174     var GJE_CurElement = null; //GJE_RootNode
9175     GJE_NodeSaved = null
9176     GJE_TableNo = 1
9177     function GJE_StyleKeyCommand(kev){
9178     keycode = kev.code
9179     console.log("GJE-Key: "+keycode)
9180     if( keycode == "Escape" ){
9181     GJE_SetStyle(this);
9182     }
9183     kev.stopPropagation()
9184     // https://developer.mozilla.org/en-US/docs/Web/API/Event/stopPropagation
9185     }
9186     var GJE_CommandMode = false
9187     function GJE_TableKeyCommand(kev,tab){
9188     wasCmdMode = GJE_CommandMode
9189     key = kev.code
9190     if( key == "Escape" ){
9191     console.log("To command mode: "+tab.nodeName+"#" +tab.id)
9192     //tab.setAttribute('contenteditable','false')
9193     tab.style.caretColor = "blue"
9194     GJE_CommandMode = true
9195     }else
9196     if( key == "KeyA" ){
9197     tab.style.caretColor = "red"
9198     GJE_CommandMode = false
9199     }else
9200     if( key == "KeyI" ){
9201     tab.style.caretColor = "red"
9202     GJE_CommandMode = false
9203     }else

```

```

9204     if( key == "KeyO" ){
9205         tab.style.caretColor = "red"
9206         GJE_CommandMode = false
9207     }else
9208     if( key == "KeyJ" ){
9209         console.log("ROW-DOWN")
9210     }else
9211     if( key == "KeyK" ){
9212         console.log("ROW-UP")
9213     }else
9214     if( key == "Keyw" ){
9215         console.log("COL-FORW")
9216     }else
9217     if( key == "Keyb" ){
9218         console.log("COL-BACK")
9219     }
9220
9221     kev.stopPropagation()
9222     if( wasCmdMode ){
9223         kev.preventDefault()
9224     }
9225 }
9226 function GJE_DragEvent(ev,elem){
9227     x = ev.clientX
9228     y = ev.clientY
9229     console.log("Dragged: "+this.nodeName+'#'+this.id+' x='+x+' y='+y)
9230 }
9231 // https://developer.mozilla.org/en-US/docs/Web/API/DragEvent
9232 // https://www.w3.org/TR/uevents/#events-mouseevents
9233 function GJE_DropEvent(ev,elem){
9234     x = ev.clientX
9235     y = ev.clientY
9236     this.style.x = x
9237     this.style.y = y
9238     this.style.position = 'absolute' // 'fixed'
9239     this.parentNode = gsh // just for test
9240     console.log("Dropped: "+this.nodeName+'#'+this.id+' x='+x+' y='+y
9241         +' parent='+this.parentNode.id)
9242 }
9243 function GJE_SetTableStyle(ev){
9244     this.innerHTML = this.value; // sync. for external representation?
9245     if(false){
9246         stid = this.parentNode.id+this.id
9247         // and remove "span" at the end
9248         e = document.getElementById(stid)
9249         //alert('SetTableStyle #'+e.id+'\n'+this.value)
9250         if( e != null ){
9251             e.innerHTML = this.value
9252         }else{
9253             console.log('Style Not found: '+stid)
9254         }
9255         //alert('event StopPropagation: '+ev)
9256     }
9257 }
9258 function setCSSofClass(cclass,cstyle){
9259     const ss = document.styleSheets[3]; // 0, 1, 2, 3, ... ?
9260     rlen = ss.cssRules.length;
9261     let tabrule = null;
9262     rulex = -1
9263
9264     // should skip white space at the top of cstyle
9265     sel = cstyle.charAt(0);
9266     selector = sel+cclass;
9267     console.log("-- search style rule for '+selector')
9268
9269     for(let i = 0; i < rlen; i++){
9270         cr = ss.cssRules[i];
9271         console.log('CSS rule ['+'+i+'+'+rlen+''] '+cr.selectorText);
9272         if( cr.selectorText === selector ){ // css class selector
9273             tabrule = ss.cssRules[i];
9274             console.log('CSS rule found for:['+'+i+'+'+rlen+''] '+selector);
9275             ss.deleteRule(i);
9276             //rlen = ss.cssRules.length;
9277             rulex = i
9278             // should search and replace the property here
9279         }
9280     }
9281     // https://developer.mozilla.org/en-US/docs/Web/API/CSSStyleSheet/insertRule
9282     if( tabrule == null ){
9283         console.log('CSS rule NOT found for:['+rlen+''] '+selector);
9284         ss.insertRule(cstyle,rlen);
9285         ss.insertRule(cstyle,0); // override by 0?
9286         console.log('CSS rule inserted:['+(rlen+1)+'']\n'+cstyle);
9287     }else{
9288         ss.insertRule(cstyle,rlen);
9289         ss.insertRule(cstyle,0);
9290         console.log('CSS rule replaced:['+(rlen+1)+'']\n'+cstyle);
9291     }
9292 }
9293 function GJE_SetStyle(te){
9294     console.log('Apply the style to:'+te.id+'\n');
9295     console.log('Apply the style to:'+te.parentNode.id+'\n');
9296     console.log('Apply the style to:'+te.parentNode.class+'\n');
9297     cclass = te.parentNode.class;
9298     setCSSofClass(cclass,te.value); // should get selector part from
9299     // selector { rules }
9300
9301     if(false){
9302         //console.log('Apply the style:')
9303         //stid = this.parentNode.id+this.id+"
9304         //stid = this.id+"style"
9305         css = te.value
9306         stid = te.parentNode.id+"style"
9307         e = document.getElementById(stid)
9308         if( e != null ){
9309             //console.log('Apply the style:'+e.id+'\n'+te.value);
9310             console.log('Apply the style:'+e.id+'\n'+css);
9311             // e.innerHTML = css; //te.value;
9312             //ncss = e.sheet;
9313             //ncss.insertRule(te.value,ncss.cssRules.length);
9314         }else{
9315             console.log('No element to Apply the style: '+stid)
9316         }
9317         tblid = te.parentNode.id+"table";
9318         e = document.getElementById(tblid);
9319         if( e != null ){
9320             //e.setAttribute('style',css);
9321             e.setProperty('style',css,'!important');
9322         }
9323     }
9324 }
9325 function makeTable(argv){
9326     //tid = ''
9327     //cwe = GJE_CurElement
9328     cwe = GJCl_Container;
9329     //cwd = GJFactory;
9330     tid = 'table_' + GJE_TableNo
9331
9332     nt = new Text('\n')
9333     cwe.appendChild(nt)
9334
9335     ne = document.createElement('span'); // the container
9336     cwe.appendChild(ne)
9337     ne.id = tid + "-span"
9338     ne.setAttribute('contenteditable',true)
9339
9340     htspan = document.createElement('span'); // html part
9341     //htspan.id = tid + "-html"
9342     //ne.innerHTML = '\n'
9343     nt = new Text('\n')
9344     ne.appendChild(nt)
9345     ne.appendChild(htspan)
9346
9347     htspan.id = tid
9348     htspan.setAttribute('class',tid)
9349
9350     ne.setAttribute('draggable','true')
9351     ne.addEventListener('drag',GJE_DragEvent);
9352     ne.addEventListener('dragend',GJE_DropEvent);
9353
9354     var col = 3
9355     var row = 2
9356     if( argv[0] != null ){
9357         col = argv[0]
9358         argv.shift()
9359     }
9360     if( argv[0] != null ){
9361         row = argv[0]
9362         argv.shift()
9363     }
9364
9365     //ne.setAttribute('class',tid)
9366     ht = '\n'
9367     //ht += '<'+table' + 'id="'+tid+'"' + ' class="'+tid+'"'
9368     ht += '<'+table'
9369         + ' onkeydown="GJE_TableKeyCommand(event,this)"'
9370         + ' ondrag="GJE_DragEvent(event,this)"\n'
9371         + ' ondragend="GJE_DropEvent(event,this)"\n'
9372         + ' draggable="true"\n'
9373         + ' contenteditable="true"'
9374         + '>\n'
9375     ht += '<'+tbody>\n';
9376     for( r = 0; r < row; r++){
9377         ht += '<'+tr>\n'
9378         for( c = 0; c < col; c++){
9379             ht += '<'+td>'
9380             ht += "ABCDEFHIJKLMNOPQRSTUVWXYZ".charAt(c) + r

```

```

9381         ht += "<+>/td>\n"
9382     }
9383     ht += "<+>/tr>\n"
9384 }
9385 ht += '<+>/tbody>\n';
9386 ht += '<+>/table>\n';
9387 htspan.innerHTML = ht;
9388 nt = new Text('\n');
9389 ne.appendChild(nt);
9390
9391 st = '#'+tid+' *{\n' // # for instanse specific
9392 + ' '+border:1px solid #aaa;\n'
9393 + ' '+background-color:#efe;\n'
9394 + ' '+color:#222;\n'
9395 + ' '+font-size:#14pt !important;\n'
9396 + ' '+font-family:monospace,Courier New !important;\n'
9397 + '}' /*+ hit ESC to apply +*/\n
9398
9399 // wish script to be included
9400 //n] = document.createElement('script')
9401 //ne.appendChild(n)
9402 //ne.innerHTML = 'function SetStyle(e){'
9403
9404 // selector seems lost in dynamic style appending
9405 if(false){
9406     ns = document.createElement('style')
9407     ne.appendChild(ns)
9408     ns.id = tid + '.style'
9409     ns.innerHTML = '\n'+st
9410     nt = new Text('\n')
9411     ne.appendChild(nt)
9412 }
9413 setCSSofClass(tid,st); // should be in JavaScript script?
9414
9415 nx = document.createElement('textarea')
9416 ne.appendChild(nx)
9417 nx.id = tid + '-style_def'
9418 nx.setAttribute('class','GJ_StyleEditor')
9419 nx.spellcheck = false
9420 nx.cols = 60
9421 nx.rows = 10
9422 nx.innerHTML = '\n'+st
9423 nx.addEventListener('change',GJE_SetTableStyle);
9424 nx.addEventListener('keydown',GJE_StyleKeyCommand);
9425 //nx.addEventListener('click',GJE_SetTableStyle);
9426
9427 nt = new Text('\n')
9428 cwe.appendChild(nt)
9429
9430 GJE_TableNo += 1
9431 return 'created TABLE id="'+tid+'"'
9432 }
9433 function GJE_NodeEdit(argv){
9434     cwe = GJE_CurElement
9435     cmd = argv[0]
9436     argv.shift()
9437     args = argv.join(' ')
9438     ret = ""
9439
9440     if( cmd == '.u' || cmd == '.un' || cmd == 'undo' ){
9441         if( GJE_NodeSaved != null ){
9442             xn = GJE_RootNode
9443             GJE_RootNode = GJE_NodeSaved
9444             GJE_NodeSaved = xn
9445             ret = '-- did undo'
9446         }else{
9447             ret = '-- could not undo'
9448         }
9449         return ret
9450     }
9451     GJE_NodeSaved = GJE_RootNode.cloneNode()
9452     if( cmd == '.c' || cmd == '.cd' || cmd == 'cd' ){
9453         if( argv[0] == null ){
9454             ne = GJE_RootNode
9455         }else{
9456             if( argv[0] == '..' ){
9457                 ne = cwe.parentNode
9458             }else{
9459                 ne = document.getElementById(argv[0])
9460             }
9461         }
9462         if( ne != null ){
9463             GJE_CurElement = ne
9464             ret = "-- current node: " + ne.id
9465         }else{
9466             ret = "-- not found: " + argv[0]
9467         }
9468     }else{
9469         if( cmd == '.mkt' || cmd == '.mktable' ){
9470             makeTable(argv)
9471         }else{
9472             if( cmd == '.m' || cmd == '.mk' || cmd == 'mk' ){
9473                 ne = document.createElement(argv[0])
9474                 //ne.id = argv[0]
9475                 ret = "-- created " + ne + " under " + cwe.tagName + "#" + cwe.id
9476                 cwe.appendChild(ne)
9477                 if( cmd == '.m' || cmd == '.mk' ){
9478                     GJE_CurElement = ne
9479                 }
9480             }else{
9481                 if( cmd == '.n' || cmd == '.nm' || cmd == 'nm' ){
9482                     cwe.id = argv[0]
9483                 }else{
9484                     if( cmd == '.r' || cmd == '.rm' || cmd == 'rm' ){
9485                         s = argv.join(' ')
9486                         cwe.innerHTML = s
9487                     }else{
9488                         if( cmd == '.a' || cmd == '.sa' || cmd == 'sa' ){
9489                             cwe.setAttribute(argv[0],argv[1])
9490                         }else{
9491                             if( cmd == '.l' ){
9492                                 }else{
9493                                     if( cmd == '.i' || cmd == '.ih' || cmd == 'ih' ){
9494                                         ret = cwe.innerHTML
9495                                     }else{
9496                                         if( cmd == '.p' || cmd == '.pw' || cmd == 'pw' ){
9497                                             ret = cwe.nodeType + " " + cwe.tagName + " " + cwe.id
9498                                             for( we = cwe.parentNode; we != null; ){
9499                                                 ret += "\n" + " " + we.nodeType + " " + we.tagName + " " + we.id
9500                                             }
9501                                             we = we.parentNode
9502                                         }
9503                                     }else{
9504                                         {
9505                                             ret = "Command: mk | rm \n"
9506                                             ret += " pw -- print current node\n"
9507                                             ret += " mk type -- make node with name and type\n"
9508                                             ret += " rm name -- set the id #name of current node\n"
9509                                             ret += " rm name -- remove named node\n"
9510                                             ret += " cd name -- change current node\n"
9511                                         }
9512                                     }
9513                                 }
9514                             }
9515                             function GJC_Command(text){
9516                                 lines = text.value.split('\n')
9517                                 line = lines[lines.length-1]
9518                                 argv = line.split(' ')
9519                                 text.value = '\n'
9520                                 if( argv[0] == '%' ){ argv.shift() }
9521                                 args0 = argv.join(' ')
9522                                 cmd = argv[0]
9523                                 argv.shift()
9524                                 args = argv.join(' ')
9525
9526                                 if( cmd == 'nolog' ){
9527                                     StopConsoleLog = true
9528                                 }else{
9529                                     if( cmd == 'new' ){
9530                                         if( argv[0] == 'table' ){
9531                                             argv.shift()
9532                                             console.log('argv='+argv)
9533                                             text.value += makeTable(argv)
9534                                         }else{
9535                                             if( argv[0] == 'console' ){
9536                                                 text.value += GJ_NewConsole('GJ_Console')
9537                                             }else{
9538                                                 text.value += '-- new { console | table }'
9539                                             }
9540                                         }
9541                                     }else{
9542                                         if( cmd == 'strip' ){
9543                                             //text.value += GJF_StripeClass()
9544                                         }else{
9545                                             if( cmd == 'css' ){
9546                                                 sel = '#table_1'
9547                                                 if( argv[0] == '0' ){
9548                                                     rule1 = sel+'(color:#00 !important; background-color:#fff !important);';
9549                                                     else{
9550                                                         rule1 = sel+'(color:#f00 !important; background-color:#eef !important);';
9551                                                         document.styleSheets[3].deleteRule(0);
9552                                                         document.styleSheets[3].insertRule(rule1,0);
9553                                                         text.value += 'CSS rule added: '+rule1
9554                                                     }
9555                                                 }else{
9556                                                     if( cmd == 'print' ){
9557                                                         e = null;
9558                                                         if( e == null ){
9559                                                             if( e == null ){
9560                                                                 e = document.getElementById('GJFactory_0')
9561                                                             }
9562                                                         }
9563                                                     }
9564                                                 }
9565                                             }
9566                                         }
9567                                     }
9568                                 }
9569                             }
9570                         }
9571                     }
9572                 }
9573             }
9574         }
9575     }
9576 }

```

```

9558     }
9559     if( e == null ){
9560         e = document.getElementById('GJFactory_1')
9561     }
9562     if( argv[0] != null ){
9563         id = argv[0]
9564         if( id == 'e' ){
9565             //e = document.getElementById('GJE_RootNode');
9566         }else{
9567             e = document.getElementById(id)
9568         }
9569         if( e != null ){
9570             text.value += e.outerHTML
9571         }else{
9572             text.value += "Not found: " + id
9573         }
9574     }else{
9575         text.value += GJE_RootNode.outerHTML
9576         //text.value += e.innerHTML
9577     }
9578 }else{
9579     if( cmd == 'destroy' ){
9580         text.value += GJFactory_Destroy()
9581     }else{
9582         if( cmd == 'save' ){
9583             e = document.getElementById('GJFactory')
9584             Permanent.setItem('GJFactory-1',e.innerHTML)
9585             text.value += "-- Saved GJFactory"
9586         }else{
9587             if( cmd == 'load' ){
9588                 gjf = Permanent.getItem('GJFactory-1')
9589                 e = document.getElementById('GJFactory')
9590                 e.innerHTML = gjf
9591                 // must restore EventListener
9592                 text.value += "-- EventListener was not restored"
9593             }else{
9594                 if( cmd.charAt(0) == '.' ){
9595                     argv0 = argv0.split(' ')
9596                     text.value += GJE_NodeEdit(argv0)
9597                 }else{
9598                     if( cmd == 'cont' ){
9599                         bannerIsStopping = false
9600                         GshMenuStop.innerHTML = "Stop"
9601                     }else{
9602                         if( cmd == 'date' ){
9603                             text.value += DateLong()
9604                         }else{
9605                             if( cmd == 'echo' ){
9606                                 text.value += args
9607                             }else{
9608                                 if( cmd == 'fork' ){
9609                                     html_fork()
9610                                 }else{
9611                                     if( cmd == 'last' ){
9612                                         text.value += MyHistory
9613                                         //h = document.createElement("span")
9614                                         //h.innerHTML = MyHistory
9615                                         //text.value += h.innerHTML
9616                                         //tx = MyHistory.replace("\n", "")
9617                                         //text.value += tx.replace("<"+"br>","\n") + "xxxx<"+"br>yyyy"
9618                                     }else{
9619                                         if( cmd == 'ne' ){
9620                                             text.value += GJE_NodeEdit(argv)
9621                                         }else{
9622                                             if( cmd == 'reload' ){
9623                                                 location.reload()
9624                                             }else{
9625                                                 if( cmd == 'mem' ){
9626                                                     text.value += GJC_Memory('GJC_Storage',args,text)
9627                                                 }else{
9628                                                     if( cmd == 'stop' ){
9629                                                         bannerIsStopping = true
9630                                                         GshMenuStop.innerHTML = "Start"
9631                                                     }else{
9632                                                         if( cmd == 'who' ){
9633                                                             text.value += "SessionId="+GJC_SessionId+" "+document.URL
9634                                                         }else{
9635                                                             if( cmd == 'wall' ){
9636                                                                 text.value += GJC_Memory('GJC_Wall','write',text)
9637                                                             }else{
9638                                                                 {
9639                                                                     text.value += "Commands: help | echo | date | last \n"
9640                                                                     + '          new | save | load | mem \n'
9641                                                                     + '          who | wall | fork | nife'
9642                                                                 }
9643                                                             }
9644                 }
9645             }
9646         function GJC_Input(){
9647             if( this.value.endsWith("\n") ){ // remove NL added by textarea
9648                 this.value = this.value.slice(0,this.value.length-1)
9649             }
9650         }
9651         var GCJ_Id = null
9652         function GJC_Resize(){
9653             GCJ_Id.style.zIndex = 20000
9654             //GCJ_Id.style.width = window.innerWidth - 16
9655             GCJ_Id.style.width = '100%'
9656             GCJ_Id.style.height = 300
9657             GCJ_Id.style.backgroundColor = "rgba(0,64,16,1.0)" // blackboard color
9658             GCJ_Id.style.color = "rgba(255,255,255,1.0)"
9659         }
9660         function GJC_FocusIn(){
9661             this.spellCheck = false
9662             SuppressGJShell = true
9663             this.onkeydown = GJC_Keydown
9664             GJC_Resize()
9665         }
9666         function GJC_FocusOut(){
9667             SuppressGJShell = false
9668             this.removeEventListener('keydown',GJC_Keydown);
9669         }
9670         window.addEventListener('resize',GJC_Resize);
9671     }
9672     function GJC_OnStorage(e){
9673         //alert('Got Message')
9674         //GCJ.value += "\n((ReceivedMessage))\n"
9675     }
9676     window.addEventListener('storage',GJC_OnStorage);
9677     //window.addEventListener('storage',()=>{alert('GotMessage')})
9678 }
9679 function GJC_Setup(gjcId){
9680     //gjcId.style.width = gsh.getBoudingClientRect().width
9681     gjcId.style.width = '100%'
9682     gjcId.value = "GJShell Console // " + GshVersion.innerHTML + "\n"
9683     //gjcId.value += "Date: " + DateLong() + "\n"
9684     gjcId.value += PS1
9685     gjcId.onfocus = GJC_FocusIn
9686     gjcId.addEventListener('input',GJC_Input);
9687     gjcId.addEventListener('focusout',GJC_FocusOut);
9688     GCJ_Id = gjcId
9689 }
9690 function GJC_Clear(id){
9691 }
9692 function GJConsole_initConsole(){
9693     if( document.getElementById("GJC_0") != null ){
9694         GJC_Setup(GJC_0)
9695     }else{
9696         GJC_Container.innerHTML = '<'
9697             + "textArea id='GJC_1' class='GJConsole'><'+/textArea>";
9698         GJC_Setup(GJC_1)
9699         factory = document.createElement('span');
9700         gsh.appendChild(factory);
9701         GJE_RootNode = factory;
9702         GJE_CurElement = GJE_RootNode;
9703     }
9704 }
9705 var initGJCF = false;
9706 function GJConsole_initFactory(){
9707     if( initGJCF ){ return; }
9708     GShell_initKeyCommands();
9709     GJConsole_initConsole();
9710 }
9711 //GJConsole_initFactory();
9712 // TODO: focus handling
9713 </script>
9714 <style>
9715 .GJ_StyleEditor {
9716     font-size:9pt !important;
9717     font-family:Courier New, monospace !important;
9718 }
9719 </style>
9720 </details>
9721 </span>
9722 <!-- ----- GJConsole END ; ----- -->
9723 */
9724 */
9725 */
9726 /*
9727 <span id="BlinderText">
9728 <style id="BlinderTextStyle">
9729 #GJLinkView {
9730     xposition:absolute; z-index:5000;
9731     position:relative;
9732     display:block;
9733     left:8px;
9734     color:#fff;

```

```

9735 width:800px; height:300px; resize:both;
9736 margin:0px; padding:4px;
9737 background-color:rgba(200,200,200,0.5) !important;
9738 }
9739 .MsggText {
9740 width:576px !important;
9741 resize:both !important;
9742 color:#000 !important;
9743 }
9744 .GjNote {
9745 font-family:Georgia !important;
9746 font-size:13pt !important;
9747 color:#22a !important;
9748 }
9749 .textField {
9750 display:inline;
9751 border:0.5px solid #444;
9752 border-radius:3px;
9753 color:#000; background-color:#fff;
9754 width:106pt; height:18pt;
9755 margin:2px;
9756 padding:2px;
9757 resize:none;
9758 vertical-align:middle;
9759 font-size:10pt; font-family:Courier New;
9760 }
9761 .textLabel {
9762 border:0px solid #000 !important;
9763 background-color:rgba(0,0,0,0);
9764 }
9765 .textURL {
9766 width:300pt !important;
9767 border:0px solid #000 !important;
9768 background-color:rgba(0,0,0,0);
9769 }
9770 .VisibleText {
9771 }
9772 .BlinderText {
9773 color:#000; background-color:#eee;
9774 }
9775 .JoinButton {
9776 font-family:Georgia !important;
9777 font-size:11pt;
9778 line-height:1.1;
9779 height:18pt;
9780 width:50pt;
9781 padding:3px !important;
9782 text-align:center !important;
9783 border-color:#aaa !important;
9784 border-radius:5px;
9785 color:#fff; background-color:#4a4 !important;
9786 vertical-align:middle !important;
9787 }
9788 .SendButton {
9789 vertical-align:top;
9790 }
9791 .ws0_log {
9792 font-size:10pt;
9793 color:#000 !important;
9794 line-height:1.0;
9795 background-color:rgba(255,255,255,0.7) !important;
9796 font-family:Courier New,monospace !important;
9797 width:99.3%;
9798 white-space:pre;
9799 }
9800 </style>
9801
9802 <!-- Form autofill test
9803 Location: <input id="xxserv" type="text" value="https://192.168.10.1/boafm/formLogin" size="80">
9804 <form method="POST" id="xxform" action="https://192.168.10.1/boafm/formLogin">
9805 dest? <input id="xds" name="dest" type="text" size="80" value="/index_contents.html">
9806 -->
9807 <details><summary>Form Auto. Filling</summary>
9808 <style>
9809 .xxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
9810 display:inline !important; font-size:10pt !important; padding:1px !important;
9811 }
9812 </style>
9813 <span style="font-family:Courier New;color:black;font-size:12pt;" onactive="">
9814 <form method="POST" id="xxform" action="https://192.168.10.1/" style="white-space:pre; size="80">
9815 Location: <input id="xxserv" class="xxinput" type="text" value="https://192.168.10.1/" size="80">
9816 Username: <input id="xxuser" class="xxinput" name="user" type="text" autocomplete="on">
9817 Password: <input id="xxpass" class="xxinput" name="pass" type="password" autocomplete="on">
9818 SessionId:<input id="xxsid" class="xxinput" name="SESSION_ID" type="text" size="80">
9819 <input id="xxsubm" class="xxinput" type="submit" value="Submit"></form>
9820 </span>
9821 <script>
9822 function XXSetFormAction(){
9823     xxform.setAttribute( 'action',xxserv.value);
9824 }
9825 xxform.setAttribute( 'action',xxserv.value);
9826 xxserv.addEventListener( 'change',XXSetFormAction);
9827 //xxserv.value = location.href;
9828 </script>
9829 </details>
9830 */
9831
9832 /*
9833 <details id="BlinderTextClass"><summary>BlinderText</summary>
9834 <span class="gsh-src">
9835 <span id="BlinderTextScript">
9836 // https://w3c.github.io/ievents/#event-type-keydown
9837 //
9838 // 2020-09-21 class BlinderText - textarea element not to be readable
9839 //
9840 // BlinderText attributes
9841 // bl_plainText = null
9842 // bl_hideChecksum = [false]
9843 // bl_showLength = [false]
9844 // bl_visible = [false]
9845 // data-bl_config = []
9846 // - min. length
9847 // - max. length
9848 // - acceptable charset in generate text
9849 //
9850 function BlinderChecksum(text){
9851     plain = text.bl_plainText;
9852     return strCRC32(plain,plain.length).toFixed(0);
9853 }
9854 function BlinderKeydown(ev){
9855     pass = ev.target;
9856     if( ev.code == 'Enter' ){
9857         ev.preventDefault();
9858     }
9859     ev.stopPropagation()
9860 }
9861 function BlinderKeyUp1(ev){
9862     blind = ev.target;
9863     if( ev.code == 'Backspace'){
9864         blind.bl_plainText = blind.bl_plainText.slice(0,blind.bl_plainText.length-1)
9865     }else
9866     if( and(ev.code == 'KeyV', ev.ctrlKey) ){
9867         blind.bl_visible = !blind.bl_visible;
9868     }else
9869     if( and(ev.code == 'KeyL', ev.ctrlKey) ){
9870         blind.bl_showLength = !blind.bl_showLength;
9871     }else
9872     if( and(ev.code == 'KeyU', ev.ctrlKey) ){
9873         blind.bl_plainText = "";
9874     }else
9875     if( and(ev.code == 'KeyR', ev.ctrlKey) ){
9876         checksum = BlinderChecksum(blind);
9877         blind.bl_plainText = checksum; //.toString(32);
9878     }else
9879     if( ev.code == 'Enter' ){
9880         ev.stopPropagation();
9881         ev.preventDefault();
9882         return;
9883     }else
9884     if( ev.key.length != 1 ){
9885         console.log('KeyUp: '+ev.code+'/'+ev.key);
9886         return;
9887     }else{
9888         blind.bl_plainText += ev.key;
9889     }
9890 }
9891
9892 leng = blind.bl_plainText.length;
9893 //console.log('KeyUp: '+ev.code+'/'+blind.bl_plainText);
9894 checksum = BlinderChecksum(blind) % 10; // show last one digit only
9895
9896 visual = '';
9897 if( !blind.bl_hideChecksum || blind.bl_showLength ){
9898     visual += '|';
9899 }
9900 if( !blind.bl_hideChecksum ){
9901     visual += '#' +checksum.toString(10);
9902 }
9903 if( blind.bl_showLength ){
9904     visual += '/' + leng;
9905 }
9906 if( !blind.bl_hideChecksum || blind.bl_showLength ){
9907     visual += '|';
9908 }
9909 if( blind.bl_visible ){
9910     visual += blind.bl_plainText;
9911 }else{
9912     visual += '*'.repeat(leng);

```



```

9912     }
9913     blind.value = visual;
9914 }
9915 function BlinderKeyUp(ev){
9916     BlinderKeyUp1(ev);
9917     ev.stopPropagation();
9918 }
9919 // https://w3c.github.io/ievents/#keyboardevent
9920 // https://w3c.github.io/ievents/#uievent
9921 // https://dom.spec.whatwg.org/#event
9922 function BlinderTextEvent(){
9923     ev = event;
9924     blind = ev.target;
9925     console.log('Event '+ev.type+'@'+blind.nodeName+'#'+blind.id)
9926     if( ev.type == 'keyup' ){
9927         BlinderKeyUp(ev);
9928     }else
9929     if( ev.type == 'keydown' ){
9930         BlinderKeyDown(ev);
9931     }else{
9932         console.log('thru-event '+ev.type+'@'+blind.nodeName+'#'+blind.id)
9933     }
9934 }
9935 //< textarea hidden id="BlinderTextClassDef" class="textField"
9936 // onkeydown="BlinderTextEvent()" onkeyup="BlinderTextEvent()"
9937 // spellcheck="false">< /textarea>
9938 //< textarea hidden id="gj_pass1"
9939 // class="textField BlinderText"
9940 // placeholder="PassWord1"
9941 // onkeydown="BlinderTextEvent()"
9942 // onkeyup="BlinderTextEvent()"
9943 // spellcheck="false">
9944 function SetupBlinderText(parent,txa,phold){
9945     if( txa == null ){
9946         txa = document.createElement('textarea');
9947         //txa.id = id;
9948     }
9949     txa.setAttribute('class','textField BlinderText');
9950     txa.setAttribute('placeholder',phold);
9951     txa.setAttribute('onkeydown','BlinderTextEvent()');
9952     txa.setAttribute('onkeyup','BlinderTextEvent()');
9953     txa.setAttribute('spellcheck','false');
9954     //txa.setAttribute('bl_plainText','false');
9955     txa.bl_plainText = '';
9956     //parent.appendChild(txa);
9957 }
9958 function DestroyBlinderText(txa){
9959     txa.removeAttribute('class');
9960     txa.removeAttribute('placeholder');
9961     txa.removeAttribute('onkeydown');
9962     txa.removeAttribute('onkeyup');
9963     txa.removeAttribute('spellcheck');
9964     txa.bl_plainText = '';
9965 }
9966 //
9967 // visible textarea like Username
9968 //
9969 function VisibleTextEvent(){
9970     if( event.code == 'Enter' ){
9971         if( event.target.NoEnter ){
9972             event.preventDefault();
9973         }
9974     }
9975     event.stopPropagation();
9976 }
9977 function SetupVisibleText(parent,txa,phold){
9978     if( false ){
9979         txa.setAttribute('class','textField VisibleText');
9980     }else{
9981         newclass = txa.getAttribute('class');
9982         if( and(newclass != null, newclass != '') ){
9983             newclass += ' ';
9984         }
9985         newclass += 'VisibleText';
9986         txa.setAttribute('class',newclass);
9987     }
9988     //console.log('SetupVisibleText class='+txa.class);
9989     txa.setAttribute('placeholder',phold);
9990     txa.setAttribute('onkeydown','VisibleTextEvent()');
9991     txa.setAttribute('onkeyup','VisibleTextEvent()');
9992     txa.setAttribute('spellcheck','false');
9993     cols = txa.getAttribute('cols');
9994     if( cols != null ){
9995         txa.style.width = '580px';
9996         //console.log('VisualText'+txa.id+' cols='+cols)
9997     }else{
9998         //console.log('VisualText'+txa.id+' NO cols')
9999     }
10000     rows = txa.getAttribute('rows');
10001     if( rows != null ){
10002         txa.style.height = '30px';
10003         txa.style.resize = 'both';
10004         txa.NoEnter = false;
10005     }else{
10006         txa.NoEnter = true;
10007     }
10008 }
10009 function DestroyVisibleText(txa){
10010     txa.removeAttribute('class');
10011     txa.removeAttribute('placeholder');
10012     txa.removeAttribute('onkeydown');
10013     txa.removeAttribute('onkeyup');
10014     txa.removeAttribute('spellcheck');
10015     cols = txa.removeAttribute('cols');
10016 }
10017 </span>
10018 <script>
10019 js = document.getElementById('BlinderTextScript');
10020 eval(js.innerHTML);
10021 //js.outerHTML = ""
10022 </script>
10023 </span><!-- end of class="gsh-src" -->
10024 <details>
10025 </span>
10026 */
10027 *
10028 <script id="GJLinkScript">
10029 function gj_key_hash(text){
10030     return strCRC32(text,text.length) % 0x10000;
10031 }
10032 function gj_addlog(e,msg){
10033     now = (new Date()).getTime() / 1000).toFixed(3);
10034     tstp = ('+now+'); //'+now+'
10035     e.value += tstp + msg;
10036     e.scrollTop = e.scrollHeight;
10037 }
10038 function gj_addlog_cl(msg){
10039     ws0_log.value += '(console.log) ' + msg + '\n';
10040 }
10041 var GJ_Channel = null;
10042 var GJ_Log = null;
10043 var gj; // the global variable
10044 function GJ_Join(){
10045     target = gj_join;
10046     if( target.value == 'Leave' ){
10047         GJ_Channel.close();
10048     }
10049     GJ_Channel = null;
10050     target.value = 'Join';
10051     return;
10052 }
10053 }
10054 }
10055 var ws0;
10056 var ws0_log;
10057 }
10058 sav_console_log = console.error
10059 console.error = gj_addlog_cl
10060 ws0 = new WebSocket(gj_servv.innerHTML);
10061 console.error = sav_console_log
10062 }
10063 GJ_Channel = ws0;
10064 ws0_log = document.getElementById('ws0_log');
10065 GJ_Log = ws0_log;
10066 }
10067 now = (new Date()).getTime() / 1000).toFixed(3);
10068 const wstats = ["CONNECTING","OPEN","CLOSING","CLOSED"];
10069 cst = wstats[ws0.readyState];
10070 gj_addlog(ws0_log,'stat '+ws0.readyState+'('+cst+') : GJ Linked\n');
10071 }
10072 ws0.addEventListener('error', function(event){
10073     gj_addlog(ws0_log,'stat error : transport error?\n');
10074 });
10075 }
10076 ws0.addEventListener('open', function(event){
10077     GJLinkView.style.zIndex = 10000;
10078 //console.log('#'+GJLinkView.id+'.zIndex='+GJLinkView.style.zIndex);
10079 date1 = new Date().getTime();
10080 date2 = (date1 / 1000).toFixed(3);
10081 seed = date1.toString(16);
10082 }
10083 // user name and key
10084 user = document.getElementById('gj_user').value;
10085 if( user.length == 0 ){
10086     gj_user.value = 'nemo';
10087     user = 'nemo';
10088 }
10089 key1 = document.getElementById('gj_ukey').bl_plainText;

```

```

10089   ukey = gjkey_hash(seed+user+key1).toString(16);
10090
10091   // session name and key
10092   chan = document.getElementById('gj_chan').value;
10093   if( chan.length == 0 ){
10094       gj_chan.value = 'main';
10095       chan = 'main';
10096   }
10097   key2 = document.getElementById('gj_ckey').bl_plainText;
10098   ckey = gjkey_hash(seed+chan+key2).toString(16);
10099
10100   msg = date2 + ' JOIN ' + user + '|' + chan + ' ' + ukey + '|' + ckey;
10101   ws0_addlog(ws0_log, 'send '+msg+'\n');
10102   ws0_send(msg);
10103
10104   target.value = 'Leave';
10105   //console.log(['+date2+'] #' +target.id+' +target.value+'\n');
10106   //gj_addlog(ws0_log, 'label '+target.value+'\n');
10107 });
10108 ws0.addEventListener('message', function(event){
10109     now = (new Date()).getTime() / 1000).toFixed(3);
10110     msg = event.data;
10111     if( false ){
10112         gj_addlog(ws0_log, 'recv '+msg+'\n');
10113     }
10114     argv = msg.split(' ');
10115     tstamp = argv[0];
10116     argv.shift();
10117     if( argv[0] == 'reload' ){
10118         location.reload()
10119     }
10120     gjcmd = argv[0];
10121     otstamp = '';
10122     if( gjcmd == 'CAST' ){ // from reflector
10123         otstamp = argv[0];
10124         argv.shift(); // original time stamp
10125         ofrom = argv[0];
10126         argv.shift(); // original from
10127     }
10128     argv.shift(); // command
10129     from = argv[0];
10130     argv.shift(); // from|to
10131     cmdl = argv[0];
10132     argv.shift(); // xxxx command
10133
10134     if( false ){
10135         gj_addlog(ws0_log, '--'
10136             + ' tstamp=' + tstamp
10137             + ' gjcmd=' + gjcmd
10138             + ' from=' + from
10139             + ',cmdl=' + cmdl + '|'
10140             + ' '+argv+'\n');
10141     }
10142     if( cmdl == 'auth' ){
10143         // doing authorization required
10144     }
10145     if( cmdl == 'echo' ){
10146         now = (new Date()).getTime() / 1000).toFixed(3);
10147         msg = now + ' +RESP '+argv.join(' ');
10148         gj_addlog(ws0_log, 'send '+msg+'\n');
10149         ws0_send(msg);
10150     }
10151     if( cmdl == 'eval' ){
10152         argv.shift();
10153         js = argv.join(' ');
10154         ret = eval(js); // <----- eval()
10155         gj_addlog(ws0_log, 'eval '+js+' = '+ret+'\n');
10156         now = (new Date()).getTime() / 1000).toFixed(3);
10157         msg = now + ' +RESP ' + ret;
10158         ws0_send(msg);
10159         gj_addlog(ws0_log, 'send '+msg+'\n')
10160     }
10161     if( cmdl == 'DRAW' ){
10162         if( false ){
10163             gj_addlog(ws0_log, 'DRAW '+argv[0]+'\n')
10164         }
10165         Pointillism_RemoteDraw(argv[0]);
10166     }
10167     if( cmdl == 'MOUSEPOSITION' ){
10168         XYeses_recvHousePosition(argv[0]);
10169     }
10170 });
10171 ws0.addEventListener('close', function(event){
10172     if( GJ_Channel == null ){
10173         gj_addlog(ws0_log, 'stat OK : GJ UnLinked\n');
10174         return;
10175     }
10176     GJ_Channel.close();
10177     GJ_Channel = null;
10178     target.value = 'Join';
10179     gj_addlog(ws0_log, 'stat error : close : GJ UnLinked unexpectedly\n');
10180 });
10181
10182 function GJ_BcastMessageUserPass(user,chan,msgbody){
10183     now = (new Date()).getTime() / 1000).toFixed(3);
10184     msg = now + ' BCAST '+user+' |'+chan+' '+msgbody;
10185     if( false ){
10186         gj_addlog(GJ_Log, 'send '+msg+'\n');
10187     }
10188     GJ_Channel.send(msg);
10189 }
10190
10191 function GJ_BcastMessage(msgbody){
10192     if( GJ_Channel == null ){
10193         gj_addlog(ws0_log, 'stat error : send : GJ not Linked\n');
10194         return;
10195     }
10196     //target = event.target;
10197     user = document.getElementById('gj_user').value;
10198     chan = document.getElementById('gj_chan').value;
10199     GJ_BcastMessageUserPass(user,chan,msgbody);
10200 }
10201 function GJ_SendMessageUserPass(user,chan,msgbody){
10202     now = (new Date()).getTime() / 1000).toFixed(3);
10203     msg = now + ' ISM '+user+' |'+chan+' '+msgbody;
10204     gj_addlog(GJ_Log, 'send '+msg+'\n');
10205     GJ_Channel.send(msg);
10206 }
10207 function GJ_SendMessage(msgbody){
10208     if( GJ_Channel == null ){
10209         gj_addlog(ws0_log, 'stat error : send : GJ not Linked\n');
10210         return;
10211     }
10212     //target = event.target;
10213     user = document.getElementById('gj_user').value;
10214     chan = document.getElementById('gj_chan').value;
10215     GJ_SendMessageUserPass(user,chan,msgbody);
10216 }
10217 function GJ_Send(){
10218     msgbody = gj_sendText.value;
10219     GJ_SendMessage(msgbody);
10220 }
10221 </script>
10222 <!-- ----- GJLINK ----->
10223 <!--
10224 - User can subscribe to a channel
10225 - A channel will be broadcasted
10226 - A channel can be a pattern (regular expression)
10227 - User is like From:(me) and channel is like To: or Recipient:
10228 - like VIABUS
10229 - watch message with SENDME, WATCH, CATCH, HEAR, or so
10230 - routing with path expression or name pattern (with routing with DNS like system)
10231 -->
10232 */
10233
10234 <span id="GJLinkGolang">
10235 // <details id="GshWebSocket"><summary>Golang / JavaScript Link</summary>
10236 // <span class="gsh-src"><!-- -->
10237 // 2020-0920 created
10238 // <a href="https://pkg.go.dev/golang.org/x/net/websocket">WS</a>
10239 // <a href="https://godoc.org/golang.org/x/net/websocket">WS</a>
10240 // INSTALL: go get golang.org/x/net/websocket
10241 // INSTALL: sudo (apt, yum) install git (if git is not instlled yet)
10242 // import "golang.org/x/net/websocket"
10243 const gshws_origin = "http://localhost:9999"
10244 const gshws_server = "localhost:9999"
10245 const gshws_port = 9999
10246 const gshws_path = "gjlink1"
10247 const gshws_url = "ws://" + gshws_server + "/" + gshws_path
10248 const GSHWS_MSGSIZE = (8*1024)
10249 func fmtstring(fmts string, params ...interface{})(string){
10250     return fmt.Sprintf(fmts,params...)
10251 }
10252 func GSHWS_MARK(what string)(string){
10253     now := time.Now()
10254     us := fmtstring("%06d",now.Nanosecond() / 1000)
10255     mark := ""
10256     if( !atConsoleLineTop ){
10257         mark += "\n"
10258         atConsoleLineTop = true
10259     }
10260     mark += "[" + now.Format(time.Stamp) + " +us+" ] -GJ- " + what + " : "
10261     return mark
10262 }
10263 func gehk(what string,err error){
10264     if( err != nil ){
10265         panic(GSHWS_MARK(what)+err.Error())
10266     }
10267 }

```

```

10266 }
10267 }
10268 func glog(what string, fmts string, params ...interface{}){
10269     fmt.Print(GSHWS_MARK(what))
10270     fmt.Printf(fmts+"\n",params...)
10271 }
10272 }
10273 var WSV = []*websocket.Conn{}
10274 func jsend(argv []string){
10275     if len(argv) <= 1 {
10276         fmt.Printf("--ij %v [-m] command arguments\n",argv[0])
10277         return
10278     }
10279     argv = argv[1:]
10280     if( len(WSV) == 0 ){
10281         fmt.Printf("--Ej-- No link now\n")
10282         return
10283     }
10284     if( 1 < len(WSV) ){
10285         fmt.Printf("--Ij-- multiple links (%v)\n",len(WSV))
10286     }
10287 }
10288 multicast := false // should be filtered with regex
10289 if( 0 < len(argv) && argv[0] == "-m" ){
10290     multicast = true
10291     argv = argv[1:]
10292 }
10293 args := strings.Join(argv, " ")
10294 }
10295 now := time.Now()
10296 msec := now.UnixNano() / 1000000;
10297 tstamp := fmtstring("%.3f",float64(msec)/1000.0)
10298 msg := fmtstring("%v SEND gshell* %v",tstamp,args)
10299 }
10300 if( multicast ){
10301     for i,ws := range WSV {
10302         wn,werr := ws.Write([]byte(msg))
10303         if( werr != nil ){
10304             fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
10305         }
10306         glog("SQ",fmtstring("(%v) %v",wn,msg))
10307     }
10308 }else{
10309     i := 0
10310     ws := WSV[i]
10311     wn,werr := ws.Write([]byte(msg))
10312     if( werr != nil ){
10313         fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
10314     }
10315     glog("SQ",fmtstring("(%v) %v",wn,msg))
10316 }
10317 }
10318 func ws_broadcast(msg string)(wn int,werr error){
10319     for i,ws := range WSV {
10320         wn,werr := ws.Write([]byte(msg))
10321         if( werr != nil ){
10322             fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
10323         }
10324         glog("SQ",fmtstring("(%v) %v",wn,msg))
10325     }
10326     return wn,werr;
10327 }
10328 func serv1(ws *websocket.Conn) {
10329     WSV = append(WSV,ws)
10330     //fmt.Print("\n")
10331     glog("CO", "accepted connections [%v]",len(WSV))
10332     //remoteAddr := ws.RemoteAddr
10333     //fmt.Printf("-- accepted %v\n",remoteAddr)
10334     //fmt.Printf("-- accepted %v\n",ws.Config())
10335     //fmt.Printf("-- accepted %v\n",ws.Config().Header)
10336     //fmt.Printf("-- accepted %v // %v\n",ws,serv1)
10337 }
10338 var reqb = make([]byte,GSHWS_MSGSIZE)
10339 for {
10340     rn, rerr := ws.Read(reqb)
10341     if( rerr != nil || rn < 0 ){
10342         glog("SQ",fmtstring("(%v) %v",rn,rerr))
10343         break
10344     }
10345     req := string(reqb[0:rn])
10346     glog("SQ",fmtstring("(%v) %v",rn,req))
10347 }
10348 margv := strings.Split(req, " ")
10349 margv = margv[1:]
10350 if( 0 < len(margv) ){
10351     if( margv[0] == "RESP" ){
10352         // should forward to the destination
10353         continue;
10354     }
10355 }
10356 now := time.Now()
10357 msec := now.UnixNano() / 1000000;
10358 tstamp := fmtstring("%.3f",float64(msec)/1000.0)
10359 res := fmtstring("%v +\"CAST\"+ %v",tstamp,req)
10360 }
10361 wn := 0;
10362 werr := error(nil);
10363 if( 0 < len(margv) && margv[0] == "BCAST" ){
10364     wn, werr = ws_broadcast(res);
10365 }else{
10366     wn, werr = ws.Write([]byte(res))
10367 }
10368 gchk("SE",werr)
10369 glog("SR",fmtstring("(%v) %v",wn,string(res)))
10370 }
10371 glog("SF", "WS response finish")
10372 }
10373 wsv := []*websocket.Conn{}
10374 wsx := 0
10375 for i, v := range WSV {
10376     if( v != ws ){
10377         wsx = i
10378     }
10379     wsv = append(wsv,v)
10380 }
10381 WSV = wsv
10382 //glog("CO", "closed %v",ws)
10383 glog("CO", "closed connection [%v/%v]",wsx+1,len(WSV)+1)
10384 ws.Close()
10385 }
10386 // url := [scheme://host[:port]/path]
10387 func decomp_URL(url string){
10388 }
10389 func full_wsURL(){
10390 }
10391 func gj_server(argv []string) {
10392     gjserv := gshws_url
10393     gjport := gshws_server
10394     gjpath := gshws_path
10395     gjscheme := "ws"
10396 }
10397 //cmd := argv[0]
10398 argv = argv[1:]
10399 if( 1 <= len(argv) ){
10400     serv = argv[0]
10401     if( 0 < strings.Index(serv,":/") ){
10402         schemev := strings.Split(serv,"://")
10403         gjscheme = schemev[0]
10404     }
10405     serv = schemev[1]
10406     if( 0 < strings.Index(serv,"/") ){
10407         pathv := strings.Split(serv,"/")
10408         serv = pathv[0]
10409         gjpath = pathv[1]
10410     }
10411     servv := strings.Split(serv,":")
10412     host := "localhost"
10413     port := 9999
10414     if( servv[0] != "" ){
10415         host = servv[0]
10416     }
10417     if( len(servv) == 2 ){
10418         fmt.Sscanf(servv[1],"%d",&port)
10419     }
10420     //glog("LC", "hostport=%v (%v : %v)",servv,host,port)
10421     gjport = fmt.Sprintf("%v:%v",host,port)
10422     gjserv = gjscheme + "://" + gjport + "/" + gjpath
10423 }
10424 glog("LS",fmtstring("listening at %v",gjserv))
10425 http.Handle("/"+gjpath,websocket.Handler(serv1))
10426 err := error(nil)
10427 if( gjscheme == "wss" ){
10428     // https://golang.org/pkg/net/http/#ListenAndServeTLS
10429     //err = http.ListenAndServeTLS(gjport,nil)
10430 }else{
10431     err = http.ListenAndServe(gjport,nil)
10432 }
10433 gchk("LE",err)
10434 }
10435 }
10436 func gj_client(argv []string) {
10437     glog("CS",fmtstring("connecting to %v",gshws_url))
10438     ws, err := websocket.Dial(gshws_url,"",gshws_origin)
10439     gchk("C",err)
10440 }
10441 var resb = make([]byte, GSHWS_MSGSIZE)
10442 for qi := 0; qi < 3; qi++ {

```

```

10443     req := fmtstring("Hello, GShell! (%v)",qi)
10444     wn, werr := ws.Write([]byte(req))
10445     glog("QM",fmtstring("(%v) %v",wn,req))
10446     gchk("QE",werr)
10447     rn, rerr := ws.Read(resb)
10448     gchk("RE",rerr)
10449     glcg("RM",fmtstring("(%v) %v",rn,string(resb)))
10450     }
10451     glog("CF", "WS request finish")
10452 }
10453 //
```

```

1062<details><summary>Live HTML Snapshot</summary>
1062<span id="LiveHTML">
1062<!-- ----- HTML Snapshot: Edit, save and load // 2020-0924 SatoxITS { -->
1062<div class="GshMenu">
1062<span class="GshMenu" onclick="html_edit();">Edit</span>
1062<span class="GshMenu" onclick="html_save();">Save</span>
1062<span class="GshMenu" onclick="html_load();">Load</span>
1062<span class="GshMenu" onclick="html_ver0();">Vers</span>
1062</div>
1062<div>
1063<input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="showLiveHTMLCode()" >
1063<span id="LiveHTML_Codeview"></span>
1063</div>
1063<script id="LiveHtmlScript">
1063function showLiveHTMLCode(){
1063showHtmlCode(LiveHTML_Codeview,LiveHTML);
1063}
1063var editable = false;
1063var savSuppressGJShell = false;
1063function ToggleEditMode(){
1064editable = !editable;
1064if(editable){
1064savSuppressGJShell = SuppressGJShell;
1064SuppressGJShell = true;
1064gsh.setAttribute('contenteditable','true');
1064GshMenuEdit.innerHTML = 'Lock';
1064GshMenuEdit.style.color = 'rgba(255,0,0,1)';
1064GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
1064}else{
1064SuppressGJShell = savSuppressGJShell;
1064gsh.setAttribute('contenteditable','false');
1064GshMenuEdit.innerHTML = 'Edit';
1064GshMenuEdit.style.color = 'rgba(16,160,16,1)';
1064GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
1064}
1065}
1065function html_edit(){
1065ToggleEditMode();
1065}
1065}
1065// Live HTML (DOM) Snapshot onto browser's localStorage
1065// 2020-0923 SatoxITS
1065var htRoot = gsh // -- Element-ID, should be selectable
1065const snappedHTML = "SnappedHTML"; // Item-ID of the HTML data in localStogate
1065// -- should be a [map] of URL
1065// -- should be with CSSOM as inline script
1065const htVersionTag = 'VersionTag'; // VesionTag Belment-ID in the HTML (in DOM)
1065function showVersion(note,w,v,u,t){
1065w.alert("note: ' + v + '\n
1065 + -- URL: ' + u + '\n
1065 + -- Time: ' + t + ' (' + DateLong0(t*1000) + ')")
1065}
1065}
1065function html_save(){
1065u = document.URL;
1065t = new Date().getTimes() / 1000;
1065v = '<+>span id="'+htVersionTag+'" data-url="'+u+'" data-time="'+t+'";>
1065v += '<+>/span>\n';
1065h = v + htRoot.outerHTML;
1065localStorage.setItem(snappedHTML,h);
1065showVersion("Saved",window,v,u,t);
1065}
1065}
1065function html_load(){
1065h = localStorage.getItem(snappedHTML);
1065if( h == null ){
1065alert("No snapshot taken yet");
1065return;
1065}
1065w = window.open('','','');
1065d = w.document;
1065d.write(h);
1065w.focus();
1065html_ver1("Loaded",w,d);
1065}
1065}
1065function html_ver1(note,w,d){
1065if( (v = d.getElementById(htVersionTag)) != null ){
1065h = v.outerHTML;
1065u = v.getAttribute('data-url');
1065t = v.getAttribute('data-time');
1065}else{
1065h = 'No version info. in the page';
1065u = '';
1065t = 0;
1065}
1065showVersion(note,w,v,u,t);
1065}
1065}
1065function html_ver0(){
1065html_ver1("Version",window,document);
1065}
1065}
1065</script>
1065<!-- LiveHTML } -->
1065</span>
1065</details>
1065/
1065/*
1065<details><summary>Event sharing</summary>
1065<span id="EventSharingCodeSpan">
1065}
1065<!-- ----- Event sharing // 2020-0925 SatoxITS { -->
1065}
1065}
1065<div id="iftestTemplate" class="iftest" hidden="">
1065<style>-:if(-ms-):family:Georgia;font-size:10pt; </style>
1065<span id="frameBody" class="iftestbody" onclick="frameClick()"><script>
1065function docadd(txt){
1065document.body.append(txt);
1065window.scrollTo(0,100000);
1065}
1065}
1065function frameClick(){
1065xy = 'x'+event.x + ' y'+event.y+'';
1065//docadd('Got Click on #'+event.target.id+' +xy+ '\n');
1065docadd('Got Click on #'+Fid.value+' , '+xy+ '\n');
1065window.scrollTo(0,100000);
1065window.parent.postMessage('OnClick: '+xy,'*');
1065}
1065}
1065function frameMouseMove(){
1065if( false ){
1065document.body.append('Mousemove on #'+event.target.id+' '
1065+ 'x'+event.x + ' y'+event.y + '\n');
1065}
1065peerWin = window.frames.iframe1;
1065document.body.append('Send to peer #'+peerWin+' ' + '\n');
1065window.scrollTo(0,100000);
1065peerWin.postMessage('Hi!', '*');
1065}
1065}
1065function frameKeydown(){
1065msg = 'Got Keydown: #'+Fid.value+' , ('+event.code+')';
1065docadd(msg+'\n');
1065window.parent.postMessage(msg,'*');
1065}
1065}
1065function frameOnMessage(){
1065docadd('Message ' + event.data + '\n');
1065window.scrollTo(0,100000);
1065}
1065}
1065if( document.getElementById('Fid') ){
1065frameBody.id = Fid.value;
1065h = '';
1065h += '<+>style*<+>';
1065h += 'font-size:10pt;white-space:pre-wrap;';
1065h += 'font-family:Courier New;';
1065h += '<+>+<+>style*<+>';
1065h += 'I am '+Fid.value+'\n';
1065document.write(h);
1065window.addEventListener('click',frameClick);
1065window.addEventListener('keydown',frameKeydown);
1065window.addEventListener('message',frameOnMessage);
1065window.addEventListener('mousemove',frameMouseMove);
1065window.parent.postMessage('Hi parent, I am '+Fid.value,'*');
1065}
1065}
1065</script></span></div>
1065}
1065}
1065<style>.iframeTest{margin:3px;resize:both;width:370px;height:120px;}</style>
1065<h2>Inter-window communication</h2>
1065}
1065}
1065<note>
1065frame0 >>> frame1 and frame2<br>
1065frame1 >>> frame0 and frame2<br>
1065frame2 >>> frame0 and frame1<br>
1065}
1065}
1065</note>
1065}
1065}
1065<div id="iframe-test">
1065<pre id="iframeTest" style="border:1px;font-family:Courier New;font-size:10pt;"/></pre>
1065<iframe id="iframe0" title="iframe0" class="iframeTest"></iframe>
1065<iframe id="iframe1" title="iframe1" class="iframeTest"></iframe>
1065<iframe id="iframe2" title="iframe2" class="iframeTest"></iframe>
1065</div>
1065}
1065}
1065<script id="if0-test-script">
1065function InterFrameComm_init(){
1065setupFrames0();
1065setupFrames1();
1065}
1065}
1065function setFrameSrcdoc(dst,src){
1065if( true ){
1065dst.contentWindow.document.write(src);
1065// this makes browser write close, and crash if accumulated !?
1065// so it should be closed after write
1065dst.contentWindow.document.close();
1065}else{

```

```

10797 // to be erased before source dump
10798 // but should be set for live snapshot
10799 dst.srcdoc = src;
10800 }
10801 }
10802 function setupFrames0(){
10803   tbody = iframe0.contentWindow.document.body;
10804   iframe0.style.width = "755px"
10805   //iframeHost.innerHTML = "Message exchange at iframes' host:\n";
10806   window.addEventListener('message',messageFromChild);
10807 }
10808 if0 = '';
10809 if0 += '<pre style="font-family:Courier New;">';
10810 if0 += '<input id="Pid" value="iframe0">';
10811 if0 += iftestTemplate.innerHTML;
10812 setFrameSrcdoc(iframe0,if0);
10813 }
10814 function clickOnChild(){
10815   console.log('clickOn #' + this.id);
10816 }
10817 function moveOnChild(){
10818   console.log('moveOn #' + this.id);
10819 }
10820 iframe0.contentWindow.document.body.style.setProperty('white-space','pre');
10821 iframe0.contentWindow.document.body.style.setProperty('font-size','9pt');
10822 }
10823 function setupFrames12(){
10824   if1 = '<input id="Pid" value="iframe1">';
10825   if1 += iftestTemplate.innerHTML;
10826   setFrameSrcdoc(iframe1,if1);
10827   //iframe1.name = "iframe1"; // this seems break contentWindow
10828 }
10829 if2 = '<input id="Pid" value="iframe2">';
10830 if2 += iftestTemplate.innerHTML;
10831 setFrameSrcdoc(iframe2,if2);
10832 }
10833 iframe1.addEventListener('message',messageFromChild);
10834 //iframe1.addEventListener('mouseover',moveOnChild);
10835 iframe2.addEventListener('message',messageFromChild);
10836 //iframe2.addEventListener('mouseover',moveOnChild);
10837 iframe1.contentWindow.postMessage(['parent0'] Hi iframe1 -- from parent.', '*');
10838 //iframe1.contentWindow.postMessage(['parent0'] Hi iframe2 -- from parent.', '*');
10839 iframe2.contentWindow.postMessage(['parent0'] Hi iframe2 -- from parent.', '*');
10840 //iframe2.contentWindow.postMessage(['Your peer is '+iframe1.contentWindow, '*');
10841 }
10842 function messageFromChild(){
10843   from = null;
10844   forw = null;
10845   if( event.source == iframe0.contentWindow ){
10846     from = 'iframe0';
10847     forw = 'iframe12';
10848   }else
10849   if( event.source == iframe1.contentWindow ){
10850     from = 'iframe1';
10851     forw = 'iframe2';
10852   }else
10853   if( event.source == iframe2.contentWindow ){
10854     from = 'iframe2';
10855     forw = 'iframe1';
10856   }else
10857   {
10858     iframeHost.innerHTML += 'Message [unknown] '
10859     + ' orig=' + event.origin
10860     + ' data=' + event.data
10861     + ' from=' + event.source
10862     ;
10863   }
10864   msglog1 = from + event.data + ' -- '
10865   + ' from=' + event.source
10866   + ' orig=' + event.origin
10867   + ' name=' + event.source.name
10868   + ' port=' + event.ports
10869   + ' evid=' + event.lastEventId
10870   + '\n'
10871   ;
10872   if( true ){
10873     if( forw == 'iframe1' || forw == 'iframe12' ){
10874       iframe1.contentWindow.postMessage(from+event.data);
10875     }
10876     if( forw == 'iframe2' || forw == 'iframe12' ){
10877       iframe2.contentWindow.postMessage(from+event.data);
10878     }
10879   }
10880   txtadd0(msglog1);
10881 }
10882 function txtadd0(txt){
10883   iframe0.contentWindow.document.body.append(txt);
10884   iframe0.contentWindow.scrollTo(0,100000);
10885 }
10886 }
10887 function es_ShowSelf(){
10888   iframe2.setAttribute('src',document.URL);
10889   iframe2.setAttribute('src',document.URL);
10890 }
10891 </script>
10892 <input class="HtmlCodeViewButton" type="button" value="ShowSelf" onclick="es_ShowSelf()">
10893 <input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="es_showHtmlCode()">
10894 <span id="EventSharingCodeview"></span>
10895 <script id="EventSharingScript">
10896 function es_showHtmlCode(){
10897   showHtmlCode(EventSharingCodeview,EventSharingCodeSpan);
10898 }
10899 }
10900 DestroyEventSharingCodeview = function(){
10901   //EventSharingCodeview.parentNode.removeChild(EventSharingCodeview);
10902   EventSharingCodeview.innerHTML = "";
10903   iframe0.style = "";
10904   //iframe0.srcdoc = "erased";
10905   //iframe1.srcdoc = "erased";
10906   //iframe2.srcdoc = "erased";
10907 }
10908 </script>
10909 </-- EventSharing -->
10910 </span>
10911 </details>
10912 *
10913 *
10914 *
10915 <!-- ----- "GShell Inside" Notification { -->
10916 <script id="script-gshell-inside">
10917 var notices = 0;
10918 function noticeGShellInside(){
10919   ver = '';
10920   if( ver = document.getElementById('GshVersion') ){
10921     ver = ver.innerHTML;
10922   }
10923   console.log('GShell Inside (^-^)' + ver);
10924   notices += 1;
10925   if( 2 <= notices ){
10926     document.removeEventListener('mousemove',noticeGShellInside);
10927   }
10928 }
10929 document.addEventListener('mousemove',noticeGShellInside);
10930 noticeGShellInside();
10931 }
10932 const FooterName = 'GshFooter'
10933 function DestroyFooter(){
10934   if( {footer = document.getElementById(FooterName)} != null ){
10935     //footer.parentNode.removeChild(footer);
10936     empty = document.createElement('div');
10937     empty.id = 'GshFooter0';
10938     footer.parentNode.replaceChild(empty,footer);
10939   }
10940 }
10941 function showFooter(){
10942   footer = document.createElement('div');
10943   footer.id = FooterName;
10944   footer.style.backgroundImage = "url('"+ITSmoreQR+"')";
10945   //GshFooter0.parentNode.appendChild(footer);
10946   if( document.getElementById('GshFooter0') != null ){
10947     GshFooter0.parentNode.replaceChild(footer,GshFooter0);
10948   }
10949 }
10950 </script>
10951 <!-- } -->
10952 *
10953 <!-- ----- Web Virtual Desktop // 2020-0927 SatoxITS { -->
10954 <style>
10955 VirtualSpace {
10956   z-index:0;
10957   width:1280px !important; xheight:720px !important;
10958   width:5120px; height:2880px;
10959   border-width:0px;
10960   xbackground-color:rgba(32,32,160,0.8);
10961   xbackground-image:url('WD-WallPaper03.png');
10962   xbackground-size:100% 100%;
10963   color:#22a;xfont-family:Georgia;font-size:10pt;
10964   xoverflow:scroll;
10965 }
10966 VirtualGrid {

```

```

10974 z-index:0;
10975 position:absolute;
10976 width:800px; height:500px;
10977 border:1px inset #fff;
10978 color:rgba(192,255,192,0.8);
10979 font-family:Georgia, Courier New;
10980 text-align:right;
10981 vertical-align:middle;
10982 font-size:200px;
10983 text-shadow:4px 4px #ccf;
10984 }
10985 .WD_GridScroll {
10986 z-index:100000;
10987 background-color:rgba(200,200,200,0.1);
10988 }
10989 .VirtualDesktop {
10990 z-index:0;
10991 position:relative;
10992 resize:both !important;
10993 overflow:scroll;
10994 display:block;
10995 min-width:120px !important; min-height:60px !important;
10996 width:800px;
10997 height:500px;
10998 border:10px inset #228;
10999 border-width:30px; border-radius:20px;
11000 background-image:url("#WD-WallPaper03.png");
11001 background-size:100% 100%;
11002 color:#22a;font-family:Georgia;font-size:10pt;
11003 }
11004 .comment {
11005 // overflow:scroll seems to bound childrens' view in the element span
11006 // specifying overflow seems fix the position of the element
11007 }
11008 .VirtualBrowserSpan {
11009 z-index:10;
11010 xxxborder:0.5px dashed #fff !important;
11011 border-color:rgba(255,255,255,0.5) !important;
11012 position:relative;
11013 left:100px;
11014 top:100px;
11015 display:block;
11016 resize:both !important;
11017 width:540px;
11018 height:320px;
11019 min-width:40px !important; min-height:20px !important;
11020 max-width:5120px !important; max-height:2880px !important;
11021 background-color:rgba(255,200,255,0.1);
11022 xoverflow:scroll;
11023 }
11024 .xVirtualBrowserLocationBar:focus {
11025 color:#f00;
11026 background-color:rgba(255,128,128,0.2);
11027 }
11028 .xVirtualBrowserLocationBar:active {
11029 color:#f00;
11030 background-color:rgba(128,255,128,0.2);
11031 }
11032 .a.VirtualBrowserLocation {
11033 color:#ccc !important;
11034 text-decoration:none !important;
11035 }
11036 .a.VirtualBrowserLocation:hover {
11037 color:#fff !important;
11038 text-decoration:underline;
11039 }
11040 .VirtualBrowserLocationBar {
11041 position:absolute;
11042 z-index:100000;
11043 display:block;
11044 width:400px;
11045 height:20px;
11046 padding-left:2px;
11047 line-height:1.1;
11048 vertical-align:middle;
11049 font-size:14px;
11050 color:#fff;
11051 background-color:rgba(128,128,128,0.2);
11052 font-family:Georgia;
11053 }
11054 .VirtualBrowserCommandBar {
11055 position:absolute;
11056 z-index:200000;
11057 xxxdisplay:inline;
11058 display:block;
11059 width:60px;
11060 height:20px;
11061 line-height:1.1;
11062 vertical-align:middle;
11063 font-size:14px;
11064 color:#ff4;
11065 background-color:rgba(128,128,128,0.1);
11066 font-family:Georgia;
11067 text-align:left;
11068 left:404px;
11069 }
11070 .VirtualBrowserFrame {
11071 xposition:relative;
11072 position:absolute;
11073 xxxdisplay:inline;
11074 display:block;
11075 z-index:10;
11076 resize:both !important;
11077 width:480px; height:240px;
11078 min-width:60px; min-height:30px;
11079 max-width:5120px; max-height:2880px;
11080 border-radius:6px;
11081 background-color:rgba(255,255,255,0.9);
11082 border-top:20px solid;
11083 border-right:4px solid;
11084 border-bottom:10px solid;
11085 }
11086 .WinFavicon {
11087 width:16px;
11088 height:16px;
11089 margin:1px;
11090 vertical-align:middle;
11091 background-color:rgba(255,255,255,1.0);
11092 }
11093 .VirtualDesktopMenuBar {
11094 xposition:absolute;
11095 color:#fff;
11096 font-size:7pt;
11097 text-align:right;
11098 padding-right:4px;
11099 background-color:rgba(128,128,128,0.7);
11100 }
11101 .VirtualDesktopCalender {
11102 color:#fff;
11103 font-size:22pt;
11104 text-align:right;
11105 padding-right:4px;
11106 xbackground-color:rgba(255,255,255,0.2);
11107 }
11108 .xxxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
11109 display:inline !important; font-size:10pt !important; padding:1px !important;
11110 }
11111 .WD_Config {
11112 display:inline !important;
11113 padding:2px !important;
11114 font-size:10pt !important;
11115 width:60px !important;
11116 height:12pt !important;
11117 line-height:1.0pt !important;
11118 height:15pt !important;
11119 }
11120 .WD_Button {
11121 display:inline !important;
11122 padding:2px !important;
11123 color:#fff !important;
11124 background-color:#228 !important;
11125 font-size:10pt !important;
11126 width:60px !important;
11127 height:12pt !important;
11128 line-height:1.0pt !important;
11129 height:16pt !important;
11130 border:2px inset #44a !important;
11131 }
11132 .WD_Href {
11133 display:inline !important;
11134 padding:2px !important;
11135 font-size:9pt !important;
11136 width:120px !important;
11137 height:12pt !important;
11138 line-height:1.0pt !important;
11139 height:15pt !important;
11140 }
11141 .LiveHtmlCodeviewText {
11142 font-size:10pt;
11143 font-family:Courier New;
11144 white-space:pre;
11145 }
11146 .WD_Panel {
11147 x-index:100 !important;

```

```

11151 color:#000 !important;
11152 margin-left:25px !important;
11153 width:800px !important;
11154 padding:4px !important;
11155 border:1px solid #888 !important;
11156 border-radius:6px !important;
11157 background-color:rgba(220,220,220,0.9) !important;
11158 font-size:9pt;
11159 font-family:Courier New;
11160 }
11161 .WD_Help {
11162 font-size:10pt !important;
11163 font-family:Courier New;
11164 line-height:1.2 !important;
11165 color:#000 !important;
11166 width:100% !important;
11167 background-color:rgba(240,240,255,0.8) !important;
11168 }
11169
11170 .WB_Zoom {
11171 }
11172 }
11173 }
11174 }
11175 }
11176 }
11177 }
11178 }
11179 }
11180 }
11181 }
11182 }
11183 }
11184 }
11185 }
11186 }
11187 }
11188 }
11189 }
11190 }
11191 }
11192 }
11193 }
11194 }
11195 }
11196 }
11197 }
11198 }
11199 }
11200 }
11201 }
11202 }
11203 }
11204 }
11205 }
11206 }
11207 }
11208 }
11209 }
11210 }
11211 }
11212 }
11213 }
11214 }
11215 }
11216 }
11217 }
11218 }
11219 }
11220 }
11221 }
11222 }
11223 }
11224 }
11225 }
11226 }
11227 }
11228 }
11229 }
11230 }
11231 }
11232 }
11233 }
11234 }
11235 }
11236 }
11237 }
11238 }
11239 }
11240 }
11241 }
11242 }
11243 }
11244 }
11245 }
11246 }
11247 }
11248 }
11249 }
11250 }
11251 }
11252 }
11253 }
11254 }
11255 }
11256 }
11257 }
11258 }
11259 }
11260 }
11261 }
11262 }
11263 }
11264 }
11265 }
11266 }
11267 }
11268 }
11269 }
11270 }
11271 }
11272 }
11273 }
11274 }
11275 }
11276 }
11277 }
11278 }
11279 }
11280 }
11281 }
11282 }
11283 }
11284 }
11285 }
11286 }
11287 }
11288 }
11289 }
11290 }
11291 }
11292 }
11293 }
11294 }
11295 }
11296 }
11297 }
11298 }
11299 }
11300 }
11301 }
11302 }
11303 }
11304 }
11305 }
11306 }
11307 }
11308 }
11309 }
11310 }
11311 }
11312 }
11313 }
11314 }
11315 }
11316 }
11317 }
11318 }
11319 }
11320 }
11321 }
11322 }
11323 }
11324 }
11325 }
11326 }
11327 }

```



```

11328 //WinDragstartTY = t.getBoudingClientRect().top.toFixed(0)
11329 if( t.style.left == '' ){
11330   WinDragstartTX = x0 = 0;
11331   t.style.left = '0px';
11332 }else{
11333   //WinDragstartTX = x0 = Number(t.style.left);
11334   WinDragstartTX = x0 = parseInt(t.style.left);
11335 }
11336 if( t.style.top == '' ){
11337   WinDragstartTY = y0 = 0;
11338   t.style.top = '0px';
11339 }else{
11340   //WinDragstartTY = y0 = Number(t.style.top);
11341   WinDragstartTY = y0 = parseInt(t.style.top);
11342 }
11343 if( true ){ // to be undo
11344   t.wasAtX = WinDragstartTX;
11345   t.wasAtY = WinDragstartTY;
11346 }
11347 wlog('DragSTA #' + t.id
11348       + event('e.x+', 'e.y+')
11349       + ' position=' + t.style.position
11350       + ' style left,top'+t.style.left+', '+t.style.top+' )'
11351 );
11352 e.stopPropagation();
11353 //e.preventDefault();
11354 return true;
11355 }
11356 function onWinDragEvent(wh,e,set,dolog){
11357   t = e.target;
11358   dx = e.x - WinDragstartX;
11359   dy = e.y - WinDragstartY;
11360   nx = WinDragstartTX + dx;
11361   ny = WinDragstartTY + dy;
11362   log = 'Drag '+wh+' #' + t.id
11363       + ' event0('+WinDragstartX+', '+WinDragstartY+')'
11364       + ' event('+e.x+', '+e.y+')'
11365       + ' diff('+dx+', '+dy+')'
11366       + ' (' + nx + ' , ' + ny + ' , '
11367       + ' (' + t.style.left + ' , ' + t.style.top + ' )'
11368       + ' wasAt(' + t.wasAtX + ' , ' + t.wasAtY + ' )'
11369 );
11370 if( e.x != 0 || e.y != 0 ){
11371   if( set == true ){
11372     //t.style.x = nx + 'px'; // not effective
11373     //t.style.y = ny + 'px'; // not effective
11374     t.style.left = nx + 'px';
11375     t.style.top = ny + 'px';
11376     log += ' Set';
11377   }else{
11378     log += ' NotSet';
11379     if( !dolog ){
11380       log = '';
11381     }
11382   }
11383 }else{
11384   log += ' What?'; // the type is event start?
11385   if( !dolog ){
11386     log = '';
11387   }
11388 }
11389 if( and(dolog, log != '' ) ){
11390   wlog(log);
11391 }
11392 if( true ){
11393   // should be propagated to parent in FireFox ?
11394   e.stopPropagation();
11395 }
11396 e.preventDefault();
11397 return false;
11398 }
11399 function onWinDrag(e){
11400   return onWinDragEvent('Ing',e,true,false);
11401 }
11402 function onWinDragend(e){
11403   return onWinDragEvent('End',e,false,true);
11404 }
11405 function onWinDragexit(e){
11406   return onWinDragEvent('Exit',e,false,true);
11407 }
11408 function onWinDragover(e){
11409   return onWinDragEvent('Over',e,false,true);
11410 }
11411 function onWinDragenter(e){
11412   return onWinDragEvent('Enter',e,false,true);
11413 }
11414 function onWinDragleave(e){
11415   return onWinDragEvent('Leave',e,false,true);
11416 }
11417 function onWinDragdrop(e){
11418   return onWinDragEvent('Drop',e,false,true);
11419 }
11420 function onFaviconChange(e){
11421   wlog('--Favicon #' + e.target.id + ' href='+e.details.href);
11422 }
11423 var savedSuppressGJShell = false;
11424 function stopGShell(e){
11425   //wlog('enter Gsh STOP');
11426   savedSuppressGJShell = SuppressGJShell;
11427   SuppressGJShell = true;
11428   e.stopPropagation();
11429   e.preventDefault();
11430 }
11431 function contGShell(e){
11432   //wlog('leave Gsh STOP');
11433   SuppressGJShell = savedSuppressGJShell;
11434   e.stopPropagation();
11435   e.preventDefault();
11436 }
11437 function WD_onKeydown(e){
11438   keycode = e.code;
11439   console.log('Keydown #' + e.target.id + ' ' + keycode);
11440   if( keycode == 'KeyG' ){
11441     WD_setGrid1(WD_Grid_1);
11442   }else
11443   if( keycode == 'KeyI' ){
11444     WD_doZoomIN();
11445   }else
11446   if( keycode == 'KeyO' ){
11447     WD_doZoomOUT();
11448   }else
11449   if( keycode == 'KeyR' ){
11450     WD_RestoreScope(null);
11451   }else
11452   if( keycode == 'KeyS' ){
11453     WD_SaveScope(null);
11454   }
11455   e.stopPropagation();
11456   e.preventDefault();
11457 }
11458 function WD_onKeyUp(e){
11459   e.stopPropagation();
11460   e.preventDefault();
11461 }
11462 function WD_EventSetup1(){
11463   WirtualDesktop_1.addEventListener('keydown', e => { WD_onKeydown(e); });
11464   WirtualDesktop_1_Content.addEventListener('keydown', e => { WD_onKeydown(e); });
11465   WD_Help_1.addEventListener('keydown', e => { WD_onKeydown(e); });
11466   WD_Help_1.addEventListener('keyup', e => { WD_onKeyUp(e); });
11467 }
11468 function settleWin(s,l,cmd,f,u,w,h,x,y,c,scale){
11469   function WirtualBrowserCommand(e,s,l,cmd,f){
11470     command = cmd.innerHTML;
11471     if( command == "Reload" ){
11472       href_id = e.target.href_id;
11473       d = document.getElementById(href_id);
11474       wlog('href_tag#' + href_id + '\n elem#' + href_id + '\n href='+d);
11475       url = d.innerHTML;
11476       wlog('---- Load href_tag#' + href_id + '\n elem#' + href_id + '\n href='+d
11477           + '\n url=' + url);
11478       wlog('---- Load target #' + f.id + ' with url=' + url);
11479       f.src = url;
11480     }else{
11481       alert('unknown command'+command+' '+e.target.id+', '+l.id+', '+f.id);
11482     }
11483   }
11484   function onKeyDown(e){
11485     if( e.code == 'Enter' ){
11486       e.stopPropagation();
11487       e.preventDefault();
11488     }
11489   }
11490   function onKeyUp(e){
11491     if( e.code == 'Enter' ){
11492       e.stopPropagation();
11493       e.preventDefault();
11494       // should reload immediately ?
11495     }
11496   }
11497   if( false ){
11498     wlog('start settle WirtualBrowser url='+u + '\n'
11499         + ' id=' + s.id + '\n'
11500         + ' width=' + s.style.width + '\n'
11501         + ' height=' + s.style.height
11502     );
11503   }
11504 }

```

```

11505 }
11506 // very important for WordPress ??
11507 s.style.width = f.style.width = 501; // for WordPress ...??
11508 s.style.height = f.style.height = 271; // for WordPress ...??
11509 if( false ){
11510     wdlol('midway settle VirtualBrowser url='+u+'\n'
11511         + 'id=' + s.id + '\n'
11512         + 'width=' + s.style.width + '\n'
11513         + 'height=' + s.style.height
11514     );
11515 }
11516 s.width = 502; // for WordPress ...??
11517 s.height = 272; // for WordPress ...??
11518 if( false ){
11519     wdlol('midway-2 settle VirtualBrowser url='+u+'\n'
11520         + 'id=' + s.id + '\n'
11521         + 'span-width=' + s.width + '\n'
11522         + 'span-height=' + s.height
11523     );
11524 }
11525
11526 s.style.width = w + 'px';
11527 s.style.height = h + 'px';
11528 f.style.width = w + 'px';
11529 f.style.height = h + 'px';
11530 //f.style.setProperty('-webkit-transform', 'scale('+scale+')');
11531 f.style.setProperty('transform', 'scale('+scale+')');
11532
11533 //wlog('--x1-- u='+u+' width s='+s.style.width+',f='+f.style.width);
11534 //wlog('--x2-- u='+u+' width s='+s.style.width+',f='+f.style.width);
11535 s.setAttribute('draggable', 'true');
11536 f.setAttribute('draggable', 'false'); // why necessary?
11537 l.setAttribute('draggable', 'false'); // why necessary?
11538 cmd.setAttribute('draggable', 'false'); // why necessary?
11539 s.addEventListener('dragstart', e => { onWinDragstart(e); });
11540 s.addEventListener('drag', e => { onWinDrag(e); });
11541 s.addEventListener('exit', e => { onWinDragexit(e); });
11542 s.addEventListener('dragend', e => { onWinDragend(e); });
11543 s.addEventListener('dragexit', e => { onWinDragexit(e); });
11544 s.addEventListener('dragenter', e => { onWinDragenter(e); });
11545 s.addEventListener('dragover', e => { onWinDragover(e); });
11546 s.addEventListener('dragleave', e => { onWinDragleave(e); });
11547 s.addEventListener('drop', e => { onWinDragdrop(e); });
11548
11549 s.addEventListener('mouseenter', e => { onEnterWin(e); });
11550 s.addEventListener('mouseleave', e => { onLeaveWin(e); });
11551
11552 if( false ){
11553     s.style.position = "absolute";
11554     s.style.x = x+'px';
11555     s.style.left = x+'px';
11556     s.style.y = y+'px';
11557     s.style.top = y+'px';
11558 }else{
11559     s.style.setProperty('position', 'absolute', 'important');
11560     s.style.setProperty('x', x+'px', 'important');
11561     s.style.setProperty('left', x+'px', 'important');
11562     s.style.setProperty('y', y+'px', 'important');
11563     s.style.setProperty('top', y+'px', 'important');
11564 }
11565
11566 favicon = './favicon.ico';
11567 uv1 = u.split('/');
11568 if( 2 <= uv1.length ){
11569     uv2 = uv1[1].split('/');
11570     if( 2 <= uv2.length ){
11571         if( uv1[0] == 'file' ){
11572             //favicon = file:/// + uv2.slice(0,uv2.length-1).join('/');
11573             // + '/favicon.ico';
11574             favicon = './favicon.ico';
11575         }else{
11576             favicon = uv1[0] + '://' + uv2[0] + '/favicon.ico';
11577         }
11578     }
11579 }
11580 //wlog("---- favicon-url="+favicon);
11581 href_id = l.id + ' href';
11582 l.innerHTML = ''
11583 + '<a id="'+href_id+'" class="VirtualBrowserLocation" href="'+u+">'+u+'</a>';
11584 //l.addEventListener('click', e => { onClickWin(e); });
11585 l.addEventListener('mouseenter', e => { stopGShell(e); });
11586 l.addEventListener('mouseleave', e => { onGShell(e); });
11587 l.addEventListener('keydown', e => { onKeyDown(e); });
11588 l.addEventListener('keyup', e => { onKeyUp(e); });
11589
11590 cmd.href_id = href_id;
11591 wdlol('()0)cmd=#'+cmd.id);
11592 wdlol('(1)href_id=#'+href_id);
11593 wdlol('(2)href_id=#'+cmd.href_id);
11594 cmd.addEventListener('click', e => { VirtualBrowserCommand(e,s,l,cmd,f); });
11595
11596 f.style.borderColor = c;
11597 f.src = u;
11598 //f.addEventListener('mouseenter', e => { onEnterWin(e); });
11599 //f.addEventListener('mouseleave', e => { onLeaveWin(e); });
11600
11601 //s.addEventListener('click', e => { onClickWin(e); });
11602 //f.addEventListener('click', e => { wdlol('click wbi'); });
11603 f.addEventListener('mozbrowsericonchange', onFaviconChange);
11604
11605 wdlol('done settle VirtualBrowser url='+u+'\n'
11606     + 'id=' + s.id + ' '
11607     + 'width=' + s.style.width + ' '
11608     + 'height=' + s.style.height + ' '
11609     + 'cmd=' + cmd.id
11610 );
11611 }
11612 }
11613
11614 function WD_EventSetup2(){
11615     dt = VirtualDesktop;
11616     dt.style.width = "800px";
11617     dt.style.height = "500px";
11618     dt.addEventListener('dragstart', e => { onWinDragstart(e); });
11619     dt.addEventListener('drag', e => { onWinDrag(e); });
11620     dt.addEventListener('exit', e => { onWinDragexit(e); });
11621 }
11622
11623 function GRonClick(){
11624     WD_SaveScope(null); // should be push
11625     t = event.target;
11626     x = t.getAttribute('data-leftx');
11627     y = t.getAttribute('data-topy');
11628     zoom = WD_Zoom_1_XY.value;
11629     x *= zoom;
11630     y *= zoom;
11631     WD_doScrollXY(event,x,y);
11632     //alert('scroll #' + t.id + ' x='+x+', y='+y);
11633 }
11634
11635 function WD_setGrid(e){
11636     t = e.target;
11637     WD_setGrid1(t);
11638 }
11639
11640 function WD_setGrid1(t){
11641     //ds = VirtualDesktop_top_1_Content; // should be VirtualSpace_1
11642     ds = VirtualDesktop_top_1_GridPlane;
11643     if( t.value == 'GridOn' ){
11644         for( col = 0; col < 16; col++ ){
11645             for( row = 0; row < 16; row++ ){
11646                 gl = document.createElement('span');
11647                 gl.setAttribute('class', 'VirtualGrid');
11648                 leftx = col * 800;
11649                 topy = row * 500;
11650                 gid = col + ' ' + row
11651                 label = '<'+span'
11652                     + 'id="'+gid+' '+ 'class="WD_GridScroll'
11653                     + ' contenteditable="false" onclick="GRonClick()'
11654                     + ' data-leftx=' + leftx + ' ' + ' data-topy=' + topy + ' '
11655                     + '>';
11656                 + gid + '</span>';
11657                 console.log('grid '+label);
11658                 gl.innerHTML = label;
11659                 gl.position = 'relative';
11660                 gl.leftx = leftx;
11661                 gl.topy = topy;
11662                 gl.style.left = gl.leftx + 'px';
11663                 gl.style.top = gl.topy + 'px';
11664                 if( col % 2 == row % 2 ){
11665                     gl.style.backgroundColor = 'rgba(255,255,255,0.3)';
11666                 }
11667                 ds.appendChild(gl);
11668             }
11669         }
11670         t.value = 'GridOff';
11671     }else{
11672         ds.innerHTML = '';
11673         t.value = 'GridOn';
11674     }
11675 }
11676
11677 var sav_scrollLeft;
11678 var sav_scrollTop;
11679 var sav_nscaie;
11680
11681 function WD_SaveScope(e){
11682     sav_scrollLeft = WD_Left_1.value;
11683     sav_scrollTop = WD_Top_1.value;
11684     sav_nscaie = WD_Zoom_1_XY.value;
11685     //console.log('saved zoom='+sav_oscaie+', '+sav_nscaie);

```

```

11682}
11683function WD_RestoreScope(e){
11684    WD_Zoom_1_XY.value = sav_nscale;
11685    WD_doZoom();
11686}
11687WD_Left_1.value = sav_scrollLeft;
11688WD_Top_1.value = sav_scrollTop;
11689WD_doScroll(null);
11690}
11691function ignoreEvent(e){
11692    e.stopPropagation();
11693    //e.preventDefault();
11694}
11695function zoomMag(){
11696    return WD_Zoom_1_MAG.value;
11697}
11698function WD_EventSetup3(){
11699    WD_Grid_1.addEventListener('click', e => { WD_setGrid(e); });
11700    WD_Zoom_1_Save.addEventListener('click', e => { WD_SaveScope(e); });
11701    WD_Zoom_1_Restore.addEventListener('click', e => { WD_RestoreScope(e); });
11702    WD_Width_1.value = dt.style.width;
11703    WD_Width_1.addEventListener('keydown', ignoreEvent);
11704    WD_Width_1.addEventListener('keyup', ignoreEvent);
11705    WD_Height_1.value = dt.style.height;
11706    WD_Height_1.addEventListener('keydown', ignoreEvent);
11707    WD_Height_1.addEventListener('keyup', ignoreEvent);
11708    WD_Zoom_1_MAG.addEventListener('keydown', ignoreEvent);
11709    WD_Zoom_1_MAG.addEventListener('keyup', ignoreEvent);
11710}
11711function escale(e,oscale,nscale){
11712    e.style.setProperty('transform','scale('+nscale+')');
11713    rscale = oscale / nscale;
11714    w = parseInt(e.style.width);
11715    h = parseInt(e.style.height);
11716    w = w * rscale; //(oscale/nscale);
11717    h = h * rscale; //(oscale/nscale);
11718    e.style.width = w + 'px';
11719    e.style.height = h + 'px';
11720}
11721function scaleWD(ds,oscale,nscale){
11722    if( true ){
11723        escale(WirtualBrowser_1,oscale,nscale);
11724        escale(WirtualBrowser_1_Location,oscale,nscale);
11725        escale(WirtualBrowser_1_Command,oscale,nscale);
11726        escale(WirtualBrowser_1_Frame,oscale,nscale);
11727    }
11728    escale(WirtualBrowser_2,oscale,nscale);
11729    escale(WirtualBrowser_2_Location,oscale,nscale);
11730    escale(WirtualBrowser_2_Command,oscale,nscale);
11731    escale(WirtualBrowser_2_Frame,oscale,nscale);
11732    escale(WirtualBrowser_3,oscale,nscale);
11733    escale(WirtualBrowser_3_Location,oscale,nscale);
11734    escale(WirtualBrowser_3_Command,oscale,nscale);
11735    escale(WirtualBrowser_3_Frame,oscale,nscale);
11736}
11737}
11738}
11739function WD_doZoom(){
11740    ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
11741    oscale = WD_Zoom_1_XY.ovalue; // hidden value for current zoom
11742    nscale = WD_Zoom_1_XY.value;
11743    ds.style.zoom = nscale;
11744    WD_Zoom_1_XY.value = ds.style.zoom;
11745    WD_Zoom_1_XY.ovalue = ds.style.zoom;
11746    scaleWD(ds,oscale,nscale);
11747}
11748}
11749function WD_EventSetup4(){
11750    WD_Zoom_1.addEventListener('click',WD_doZoom);
11751    WD_Zoom_1_XY.addEventListener('keydown',ignoreEvent);
11752    WD_Zoom_1_XY.addEventListener('keyup',ignoreEvent);
11753}
11754}
11755function WD_doZoomOUT(){
11756    ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
11757    oscale = WD_Zoom_1_XY.value;
11758    if( oscale == 0 || oscale == '' ){
11759        oscale = 1;
11760    }
11761    nscale = oscale / zoomMag();
11762    ds.style.zoom = nscale;
11763    WD_Zoom_1_XY.value = ds.style.zoom;
11764    WD_Zoom_1_XY.ovalue = ds.style.zoom;
11765    scaleWD(ds,oscale,nscale);
11766}
11767}
11768function WD_doZoomIN(){
11769    ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
11770    oscale = WD_Zoom_1_XY.value;
11771    if( oscale == 0 || oscale == '' ){
11772        oscale = 1;
11773    }
11774    nscale = oscale * zoomMag();
11775    if( 4 < nscale ){
11776        alert('maybe too large, zoom='+nscale);
11777        return;
11778    }
11779    ds.style.zoom = nscale;
11780    WD_Zoom_1_XY.value = ds.style.zoom;
11781    scaleWD(ds,oscale,nscale);
11782}
11783}
11784function WD_EventSetup5(){
11785    WD_Zoom_1_OUT.addEventListener('click',WD_doZoomOUT);
11786    WD_Zoom_1_IN.addEventListener('click',WD_doZoomIN);
11787}
11788}
11789function WD_doResize(e){
11790    dt = WirtualDesktop_1;
11791    dt.style.width = WD_Width_1.value;
11792    dt.style.height = WD_Height_1.value;
11793    WD_Width_1.value = dt.style.width;
11794    WD_Height_1.value = dt.style.height;
11795}
11796}
11797}
11798function WD_doRSResize(e){
11799    //alert('Resize Space');
11800    ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
11801    ds.style.width = WS_1_Width.value;
11802    ds.style.height = WS_1_Height.value;
11803    WS_1_Width.value = ds.style.width;
11804    WS_1_Height.value = ds.style.height;
11805}
11806}
11807function WD_EventSetup6(){
11808    ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
11809    ds.style.width = '5120px';
11810    ds.style.height = '2880px';
11811    WS_1_Width.value = ds.style.width;
11812    WS_1_Height.value = ds.style.height;
11813    WS_1_Width.addEventListener('keydown', ignoreEvent);
11814    WS_1_Height.addEventListener('keydown', ignoreEvent);
11815    WS_1_Width.addEventListener('keyup', ignoreEvent);
11816    WS_1_Height.addEventListener('keyup', ignoreEvent);
11817    WS_Resize_1.addEventListener('click',e => { WD_doRSResize(e); });
11818}
11819}
11820function WD_doScrollXY(e,sleft,stop){
11821    dt = WirtualDesktop_1;
11822    dt.scrollLeft = sleft;
11823    dt.scrollTop = stop;
11824    WD_Left_1.value = dt.scrollLeft;
11825    WD_Top_1.value = dt.scrollTop;
11826    console.log('--Scroll #' + dt.id + ' (' + sleft + ', ' + stop + ')');
11827}
11828}
11829function WD_doScroll(e){
11830    //dt = WirtualDesktop_1_Content;
11831    dt = WirtualDesktop_1;
11832    sleft = parseInt(WD_Left_1.value);
11833    stop = parseInt(WD_Top_1.value);
11834    dt.scrollLeft = sleft;
11835    dt.scrollTop = stop;
11836    WD_Left_1.value = dt.scrollLeft;
11837    WD_Top_1.value = dt.scrollTop;
11838    console.log('--Scroll #' + dt.id + ' (' + sleft + ', ' + stop + ')');
11839}
11840}
11841function showScrollPosition(){
11842    if( false )
11843        console.log(
11844            'wtop=' + WirtualDesktop_1.style.top + ', ' +
11845            'wxs=' + WirtualDesktop_1.style.x + ', ' +
11846            'wss=' + WirtualDesktop_1.scrollLeft + ', ' +
11847            'wdtop=' + WirtualDesktop_1_Content.style.top + ', ' +
11848            'wds=' + WirtualDesktop_1_Content.style.y + ', ' +
11849            'wds=' + WirtualDesktop_1_Content.scrollLeft + ', '
11850        );
11851    WD_Left_1.value = WirtualDesktop_1.scrollLeft;
11852    WD_Top_1.value = WirtualDesktop_1.scrollTop;
11853}
11854}
11855function WD_EventSetup7(){
11856    WD_Scroll_1.addEventListener('click',e => { WD_doScroll(e); });
11857    WD_Left_1.addEventListener('keydown',ignoreEvent);
11858    WD_Left_1.addEventListener('keyup',ignoreEvent);
11859    WD_Top_1.addEventListener('keydown',ignoreEvent);
11860    WD_Top_1.addEventListener('keyup',ignoreEvent);
11861}
11862}
11863function WD_EventSetup8(){

```

```

11859 WirtualDesktop_1.addEventListener('scroll',showScrollPosition);
11860 WirtualDesktop_1_Content.addEventListener('scroll',showScrollPosition);
11861 }
11862
11863 if( false ){
11864   w = 1000 + 'px';
11865   dt.style.width = w;
11866   dt.style.height = "300px";
11867   dt.style.resize = 'both';
11868   dt.style.borderWidth = 50 + 'px';
11869   dt.style.borderRadius = 25 + 'px';
11870   console.log('---2----- #' + dt.id + ' style=' + dt.style);
11871   console.log('----- #' + dt.id + ' width=' + dt.style.width);
11872   console.log('----- #' + dt.id + ' left=' + dt.style.left);
11873   console.log('----- #' + dt.id + ' border=' + dt.style.border);
11874 }
11875 function onDTResize(e){
11876   dt = e.target;
11877   h = parseInt(dt.style.height);
11878   dt.style.borderWidth = (h * 0.075) + 'px';
11879   console.log('----- borderWidgh=' + dt.style.borderWidth);
11880 }
11881
11882 WirtualDesktopDetails.addEventListener('toggle',WirtualDesktop_init);
11883 function WirtualDesktop_init(){
11884   if( !WirtualDesktopDetails.open ){
11885     return;
11886   }
11887   //GJ_Join();
11888   WirtualDesktop_1.addEventListener('resize', e => { onDTResize(e); });
11889   //console.log('----- #' + dt.id
11890   // ' borderWidgh=' + dt.style.getProperty('border-width'));
11891
11892   settleWin(
11893     WirtualBrowser_1,
11894     WirtualBrowser_1_Location,
11895     WirtualBrowser_1_Command,
11896     WirtualBrowser_1_Frame,
11897     document.URL,
11898     500,280,50,20,'#262',1.0);
11899   settleWin(
11900     WirtualBrowser_2,
11901     WirtualBrowser_2_Location,
11902     WirtualBrowser_2_Command,
11903     WirtualBrowser_2_Frame,
11904     'https://its-more.jp/1a.jp/',
11905     500,280,150,100,'#448',1.0);
11906   settleWin(
11907     WirtualBrowser_3,
11908     WirtualBrowser_3_Location,
11909     WirtualBrowser_3_Command,
11910     WirtualBrowser_3_Frame,
11911     '../gshell/gsh-go.html',
11912     'http://gshell.org/gshell/gsh-go.html',
11913     'https://golang.org',
11914     500,280,250,180,'#444',1.0);
11915     //1000,720,0,0,'#444',0.125);
11916     //1200,720,-100,-50,'#444',0.4);
11917   function WD_ClockUpdate(e){
11918     WirtualDesktop_1_Clock.innerHTML = DateShort();
11919     WirtualDesktop_1_Calender.innerHTML = DateHourMin();
11920   }
11921   window.setInterval(WD_ClockUpdate,500);
11922
11923   WD_EventSetup1();
11924   WD_EventSetup2();
11925   WD_EventSetup3();
11926   WD_EventSetup4();
11927   WD_EventSetup5();
11928   WD_EventSetup6();
11929   WD_EventSetup7();
11930   WD_EventSetup8();
11931 }
11932 //WirtualDesktop_init();
11933
11934 Destroy_WirtualDesktop = function(){
11935   WirtualDesktop_1.style = "";
11936
11937   WirtualBrowser_1.removeAttribute('style');
11938   WirtualBrowser_1_Location.innerHTML = '';
11939   WirtualBrowser_1_Frame.removeAttribute('src');
11940   WirtualBrowser_1_Frame.removeAttribute('style');
11941   WirtualBrowser_1_Frame.style="";
11942
11943   WirtualBrowser_2.removeAttribute('style');
11944   WirtualBrowser_2_Location.innerHTML = '';
11945   WirtualBrowser_2_Frame.removeAttribute('src');
11946   WirtualBrowser_2_Frame.style="";
11947
11948   WirtualBrowser_3.removeAttribute('style');
11949   WirtualBrowser_3_Location.innerHTML = '';
11950   WirtualBrowser_3_Frame.removeAttribute('src');
11951   WirtualBrowser_3_Frame.style="";
11952
11953   GFactory_1.style = "";
11954   iframe0.style = "";
11955   WirtualDesktop_1.style = "";
11956 }
11957
11958 </script>
11959 <!-- WirtualDesktop -->
11960 </details>
11961 *! //</span>
11962
11963 <!-- ===== Work { ===== -->
11964 </span id="SBSidebar_WorkCodeSpan">
11965 /
11966 <details><summary>SBSidebar</summary>
11967 <!-- ===== SBSidebar // 2020-0928 SatoxITS { -->
11968 <h2>SBSidebar</h2>
11969 <input id="SBSidebar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
11970 <input id="SBSidebar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
11971 <input id="SBSidebar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
11972 <span id="SBSidebar_WorkCodeView"></span>
11973 <script id="SBSidebar_WorkScript">
11974 function SBSidebar_openWorkCodeView(){
11975   function SBSidebar_showWorkCode(){
11976     showHtmlCode(SBSidebar_WorkCodeView,SBSidebar_WorkCodeSpan);
11977   }
11978   SBSidebar_WorkCodeViewOpen.addEventListener('click',SBSidebar_showWorkCode);
11979 }
11980 SBSidebar_openWorkCodeView(); // should be invoked by an event
11981
11982 console.log('--- SbsSlider // 2020-1006-01 SatoxITS ---');
11983 function SetSidebar(){
11984   sidebar = document.getElementById('secondary');
11985   console.log('primary='+primary+' + secondary='+sidebar+' + main='+main' );
11986   wrap = wrap.parentNode;
11987   console.log('--- SbsSlider parent is '+wrap+', #' + wrap.id + ' .'+wrap.class);
11988   //wrap = wrap.parentNode;
11989   //console.log('--- SbsSlider parent is '+wrap+', #' + wrap.id + ' .'+wrap.class);
11990   //wrap = wrap.parentNode;
11991   //console.log('--- SbsSlider parent is '+wrap+', #' + wrap.id + ' .'+wrap.class);
11992   //nsb = sidebar.cloneNode();
11993   nsb = sidebar;
11994   nsb.style.width = '100%';
11995   slider = document.createElement('div');
11996   slider.id = 'SbsSlider';
11997   slider.appendChild(nsb);
11998   slider.setAttribute('class','SbsSlider');
11999   nsb.style.position = 'relative';
12000   slider.style.position = 'fixed';
12001   slider.style.display = 'block'; // 'inline';
12002   slider.style.zIndex = 100000;
12003   // nsb.style.zIndex = 200000;
12004   nsb.style.position = 'absolute';
12005   nsb.style.minWidth = '80px';
12006   nsb.style.left = '0px';
12007   nsb.style.top = '0px';
12008
12009   w = window.innerWidth;
12010   console.log('sliderWidth '+w+', '(w/3)+'px');
12011   if( w < 640 ){
12012     slider.style.setProperty('width',(w/3) + 'px','important');
12013   }
12014   main.style.marginLeft = (parseInt(slider.style.width)/2 - 20) + 'px';
12015
12016   slider.style.resize = "both";
12017   slider.draggable = "true";
12018   wrap.appendChild(slider);
12019   console.log('--- added SbsSlider');
12020   //nsb.addEventListener('scroll',SbsScrolled);
12021
12022   buttons = document.createElement('div');
12023   buttons.id = 'NaviButtons';
12024   buttons.setAttribute('class','NaviButtons');
12025   buttons.align = "center";
12026   buttons.innerHTML += '< +> <p><a href="#TopOfPost" draggable="true">TOP</a></p>';
12027   buttons.innerHTML += '< +> <p><a href="#EndOfPost" draggable="true">END</a></p>';
12028   page.appendChild(buttons);
12029   buttons.style.position = 'fixed';
12030   buttons.style.zIndex = 30000;
12031   buttons.style.width = '180%';
12032   buttons.style.top = '320px';
12033   buttons.style.left = parseInt(w) * 1.0 + 'px';
12034   console.log('--- SbsSlider installed (~)/ SatoxITS');
12035 }

```

```

12036//window.addEventListener('load',setSidebar); // after load
12037DestroyNavButtons = function(){
12038  nb = document.getElementById('NaviButtons');
12039  if( nb != null ){
12040    nb.parentNode.removeChild(NaviButtons);
12041  }
12042}
12043</script>
12044
12045// 2020-1006 its-more.jp-blog-60000-style.css
12046<!-- {
12047<style>
12048#NaviButtons {
12049  position:fixed;
12050  display:block;
12051  width:100%;
12052  xtop:100px;
12053  xleft:10px;
12054  z-index:30000;
12055  font-size:10pt;
12056  color:#fff !important;
12057  text-align:center;
12058  background-color:rgba(230,230,230,0.01);
12059}
12060#NaviButtons a {
12061  color:#2a2 !important;
12062  font-size:20px;
12063  text-align:center;
12064  xtext-shadow:2px 2px #8ff;
12065  resize:both;
12066  padding:0px;
12067  margin:10px;
12068  border:1px solid #288 !important;
12069  border-radius:3px;
12070  background-color:rgba(160,160,160,0.05);
12071}
12072#SbSlider {
12073  overflow:auto;
12074  resize:both !important;
12075  xoverflow-y:hidden !important;
12076  height:100px !important;
12077  display:inline !important;
12078  position:fixed !important;
12079  left:0px;
12080  top:0px;
12081  xwidth:180px;
12082  width:24%;
12083  min-width:80px;
12084  height:100% !important;
12085  background-color:rgba(100,100,200,0.1);
12086}
12087#secondary {
12088  position:fixed;
12089  left:0px;
12090  top:0px;
12091  xzx-index:60000;
12092  scroll-behavior: overflow !important;
12093  xxxxxxxxxxxxxxxxxxxxxxxxxxxwidth:18% !important;
12094  padding-left:4pt;
12095  color:#fff;
12096  font-size:0.5em;
12097  background-color:rgba(64,160,64,0.6) !important;
12098  white-space:nowrap;
12099}
12100#secondary a {
12101  color:#fff !important;
12102  text-decoration:disable !important;
12103}
12104#primary {
12105  position:relative;
12106  width:75% !important;
12107  left:25% !important;
12108}
12109#main {
12110  position:relative;
12111  width:75% !important;
12112  left:25% !important;
12113}
12114#site-navigation {
12115  position:relative;
12116  left:120px;
12117}
12118#adswc_countertext {
12119  color:#4169e1;
12120  font-size:16pt !important;
12121  xfont-size:10% !important;
12122  font-weight:bold;
12123}
12124#nowTime {
12125  color:#a0ffa0;
12126  font-size:16pt !important;
12127  xfont-size:10% !important;
12128  font-weight:bold;
12129  text-shadow:1px 1px #fff;
12130}
12131.navigation-top {
12132  color:#22a !important;
12133  border:0px;
12134  background-color:rgba(220,220,220,0.1);
12135}
12136
12137.visible-scrollbar, .invisible-scrollbar, .mostly-customized-scrollbar {
12138  display: block;
12139  xxwidth: 1em;
12140  xoverflow: auto;
12141  xxheight: 1em;
12142}
12143.invisible-scrollbar::-webkit-scrollbar {
12144  xdisplay: none;
12145  width:1px !important;
12146  height:1px !important;
12147}
12148.mostly-customized-scrollbar::-webkit-scrollbar {
12149  width: 2px;
12150  height: 2px;
12151  xbackground-color: #aaa; xxx:or add it to the track;
12152}
12153.mostly-customized-scrollbar::-webkit-scrollbar-thumb {
12154  background: #000;
12155}
12156</style>
12157} -->
12158
12159</details>
12160<!-- SBSidebar_WorkCodeSpan } -->
12161<!-- *//</span> //</span>
12162<!-- ===== Work } ===== -->
12163
12164
12165
12166</span id="Affiliate_WorkCodeSpan">
12167*
12168<details id="Affiliate_Test"><summary>Affiliate</summary>
12169<!-- Affiliate // 2020-1010 SatoxITS { -->
12170<div id="AffViewDock">
12171<div id="AffView" class="AffView" draggable="true" style="">
12172<div id="AffiSet" class="AffiPlate">
12173<iframe id="aff_1" class="Affitem"></iframe>
12174<iframe id="aff_2" class="Affitem"></iframe>
12175<iframe id="aff_3" class="Affitem"></iframe>
12176<iframe id="aff_4" class="Affitem"></iframe>
12177<iframe id="aff_5" class="Affitem"></iframe>
12178</div>
12179</div></div>
12180<h2>Supportive Affiliate</h2>
12181<style>
12182.AffView {
12183  z-index:0;
12184  overflow-x:scroll;
12185  overflow-y:scroll;
12186  position:fixed;
12187  max-width:2560px;
12188  max-height:100%;
12189  width:270px;
12190  left:75%;
12191  height:95%;
12192  resize:both;
12193  xleft:-10%;
12194  margin-top:40px;
12195  xleft:0;
12196  xalign:right;
12197  display:block;
12198  border:4px inset rgba(255,255,255,0.1);
12199  background-color:rgba(255,255,255,0.1);
12200}
12201.AffView:hover {
12202  z-index:1;
12203  width:300px;
12204  overflow:scroll;
12205  border:4px inset #fcc;
12206  background-color:rgba(80,80,255,0.2);
12207  background-color:#fcc;
12208}
12209.AffPlate:hover {
12210  border-left:4px dashed #888;
12211}

```

```

12211 }
12212 .AffPlate{
12213   overflow-x:visible;
12214   border-left:4px dashed rgba(255,255,255,0.1);
12215   max-width:2560px;
12216   max-height:2880px;
12217   margin-top:10px;
12218   margin-bottom:10px;
12219   margin-left:4px;
12220   width:300px;
12221   xheight:1440px;
12222 }
12223 .AffItem {
12224   overflow-x:visible;
12225   overflow-y:scroll;
12226   max-width:2560px;
12227   max-height:1440px;
12228   z-index:0;
12229   display:block;
12230   position:fixed;
12231   xposition:absolute;
12232   position:relative;
12233   //left:300px;
12234   xresize:both;
12235   padding:0px;
12236   width:600px;
12237   height:400px;
12238   max-height:800px !important;
12239   margin-top:0%;
12240   margin-left:0%;
12241   margin-right:0% !important;
12242   border:10px inset #ccc;
12243   transform:scale(0.5);
12244   background-color:rgba(255,255,255,0.2);
12245   xalign:right;
12246 }
12247 .AffItem:hover {
12248   border:10px inset #bbf;
12249 }
12250 </style>
12251 <input id="Affiliate_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12252 <input id="Affiliate_WorkCodeViewSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12253 <span id="Affiliate_WorkCodeView"></span>
12254 <script id="Affiliate_WorkScript">
12255 function Affiliate_openWorkCodeView(){
12256   function Affiliate_showWorkCode(){
12257     showHtmlCode(Affiliate_WorkCodeView,Affiliate_WorkCodeSpan);
12258   }
12259   Affiliate_WorkCodeViewOpen.addEventListener('click',Affiliate_showWorkCode);
12260 }
12261 Affiliate_openWorkCodeView(); // should be invoked by an event
12262 </iframe id="aff_8" xxsrc="https://stackoverflow.com/tags" class="AffItem"></iframe>
12263 </iframe id="aff_9" xxsrc="https://developer.mozilla.org/en-US/docs/Web" class="AffItem"></iframe>
12264 var Aff_IsSetup = false;
12265 Affiliate_Reset.addEventListener('click',Aff_Setup);
12266 function Aff_Setup(){
12267   if (Aff_IsSetup) { return; } Aff_IsSetup = true;
12268   parent = document.documentElement;
12269   parent.appendChild(AffView);
12270   AffView.style.top = '0px';
12271   AffView.style.left = (window.innerWidth - 280) + 'px';
12272   var off = 100;
12273   zoom = 0.5;
12274   ozoom = 0.3;
12275   leftx = window.innerWidth - 300;
12276   left = leftx + 'px';
12277   left = -130 + 'px';
12278   console.log('aff-init window.innerWidth='+window.innerWidth);
12279   w = 1000;
12280   h = 560;
12281   aff_0.src = "../gshell/gsh.go.html";
12282   aff_1.src = "https://golang.org";
12283   aff_2.src = "https://drafts.csswg.org/";
12284   aff_3.src = "https://html.spec.whatwg.org/dev/";
12285   aff_4.src = "https://wikipedia.org";
12286   aff_5.src = "https://www.bing.com/translator";
12287 //parent.appendChild(aff_0);
12288 aff_0.style.width = zoom*w+'px';
12289 aff_0.style.height = zoom*h+'px';
12290 aff_0.style.left = left;
12291 //aff_0.style.top = off+'px'; off += ozoom*h;
12292 aff_0.draggable = 'true';
12293 //parent.appendChild(aff_1);
12294 aff_1.style.width = zoom*w+'px';
12295 aff_1.style.height = zoom*h+'px';
12296 aff_1.style.left = left;
12297 //aff_1.style.top = off+'px'; off += ozoom*h;
12298 aff_1.style.top = '-150px';
12299 aff_1.draggable = 'true';
12300 //parent.appendChild(aff_2);
12301 aff_2.style.width = zoom*w+'px';
12302 aff_2.style.height = zoom*h+'px';
12303 aff_2.style.left = left;
12304 //aff_2.style.top = off+'px'; off += ozoom*h;
12305 aff_2.style.top = '-300px';
12306 aff_2.draggable = 'true';
12307 //parent.appendChild(aff_3);
12308 aff_3.style.transform = "scale(0.25)";
12309 aff_3.style.width = 2*zoom*w+'px';
12310 aff_3.style.height = 2*zoom*h+'px';
12311 //aff_3.style.left = -390 + 'px'; //left+2;
12312 //aff_3.style.left = (leftx - 265) + 'px';
12313 //aff_3.style.left = -395 + 'px';
12314 //aff_3.style.top = (-155+off)+'px'; off += ozoom*h;
12315 aff_3.style.top = '-600px';
12316 aff_3.draggable = 'true';
12317 //parent.appendChild(aff_4);
12318 aff_4.style.width = zoom*w+'px';
12319 aff_4.style.height = zoom*h+'px';
12320 aff_4.style.left = left;
12321 //aff_4.style.top = off+'px'; off += ozoom*h;
12322 aff_4.style.top = '-900px';
12323 aff_4.draggable = 'true';
12324 //parent.appendChild(aff_5);
12325 aff_5.style.transform = "scale(0.300)";
12326 aff_5.style.width = zoom*(w*1.67)+ 'px';
12327 aff_5.style.height = zoom*(h*1.67)+ 'px';
12328 aff_5.style.border = "25px inset #ccc";
12329 aff_5.style.left = -308+'px';
12330 //aff_5.style.left = (-175+leftx)+'px';
12331 //aff_5.style.top = -295+off+'px'; off += ozoom*h;
12332 //aff_5.style.left = '0px';
12333 //aff_5.style.top = '0px';
12334 aff_5.style.top = '-1150px';
12335 //aff_5.style.align = 'right';
12336 aff_5.draggable = 'true';
12337 window.addEventListener('resize',affresize);
12338 }
12339 function affresize(){
12340   AffView.style.left = (window.innerWidth - 280) + 'px';
12341   leftx = window.innerWidth - 400;
12342   left = leftx + 'px';
12343   console.log('aff-resize window.innerWidth='+window.innerWidth);
12344 }
12345 //window.addEventListener('resize',affresize);
12346 //document.addEventListener('resize',affresize);
12347 //gsh.addEventListener('resize',affresize);
12348 function ResetAffView(){
12349   AffViewDock.appendChild(AffView);
12350   AffView.removeAttribute('style');
12351   aff_0.removeAttribute('src');
12352   aff_0.removeAttribute('style'); aff_0.removeAttribute('draggable');
12353   aff_1.removeAttribute('src');
12354   aff_1.removeAttribute('style'); aff_1.removeAttribute('draggable');
12355   aff_2.removeAttribute('src');
12356   aff_2.removeAttribute('style'); aff_2.removeAttribute('draggable');
12357   aff_3.removeAttribute('src');
12358   aff_3.removeAttribute('style'); aff_3.removeAttribute('draggable');
12359   aff_4.removeAttribute('src');
12360   aff_4.removeAttribute('style'); aff_4.removeAttribute('draggable');
12361   aff_5.removeAttribute('src');
12362   aff_5.removeAttribute('style'); aff_5.removeAttribute('draggable');
12363 }
12364 </script>
12365 </details>
12366 <!-- Affiliate_WorkCodeSpan -->
12367 </span>
12368 </!-- Work -->
12369 </span id="TextCanvas_WorkCodeSpan">
12370 *

```

```

1239<details id="TextCanvas_Section"><summary id="TextCanvas_Summary">TextCanvas</summary>
1239</details><!-- TextCanvas // 2020-1013 SatoxITS { -->
1239
1239<details id="FontSelect_Section"><summary id="FontSelect_Summary">Font Selection</summary>
1239<h2>Font Selection</h2>
1239<div>
1239<div id="FontList"></div>
1239</div>
1239<style>
1239#FontList {
12400 overflow:visible;
12401 background-color:rgba(240,245,255,1.0) !important;
12402 }
12403#FontList td {
12404 font-size:16px;
12405 padding:0px;
12406 padding-left:2px;
12407 padding-right:2px;
12408 margin:0px;
12409 line-height:1.2;
12410 border:0px;
12411 }
12412#FontList td:hover {
12413 background-color:#228;
12414 }
12415#FontList tr:hover {
12416 color:#fff;
12417 background-color:#000;
12418 xborder:1px solid #000;
12419 }
12420.xcourier { colr:#000; font-size:16px; font-family:courier; }
12421.xcursive { colr:#000; font-size:16px; font-family:cursive; }
12422.xfantasy { colr:#000; font-size:16px; font-family:fantasy; }
12423.xgeorgia { colr:#000; font-size:16px; font-family:georgia; }
12424.xmonospace { colr:#000; font-size:16px; font-family:monospace; }
12425</style>
12426<script>
12427function fontstr(name,text){
12428 //tr = '<+tr style=\ font-family:"'+name+'";\>\n';
12429 tr = '<+tr style=\ font-family:"'+name+'";\>\n';
12430 tr += '<+td style="font-family:Arial;font-size:12pt;">'+name+'</+td>';
12431 tr += '<+td data-fsty="n">'+text+'</+td>';
12432 tr += '<+td data-fsty="b">'+b+'<+td data-fsty="i">'+i+'</+td>';
12433 tr += '<+td data-fsty="i">'+i+'<+td data-fsty="b">'+b+'</+td>';
12434 tr += '<+td data-fsty="bi">'+bi+'<+td data-fsty="ib">'+ib+'</+td>';
12435 tr += '<+tr>';
12436 return tr;
12437 }
12438function lsfont(){
12439 text = 'GShell-Go126#x1f923';
12440
12441 fl = '';
12442 fl += '<table>\n';
12443 fl += fontstr('Arial',text);
12444 fl += fontstr('Courier',text);
12445 fl += fontstr('Courier New',text);
12446 fl += fontstr('Georgia',text);
12447 fl += fontstr('Helvetica',text);
12448 fl += fontstr('Verdana',text);
12449 fl += fontstr('Times',text);
12450
12451 fl += fontstr('Osaka',text);
12452 fl += fontstr('Meiryo',text);
12453 fl += fontstr('YuMincho',text);
12454
12455 //fl += fontstr('Roman',text);
12456 //document.fonts.load("30px cursive");
12457 fl += fontstr('Serif',text);
12458 fl += fontstr('Sans-Serif',text);
12459 fl += fontstr('System-UI',text);
12460 fl += fontstr('Monospace',text);
12461 fl += fontstr('Cursive',text);
12462 fl += fontstr('Fantasy',text);
12463 fl += '</table>\n';
12464
12465 if( false ){
12466 fss = document.fonts.entries(); // FontFaceSet
12467 console.log('FSS= '+fss);
12468 while( true ){
12469 font = fss.next();
12470 if( font.done ){
12471 break;
12472 }
12473 fl += font.value[0] + '<br>';
12474 }
12475 }
12476 FontList.innerHTML = fl;
12477 }
12478function selectFont(e){
12479 t = e.target;
12480 let fsty = '';
12481 for( i = 0; i < 4; i++ ){
12482 //console.log('FontSelect '+t.nodeName+' #' + t.id+' '+t.style);
12483 if( t.nodeName == 'TD' ){
12484 //console.log('FontSelect '+t.outerHTML);
12485 if( t.hasAttribute('data-fsty') ){
12486 fsty = t.getAttribute('data-fsty');
12487 //console.log('FontSelect = ' + fsty);
12488 }
12489 }
12490 if( t.nodeName != 'TD' ){
12491 if( t.style != '' ){
12492 if( t.style.fontFamily != '' ){
12493 break;
12494 }
12495 }
12496 }
12497 t = t.parentNode;
12498 }
12499 if( t.style != '' ){
12500 font = t.style.fontFamily;
12501 //console.log('FontSelect: '+font);
12502 //console.log('FontSelect == ' + fsty);
12503 if( font != '' ){
12504 sel = document.getElementById("TextCanvas_1_Font");
12505 if( sel != null ){
12506 if( fsty != '' ){
12507 TextCanvas_1_Bold.checked = 0 <= fsty.indexOf('b');
12508 TextCanvas_1_Italic.checked = 0 <= fsty.indexOf('i');
12509 }
12510 sel.value = font;
12511 RedrawTextCanvas();
12512 }else{
12513 alert('Event: ' + e.target.nodeName + ' #' + font);
12514 }
12515 }
12516 }
12517 }
12518 FontList.addEventListener('click',selectFont);
12519 document.fonts.onloadingdone = function(fsse){
12520 //alert('font-loaded '+fsse.fontfaces.length);
12521 }
12522 }
12523function FontList_Setup(){
12524 if( FontSelect_Summary.open ){
12525 lsfont();
12526 }
12527 }
12528 FontSelect_Summary.addEventListener('click',lsfont);
12529</script>
12530</details>
12530
12531<h2>Drawing Text on Canvas</h2>
12531<!-- 2020-1012 -- Drawing Text on Canvas // SatoxITS { -->
12531<div id="TextCanvas_1_Panel" class="CanvasLabel">
12532<input id="TextCanvas_1_Clear" class="PanelButton" type="button" value="Clear">
12533<input id="TextCanvas_1_Draw" class="PanelButton" type="button" value="Draw">
12534<input id="TextCanvas_1_Font" class="CanvasPanel" type="text" size="14" value="Georgia">
12535<input id="TextCanvas_1_Bold" class="CanvasBox" type="checkbox" checked="">Bold
12536<input id="TextCanvas_1_Italic" class="CanvasBox" type="checkbox" checked="">Italic
12537<br>
12538<input id="TextCanvas_1_Size" class="CanvasPanel" type="text" size="3" value="64">Pixels
12539<input id="TextCanvas_1_Color" class="CanvasPanel" type="text" size="6" value="#22a">
12540<!-- to be PBlue series ? -->
12541<p>
12542<input id="TextCanvas_1_Text" class="TextCanvasText" type="text" size="50" value="GShell">
12543</p>
12544</div>
12545<p>
12546<canvas id="TextCanvas_1" class="TextCanvas" width="400px" height="100px" draggable="true"></canvas>
12547</p>
12548<div class="CanvasLabel">
12549<input id="TextCanvas_1_ToImage" class="PanelButton" type="button" value="ToImage">
12550<input id="TextCanvas_1_ToPNG" class="PanelRadio" type="radio" name="ImageType" value="ToPNG" checked="">PNG
12551<input id="TextCanvas_1_ToJPEG" class="PanelRadio" type="radio" name="ImageType" value="ToJPEG">JPEG
12552<input id="TextCanvas_1_DataURL" class="CanvasBox" type="checkbox" checked="">DataURL
12553<div class="TextCanvas_1_Image" data-url=""></div>
12554<div id="TextCanvas_1_BgImage" class="CanvasImage"><br></div>
12555<div id="TextCanvas_1_DataURLView" class="DataURLView"><span id="TextCanvas_1_DataURLText"></span></div>
12556</div>
12557<style>
12558.CommandUsageText {
12559 font-family:Courier New;
12560 }
12561.TextCanvas {
12562 border:1px solid #000;

```

```

12567 resize:both;
12568 display:inline !important;
12569 }
12570.DataUrView {
12571 width:100%;
12572 font-size:10pt;
12573 font-family:Courier New, monospace;
12574 color:#000;
12575 xbackground-color:#eee;
12576 margin-bottom:10px;
12577 xdisplay:block;
12578 xoverflow:scroll;
12579 }
12580.CanvasImage {
12581 border:1px dashed #000;
12582 }
12583.TextCanvasText {
12584 font-size:12pt;
12585 width:100%;
12586 }
12587.CanvasLabel {
12588 font-size:10pt;
12589 color:#000;
12590 }
12591.CanvasInImage {
12592 width:100%;
12593 height:160px;
12594 margin-bottom:10px;
12595 color:rgba(32,160,32,0.5);
12596 text-shadow:3px 3px #eee;
12597 background-color:#eee;
12598 xborder:1px solid #000;
12599 font-size:18pt;
12600 vertical-align:middle;
12601 }
12602.PanelRadio {
12603 font-size:12pt !important;
12604 color:#000 !important;
12605 vertical-align:middle;
12606 }
12607.CanvasBox {
12608 vertical-align:middle;
12609 margin-left:4px !important;
12610 margin-right:2px !important;
12611 }
12612.CanvasPanel {
12613 vertical-align:middle !important;
12614 height:14pt !important;
12615 width:inherit !important;
12616 padding:2px !important;
12617 margin:4px !important;
12618 margin-left:4px !important;
12619 margin-right:2px !important;
12620 font-size:10pt !important;
12621 font-family:Georgia !important;
12622 color:#000;
12623 display:inline !important;
12624 }
12625.TextCanvasPanel {
12626 vertical-align:middle;
12627 font-size:10pt !important;
12628 font-family:Georgia !important;
12629 color:#000;
12630 display:inline !important;
12631 }
12632.PanelButton {
12633 font-size:10pt !important;
12634 font-family:Georgia !important;
12635 vertical-align:middle;
12636 width:45pt !important;
12637 height:14pt !important;
12638 line-height:1.2 !important;
12639 padding:2px !important;
12640 margin:1px !important;
12641 display:inline !important;
12642 padding:1px !important;
12643 color:#fff !important;
12644 background-color:#228 !important;
12645 }
12646</style>
12647<script>
12648 function DrawTextCanvas(){
12649   ctx = TextCanvas_1.getContext('2d');
12650   var textFont = '';
12651   if( TextCanvas_1_Italic.checked ) textfont += ' italic';
12652   if( TextCanvas_1_Bold.checked ) textfont += ' bold';
12653   textfont += ' '+TextCanvas_1_Size.value+'px';
12654   textfont += ' '+TextCanvas_1_Font.value;
12655   //ctx.font = 'italic bold 60px Georgia';
12656   //console.log('TxFont='+textfont);
12657   ctx.fillStyle = TextCanvas_1_Color.value; //'#22a';
12658   ctx.strokeStyle = textfont;
12659   ctx.fillText(TextCanvas_1_Text.value,10,80);
12660 }
12661 TextCanvas_1_Draw.addEventListener('click',DrawTextCanvas);
12662 function ClearTextCanvas(){
12663   cv = TextCanvas_1;
12664   ctx = cv.getContext('2d');
12665   ctx.clearRect(0,0,cv.width,cv.height);
12666 }
12667 TextCanvas_1_Clear.addEventListener('click',ClearTextCanvas);
12668 function RedrawTextCanvas(){
12669   ClearTextCanvas();
12670   DrawTextCanvas();
12671 }
12672 function ab2str(buf) {
12673   return String.fromCharCode.apply(null, new Uint16Array(buf));
12674 }
12675 }
12676 // 2020-1024, canvas to image
12677 function CanvasToImage(){
12678   canvas = TextCanvas_1;
12679   // http://developer.mozilla.org/en-US/docs/Web/API/HTMLCanvasElement/toDataURL
12680   if( TextCanvas_1_ToPNG.checked ){
12681     url = canvas.toDataURL("image/png");
12682   }else{
12683     url = canvas.toDataURL("image/jpeg",1.0);
12684   }
12685 }
12686 //alert('CanvasToImage: length='+url.length+'\n'+url);
12687 TextCanvas_1_Image.src = url;
12688 TextCanvas_1_BgImage.style.backgroundImage = 'url('+url+')';
12689 if( TextCanvas_1_DataURL.checked ){
12690   //TextCanvas_1_DataUrView.innerHTML = url;
12691   txa = TextCanvas_1_DataUrText;
12692   utxa = document.createElement('textArea');
12693   utxa.id = 'TextCanvas_1_DataUrText';
12694   utxa.style.width = '100%';
12695   utxa.style.height = '50pt';
12696   utxa.value = url;
12697   txa.parentNode.replaceChild(utxa,txa);
12698 }
12699 return TextCanvas_1_Image;
12700 }
12701 var image = new Image();
12702 image.src = url;
12703 urla = str2ab(url);
12704 blob = new Blob(urla,{type:'text/plain'});
12705 link = document.createElement('a');
12706 link.href = URL.createObjectURL(blob);
12707 link.download = 'character.txt';
12708 link.click();
12709 return image;
12710 }
12711 TextCanvas_1_ToImage.addEventListener('click',CanvasToImage);
12712 }
12713 if( TextCanvas_Section.open ){
12714   DrawTextCanvas();
12715 }
12716 }
12717 TextCanvas_Summary.addEventListener('click',DrawTextCanvas);
12718 </script>
12719 </-->
12720 <script>
12721 //TextCanvas_1_Panel.addEventListener('mouseover',OffGJShell);
12722 //TextCanvas_1_Panel.addEventListener('mouseout',OnGJShell);
12723 </script>
12724 <!-- Clicking the textarea is necessary to see upto the end of text. why? -->
12725 <input id="TextCanvas_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12726 <input id="TextCanvas_WorkCodeSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12727 <input id="TextCanvas_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12728 <span id="TextCanvas_WorkCodeView"></span>
12729 <script id="TextCanvas_WorkCodeScript">
12730 function TextCanvas_openWorkCodeView(){
12731   function TextCanvas_showWorkCode(){
12732     showHtmlCode(TextCanvas_WorkCodeView,TextCanvas_WorkCodeSpan);
12733   }
12734   TextCanvas_WorkCodeViewOpen.addEventListener('click',TextCanvas_showWorkCode);
12735 }
12736 TextCanvas_openWorkCodeView(); // should be invoked by an event
12737 </script>
12738 </details>
12739 <!-- TextCanvas_WorkCodeSpan -->
12740 <!-- //</span>
12741 <!--<!-- Work } ===== -->
12742 }
12743 }

```



```

1274://<!-- ===== Work { ===== -->
1274://<span id="Shading_WorkCodeSpan">
1274://
1274:<details><summary>Shading Canvas</summary>
1274:<!-- ===== Shading Canvas // 2020-1011 SatoxITS { -->
1274:<h2>Shading Canvas</h2>
1275:<note class="CommandUsageText">
1275:<b>Commands</b><br>
1275:Placement Mode<br>
1275:a ... apply (into absolute position)<br>
1275:j ... bring down (ArrowDown)<br>
1275:k ... bring up (ArrowUp)<br>
1275:h ... bring left (ArrowLeft)<br>
1275:l ... bring right (ArrowRight)<br>
1275:0 ... z-index = 0<br>
1275:+ ... z-index += 1<br>
1275:- ... z-index -= 1<br>
1275:r ... return to here (relative position)<br>
1275:c ... clear the log text<br>
1276:Note: the HTML text must be contenteditable to catch Key Event.<br>
1276:</note>
1276:
1276:
1276:<div id="Shading_1" class="ShadingPlate" draggable="true" contenteditable="true">
1276:<div id="Shading_1_Html" class="ShadingHtml" draggable="true"></div>
1276:<div id="Shading_1_Log" class="ShadingLog" draggable="true" style=""></div>
1277:
1277:<canvas id="Shading_1_Canvas" class="ShadingCanvas" width="300px" height="300px" draggable="true" style="">My Canvas.</canvas></div>
1277:
1277:
1277:ShadingPlate {
1277:  z-index:0;
1277:  position:static;
1277:  overflow:scroll;
1277:  display:block;
1277:  width:100%;
1277:  height:400px;
1277:  font-size:9pt;
1278:  font-family:Courier New;
1278:  border:1px dashed #000;
1278:  color:#444;
1278:}
1278:
1278:ShadingLog {
1278:  z-index:0;
1278:  position:relative;
1278:  display:block;
1278:  top:0px;
1278:  left:0px;
1278:  overflow:scroll;
1278:  width:100%;
1278:  font-size:9pt;
1278:  font-family:Courier New;
1278:  color:#666;
1278:  overflow:scroll;
1278:  background:rgba(200,255,200,0.4);
1278:  height:400px;
1278:}
1278:
1278:ShadingHtml {
1278:  z-index:2;
1278:  position:relative;
1278:  display:block;
1278:  top:0px;
1278:  left:0px;
1278:  overflow:scroll;
1278:  width:100%;
1278:  font-size:12pt;
1278:  font-family:Courier New;
1278:  color:#666;
1278:  overflow:scroll;
1278:  background:rgba(200,255,200,0.4);
1278:  height:400px;
1278:}
1278:
1278:ShadingCanvas {
1278:  z-index:3;
1278:  position:relative;
1278:  xdisplay:block;
1278:  top:0px;
1278:  left:0px;
1278:  resize:both;
1278:  border:1px solid #000;
1278:}
1278:
1278:
1278:</style>
1278:<input id="Shading_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
1278:<input id="Shading_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
1278:<input id="Shading_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
1278:<span id="Shading_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="Signature">
1278:<script id="Shading_WorkScript">
1278:
1278:function Shading_openWorkCodeView(){
1278:  function Shading_showWorkCode(){
1278:    showHtmlCode(Shading_WorkCodeView,Shading_WorkCodeSpan);
1278:  }
1278:  Shading_WorkCodeViewOpen.addEventListener('click',Shading_showWorkCode);
1278:}
1278:
1278:const BR = '<br>';
1278:Shading_openWorkCodeView(); // should be invoked by an event
1278:
1278:function sh_onClick(e){
1278:  Shading_1_Log.innerHTML += 'Click '+e.target.nodeName+'#'+e.target.id
1278:  + ' offset('+e.offsetX+', '+e.offsetY+')'
1278:  + ' client('+e.clientX+', '+e.clientY+')'
1278:  + ' page('+e.pageX+', '+e.pageY+')'
1278:  + ' screen('+e.screenX+', '+e.screenY+')'
1278:  +BR;
1278:  e.stopPropagation();
1278:  e.preventDefault();
1278:}
1278:
1278:function sh_onKeyUp(e){
1278:  if (Shading_1.style.zIndex == 0){
1278:    Shading_1.style.zIndex = 0;
1278:  }
1278:  zi = parseInt(Shading_1.style.zIndex);
1278:  if (e.key.length == 1){
1278:    Shading_1_Html.innerHTML += e.key;
1278:  }
1278:  if (e.key == '0'){ zi = 0; }else
1278:  if (e.key == '+'){ zi += 1; }else
1278:  if (e.key == '-'){ zi -= 1; }else
1278:  if (e.key == 'c'){
1278:    Shading_1_Log.innerHTML = '';
1278:  }else
1278:  if (e.key == 'r'){
1278:    Shading_1.style.position = "relative";
1278:    Shading_1.style.top = '0px';
1278:    Shading_1.style.left = '0px';
1278:    zi = 0;
1278:  }else
1278:  if (e.key == 'j' || e.code == 'ArrowDown'){
1278:    topx = parseInt(Shading_1.style.top) + 50;
1278:    Shading_1.style.top = topx + 'px';
1278:  }else
1278:  if (e.key == 'k' || e.code == 'ArrowUp'){
1278:    topx = parseInt(Shading_1.style.top) - 50;
1278:    Shading_1.style.top = topx + 'px';
1278:  }else
1278:  if (e.key == 'l' || e.code == 'ArrowRight'){
1278:    lefty = parseInt(Shading_1.style.left) + 50;
1278:    Shading_1.style.left = lefty + 'px';
1278:  }else
1278:  if (e.key == 'h' || e.code == 'ArrowLeft'){
1278:    lefty = parseInt(Shading_1.style.left) - 50;
1278:    Shading_1.style.left = lefty + 'px';
1278:  }else
1278:  if (e.key == 'a'){
1278:    Shading_1.style.position = "absolute";
1278:    Shading_1.style.top = '0px';
1278:    Shading_1.style.left = '0px';
1278:  }else{
1278:    Shading_1.style.zIndex = zi;
1278:    Shading_1_Log.innerHTML += 'Keypup..' + e.target.nodeName + '#' + e.target.id
1278:    + 'Up('+e.key+'/' + e.code+')'
1278:    + 'z-index:'+zi+'/' + Shading_1.style.zIndex
1278:    + 'top: '+Shading_1.style.top
1278:    +BR;
1278:    e.stopPropagation();
1278:    e.preventDefault();
1278:  }
1278:  function sh_onKeyDown(e){
1278:    Shading_1_Log.innerHTML += 'Keypdown'+e.target.nodeName+'#'+e.target.id
1278:    + 'Down('+e.key+'/' + e.code+')'+BR;
1278:    e.stopPropagation();
1278:    e.preventDefault();
1278:  }
1278:}
1278:
1278:
1278:function Shading_Setup(){
1278:  Shading_1_Log.innerHTML += '<+>h4>Click here<+>/h4>';
1278:  Shading_1.addEventListener('keydown',sh_onKeyDown);
1278:  Shading_1.addEventListener('keyup',sh_onKeyUp);
1278:  Shading_1.addEventListener('click',sh_onClick);
1278:  Shading_1.addEventListener('click',sh_onClick);
1278:  Shading_1_Log.style.top = "-400px";
1278:  Shading_1_Log.style.left = "200px";
1278:  Shading_1.appendChild(Shading_1_Canvas);
1278:  Shading_1_Canvas.style.width = "300px";
1278:  Shading_1_Canvas.style.height = "300px";
1278:  Shading_1_Canvas.style.position = "relative";

```

```

12921 Shading_1_Canvas.style.top = "-750px";
12922 Shading_1_Canvas.style.left = "100px";
12923
12924 const ctx = Shading_1_Canvas.getContext('2d');
12925 ctx.fillStyle = 'rgba(160,0,0,0.9)';
12926 ctx.fillRect(50,50,40,40);
12927 ctx.fillStyle = 'rgba(0,160,0,0.9)';
12928 ctx.fillRect(60,60,40,40);
12929 ctx.fillStyle = 'rgba(0,0,160,0.9)';
12930 ctx.fillRect(70,70,40,40);
12931 }
12932 function Reset_ShadingCanvas(){
12933   Shading_1_Log.removeAttribute('style');
12934   Shading_1_Log.innerHTML = "";
12935   Shading_1_Canvas.style = "";
12936   //Shading_1_Canvas.removeAttribute('style');
12937 }
12938
12939 </script>
12940 </details>
12941 <!-- Shading_WorkCodeSpan -->
12942 *//</span>
12943 <!-- ===== Work } ===== -->
12944
12945
12946 <!-- ===== Work { ===== -->
12947 <span id="Charmap_WorkCodeSpan">
12948 /*
12949 <details id="Charmap_Work"><summary>Character Map Mandala</summary>
12950 <!-- ----- UnicodeCharmap // 2020-1015 SatoxITS { -->
12951 <h2>Unicode Character Map</h2>
12952 <notecode 0x0000 - 0xffff // 16px / 3200 x 3200 px / zoom:0.25</note>
12953 <div id="Charmap_1_Frame">
12954 <div id="Charmap_1_Text" class="Charmap">
12955 </div>
12956 </div>
12957 <br>
12958
12959 <style>
12960 #Charmap_1_Frame {
12961   overflow:scroll;
12962   height:3200px; width:3200px;
12963   xtransform:scale(0.5);
12964   zoom:0.25;
12965   resize:both;
12966   background-color:#fff;
12967 }
12968 Charmap {
12969   xzoom:0.25;
12970   font-size:16px;
12971   line-height:1.0;
12972   xfont-family:Georgia;
12973   color:#000;
12974 }
12975 </style>
12976 <script>
12977 function charmapgen(){
12978   text = '';
12979   for( cc = 0; cc < 0x10000; cc++ ){
12980     text += String.fromCharCode(cc);
12981   }
12982   Charmap_1_Text.innerHTML = text;
12983 }
12984 Charmap_Work.addEventListener('click', charmapgen);
12985 //charmapgen();
12986 </script>
12987
12988 <input id="Charmap_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12989 <input id="Charmap_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12990 <input id="Charmap_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12991 <span id="Charmap_WorkCodeView"></span>
12992 <script id="Charmap_WorkScript">
12993 function Charmap_openWorkCodeView(){
12994   function Charmap_showWorkCode(){
12995     showHtmlCode(Charmap_WorkCodeView,Charmap_WorkCodeSpan);
12996   }
12997   Charmap_WorkCodeViewOpen.addEventListener('click', Charmap_showWorkCode);
12998 }
12999 Charmap_openWorkCodeView(); // should be invoked by an event
13000 </script>
13001 </details>
13002 <!-- Charmap_WorkCodeSpan -->
13003 *//</span>
13004 <!-- ===== Work } ===== -->
13005
13006
13007 <!-- ===== Work { ===== -->
13008 <span id="Pointillism_WorkCodeSpan">
13009 /*
13010 <details><summary>Collaborated Pointillism</summary>
13011 <!-- ----- CollaboratedPointillism // 2020-1016 SatoxITS { -->
13012 <h2><a name="Pointillism" class="Pointillism"><a href="#Pointillism">Collaborated Pointillism</a></h2>
13013
13014 <input id="Pointillism_1_Share" type="checkbox" class="HtmlCodeViewButton" value="Share"> Share
13015 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ResetCanvas()" value="Reset">
13016 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Clear">
13017 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ReplayCanvas()" value="Replay">
13018 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_RepeatCanvas()" value="Repeat">
13019 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_SaveCanvas()" value="Save">
13020 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_LoadCanvas()" value="Load">
13021 <div id="Pointillism_1" class="Pointillism">
13022
13023 <span id="Pointillism_1_Unit_1" class="Pointillism_Unit">
13024 <span id="Pointillism_1_XY_1" class="Pointillism_XY">XY1</span>
13025 <span id="Pointillism_1_XY_1_Remote" class="Pointillism_XY_Remote">XY1-remote</span>
13026 <canvas id="Pointillism_1_Canvas_1" class="Pointillism_Canvas" width="300px" height="300px"></canvas>
13027 </span>
13028
13029 <span id="Pointillism_1_Unit_2" class="Pointillism_Unit">
13030 <span id="Pointillism_1_XY_2" class="Pointillism_XY">XY2</span>
13031 <span id="Pointillism_1_XY_2_Remote" class="Pointillism_XY_Remote">XY2-remote</span>
13032 <canvas id="Pointillism_1_Canvas_2" class="Pointillism_Canvas" width="300px" height="300px"></canvas>
13033 </span>
13034 </div>
13035 <br>
13036
13037 <style>
13038 Pointillism {
13039   xdisplay:block;
13040   resize:both;
13041   width:680px;
13042   height:380px;
13043   min-width:240px;
13044   min-height:270px;
13045   background-color:#eee;
13046   overflow:scroll;
13047   font-size:16px;
13048   font-family:Georgia;
13049   color:#000;
13050   vertical-align:middle;
13051 }
13052
13053 Pointillism_Unit {
13054   position:relative;
13055   top:0px;
13056   display:block;
13057   overflow:scroll;
13058   width:300px;
13059   height:300px;
13060   margin:5px;
13061   padding:10px;
13062   background-color:rgba(255,255,127,0.7);
13063 }
13064
13065 Pointillism_XY {
13066   display:block;
13067   vertical-align:middle;
13068   width:290px;
13069   xheight:20px;
13070   font-size:12px;
13071   line-height:1.2;
13072   padding:5px;
13073   margin:0px;
13074   color:#fff;
13075   background-color:#444;
13076 }
13077
13078 Pointillism_XY_Remote {
13079   display:block;
13080   vertical-align:middle;
13081   width:290px;
13082   xheight:20px;
13083   font-size:12px;
13084   line-height:1.2;
13085   padding:5px;
13086   color:#fff;
13087   background-color:#444;
13088 }
13089
13090 Pointillism_Canvas {
13091   display:block;
13092   position:relative;
13093   xpadding:20px;
13094   xleft:20px;
13095   xtop:20px;
13096   background-color:#333;
13097 }
13098 </style>
13099 <script>
13100 var points = [];

```

```

1309 var replay = [];
1309 var replayx = 0;
1310 function pClearCanvas(can){
1310   ctx = can.getContext('2d');
1310   ctx.clearRect(0,0,can.width,can.height);
1310 }
1310 function Pointillism_1_ClearCanvas(){
1310   pClearCanvas(Pointillism_1_Canvas_1);
1310   pClearCanvas(Pointillism_1_Canvas_2);
1310 }
1310 function PointsReset(){
1310   points = [];
1310   replay = [];
1310   inRepeat = false;
1310   inReplay = false;
1310   Pointillism_1_ClearCanvas();
1310 }
1310 function Pointillism_1_ResetCanvas(){
1310   PointsReset();
1310   if( Pointillism_1_Share.checked ){
1310     //alert('---thread cast reset\n');
1310     GJ_BcastMessage('DRAW RESET');
1310   }
1310 }
1310 function Pointillism_1_ResetCanvasReceive(){
1310   //alert('---received reset\n');
1310   PointsReset();
1310 }
1310 function drawPoint(can,x,y,r,g,b){
1310   const ctx = can.getContext('2d');
1310   ctx.fillStyle = 'rgba('+r+', '+g+', '+b+', 0.7)';
1310   ctx.fillRect(x,y,r,g);
1310 }
1310 function waitMs(serno,ms){
1310   console.log('-- wait #' +serno+ ' '+ms+'ms');
1310   until = new Date();
1310   now = until.getTime();
1310   untilMs = now + ms;
1310   for( wi = 0 ; wi <+ ; ){
1310     now = new Date();
1310     nowMs = now.getTime();
1310     remMs = untilMs - nowMs;
1310     //console.log('wait '+wi+': '+remMs+'/' +ms);
1310     if( remMs < 0 ){
1310       break;
1310     }
1310   }
1310 }
1310 var inReplay = false;
1310 function replay1(){
1310   rx = replayx;
1310   if( replay.length <= rx ){
1310     return;
1310   }
1310   replayx += 1;
1310   pl = replay[rx];
1310   if( pl[1] == 1 ){
1310     can = Pointillism_1_Canvas_1;
1310   }else{
1310     can = Pointillism_1_Canvas_2;
1310   }
1310   drawPoint(can,pl[2],pl[3],pl[4],pl[5],pl[6]);
1310   if( inReplay == false ){
1310     console.log('wait '+replayx+' Stopped');
1310     return;
1310   }
1310   if( rx < replay.length-1 ){
1310     prevMs = replay[rx][0].getTime();
1310     nextMs = replay[rx+1][0].getTime();
1310     delayMs = nextMs - prevMs;
1310     //console.log('wait '+replayx+' '+delayMs+'ms');
1310     window.setTimeout(replay1,delayMs);
1310   }else{
1310     console.log('wait '+replayx+' Finished');
1310     if( inRepeat ){
1310       window.setTimeout(repeat1,1000);
1310     }
1310   }
1310 }
1310 function Pointillism_1_ReplayCanvas(can){
1310   Pointillism_1_ClearCanvas();
1310   replay = points;
1310   replayx = 0;
1310   inReplay = true;
1310   replay1();
1310 }
1310 var inRepeat = false;
1310 function repeat1(){
1310   Pointillism_1_ClearCanvas();
1310   replay = points;
1310   replayx = 0;
1310   replay1();
1310   if( inRepeat ){
1310     //window.setTimeout(repeat1,1000);
1310   }
1310 }
1310 function Pointillism_1_RepeatCanvas(can){
1310   if( inRepeat ){
1310     inRepeat = false;
1310     inReplay = false;
1310   }else{
1310     inRepeat = true;
1310     inReplay = true;
1310     repeat1();
1310   }
1310 }
1310 function CopyLocal(){ return Pointillism_1_Share.checked == false; }
1310 function Pointillism_Setup(){
1310   var moveCount1 = 0;
1310   var moveCount2 = 0;
1310 }
1310 var gJlinked = false;
1310 function GJdraw(msg){
1310   if( gJlinked == false ){
1310     //GJLink_Section.open = true;
1310     GJ_Join();
1310     gJlinked = true;
1310   }
1310   GJ_BcastMessage('DRAW '+msg);
1310 }
1310 function showXY1(e){
1310   moveCount1 += 1;
1310   x = e.offsetX;
1310   y = e.offsetY;
1310   Pointillism_1_XY_1.innerHTML = 'XY1: '+ 'x='+x +', y='+y + ' /'+moveCount1+'/' +points.length;
1310   Pointillism_1_XY_2_Remote.innerHTML = 'XY1: '+ 'x='+x +', y='+y + ' /'+moveCount1;
1310   if( e.buttons || CopyLocal() ){
1310     points.push([new Date(),1,x,y,64,64,255]);
1310     drawPoint(Pointillism_1_Canvas_1,x,y,128,128,255);
1310     if( CopyLocal() ){
1310       drawPoint(Pointillism_1_Canvas_2,x,y,160,160,255);
1310     }
1310     GJdraw('1,'+x+', '+y);
1310   }
1310 }
1310 function showXY2(e){
1310   moveCount2 += 1;
1310   x = e.offsetX;
1310   y = e.offsetY;
1310   Pointillism_1_XY_2_Remote.innerHTML = 'XY2: '+ 'x='+x +', y='+y + ' /'+moveCount2+'/' +points.length;
1310   Pointillism_1_XY_1_Remote.innerHTML = 'XY2: '+ 'x='+x +', y='+y + ' /'+moveCount1;
1310   if( e.buttons || CopyLocal() ){
1310     points.push([new Date(),2,x,y,64,255,64]);
1310     drawPoint(Pointillism_1_Canvas_2,x,y,128,255,128);
1310     if( CopyLocal() ){
1310       drawPoint(Pointillism_1_Canvas_1,x,y,160,255,160);
1310     }
1310     //GJdraw('2,'+x+', '+y);
1310   }
1310 }
1310 Pointillism_1_Canvas_1.addEventListener('mousemove',showXY1);
1310 Pointillism_1_Canvas_2.addEventListener('mousemove',showXY2);
1310 Pointillism_1_Unit_2.style.left = '340px';
1310 Pointillism_1_Unit_2.style.top = '-375px';
1310 }
1310 function Pointillism_RemoteDraw(arg){
1310   //alert('Draw at '+arg);
1310   //drawPoint(Pointillism_1_Canvas_2,x,y,160,160,255);
1310   if( arg == 'RESET' ){
1310     Pointillism_1_ResetCanvasReceive();
1310   }else{
1310     argv = arg.split(',');
1310     x = argv[1];
1310     y = argv[2];
1310     Pointillism_1_XY_2_Remote.innerHTML = 'XYR: '+ 'x='+x +', y='+y + ' /'+points.length;
1310     drawPoint(Pointillism_1_Canvas_2,x,y,255,0,0);
1310   }
1310 }
1310 </script>
1310
1310 <input id="Pointillism_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
1310 <input id="Pointillism_WorkOpensSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
1310 <input id="Pointillism_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
1310 <span id="Pointillism_WorkCodeView"></span>
1310 <script id="Pointillism_WorkScript">

```

```

13271 function Pointillism_openWorkCodeView(){
13272     function Pointillism_showWorkCode(){
13273         showHtmlCode(Pointillism_WorkCodeView,Pointillism_WorkCodeSpan);
13274     }
13275     Pointillism_WorkCodeViewOpen.addEventListener('click',Pointillism_showWorkCode);
13276 }
13277 Pointillism_openWorkCodeView(); // should be invoked by an event
13278 </script>
13279 </details>
13280 <!-- Pointillism_WorkCodeSpan } -->
13281 * //</span>
13282 <!-- ===== Work } ===== -->
13283
13284
13285 <!-- ===== Work { ===== -->
13286 <span id="StatCounter_WorkCodeSpan">
13287 *
13288 <details><summary>StatCounter</summary>
13289 <!-- ----- StatCounter // 2020-1018 SatoxITS { -->
13290 <h2>StatCounter</h2>
13291
13292 <div class="statcounter"><a title="hit counter" href="https://statcounter.com/" target="_blank"></a>
13293 <style>
13300 .statcounter {
13301     vertical-align:middle;
13302 }
13303 #sc_SatoxITS {
13304     color:#000;
13305     font-size:12pt;
13306     height:30px;
13307     width:100%;
13308     background-color:#ddd;
13309 }
13310 </style>
13311
13312 <div>
13313 <script>
13314     var sc_project=12411639;
13315     var sc_invisible=0;
13316     var sc_security="1aeb2a3a";
13317     var sc_https=1;
13318     var scJsHost = "https://";
13319 </script>
13320 <!-- script src="https://statcounter.com/counter.js" -->
13321 <!-- /script --> (counter by inline script)
13322 </div>
13323
13324 <input id="StatCounter_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
13325 <input id="StatCounter_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
13326 <input id="StatCounter_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
13327 </span id="StatCounter_WorkCodeView"></span>
13328 <script id="StatCounter_WorkScript">
13329 function StatCounter_openWorkCodeView(){
13330     function StatCounter_showWorkCode(){
13331         showHtmlCode(StatCounter_WorkCodeView,StatCounter_WorkCodeSpan);
13332     }
13333     StatCounter_WorkCodeViewOpen.addEventListener('click',StatCounter_showWorkCode);
13334 }
13335 StatCounter_openWorkCodeView(); // should be invoked by an event
13336 </script>
13337
13338 </details>
13339 <!-- StatCounter_WorkCodeSpan } -->
13340 * //</span>
13341 <!-- ===== Work } ===== -->
13342
13343
13344 <!-- ===== Work { ===== -->
13345 <span id="CascadedCanvasBook_WorkCodeSpan">
13346 *
13347 <details id="CascadedCanvasBook_Section"><summary id="CascadedCanvasBook_Summary">CascadedCanvasBook</summary>
13348 <!-- ----- CascadedCanvasBook // 2020-1031 SatoxITS { -->
13349 <h2>Cascaded Canvas Book</h2>
13350
13351 <div id="CBPanel" class="CBPanel">
13352 <input id="CS_new" type="button" value="NewCanvas">
13353 </div>
13354 <br>
13355
13356 <h3>Undo / Redo / Replay</h3>
13357
13358 <div id="CanvasTool_UndoRedo">
13359 <span id="DrawReplay" class="CanvasTool" draggable="true" contenteditable="">
13360 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13361 <input class="CV_Button" type="button" value="Redraw">
13362 From <input data-name="D" class="ColorParam" type="text" value="0" onwheel="OnWheelInt()">
13363 To <input id="DrawingSernoView" data-name="D" class="ColorParam" type="text" value="0" onwheel="OnWheelInt()">
13364 </span>
13365 </div>
13366
13367 <script>
13368 function childByName(node,name){
13369     for (let i = 0; i < node.children.length; i++) {
13370         ch = node.children[i];
13371         name = ch.getAttribute('data-name');
13372         if (name == name){
13373             return ch;
13374         }
13375     }
13376     return null;
13377 }
13378
13379 function OnWheelInt(){
13380     event.preventDefault();
13381     t = event.target;
13382     n = t.nodeName;
13383     i = t.id;
13384     p = t.parentNode;
13385     y = event.deltaY;
13386     //console.log('OnWheelInt '+y+' '+n+'#'+i+' '+t.value);
13387     if (y < 0) { // scroll forward (up)
13388         inc = -y;
13389     }else{
13390         inc = y;
13391     }
13392     inc /= 6;
13393     val = parseFloat(t.value) + inc;
13394     t.value = val.toFixed(0);
13395     return val;
13396 }
13397
13398 var DrawingSerno = 0;
13399 function saveDrawing(){
13400     DrawingSerno += 1;
13401     DrawingSernoView.value = DrawingSerno;
13402 }
13403
13404 function to2x(x){
13405     if (x <= 0xf) {
13406         return '0'+x.toString(16);
13407     }else{
13408         return x.toString(16);
13409     }
13410 }
13411 </script>
13412
13413 <div id="InstaColorPicker" draggable="true">
13414 <h3>Color Picker</h3>
13415 <div id="CanvasColor">
13416
13417 Select value by
13418 <input id="ICMotion" type="checkbox" checked="">Mouse Motion
13419 <input id="ICWheel" type="checkbox" checked="">Mouse Wheel
13420 <input id="ICPautoAddWheel" type="checkbox" checked="">Auto. add to history
13421
13422 <div data-name="Fore" id="CanvasTool_Color Fore" class="CanvasTool" onchange="showColor1Sample()">
13423 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13424 <input class="CV_Button" type="button" value="Fore">
13425 <input data-name="R" class="CanvasParam" type="text" value="22" onwheel="OnWheelHex()">
13426 <input data-name="G" class="CanvasParam" type="text" value="32" onwheel="OnWheelHex()">
13427 <input data-name="B" class="CanvasParam" type="text" value="32" onwheel="OnWheelHex()">
13428 <input data-name="A" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13429 <input data-name="C" class="ColorParam" type="text" value="#000000ff">
13430 <span data-name="Sample">Sample</span>
13431 </div>
13432
13433 <div data-name="Fill" id="CanvasTool_Color Fill" class="CanvasTool" onchange="showColor1Sample()">
13434 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13435 <input class="CV_Button" type="button" value="Fill">
13436 <input data-name="R" class="CanvasParam" type="text" value="0" onwheel="OnWheelHex()">
13437 <input data-name="G" class="CanvasParam" type="text" value="0" onwheel="OnWheelHex()">
13438 <input data-name="B" class="CanvasParam" type="text" value="0" onwheel="OnWheelHex()">
13439 <input data-name="A" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13440 <input data-name="C" class="ColorParam" type="text" value="#000000ff">
13441 <span data-name="Sample">Sample</span>
13442 </div>
13443
13444 <div data-name="Back" id="CanvasTool_Color Back" class="CanvasTool" onchange="showColor1Sample()">
13445 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13446 <input class="CV_Button" type="button" value="Back">
13447 <input data-name="R" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13448 <input data-name="G" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13449 <input data-name="B" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13450 </div>

```

```

13452 <input data-name="A" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13453 RGB<input data-name="C" class="ColorParam" type="text" value="#ffffff">
13454 <span data-name="Sample"><span>
13455 </div>
13456
13457 <div data-name="Border" id="CanvasTool_Color_Border" class="CanvasTool" onchange="showColor1Sample()">
13458 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13459 <input class="CV_Button" type="button" value="Border">
13460 <input data-name="R" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13461 <input data-name="G" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13462 <input data-name="B" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13463 <input data-name="A" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13464 RGB<input data-name="C" class="ColorParam" type="text" value="#ffffff">
13465 <span data-name="Sample"><span>
13466 </div>
13467
13468 <div id="ColorComposition" class="ColorComposition">
13469 Sample
13470 </div>
13471 <br>
13472 <input class="LargeButton CV_Button" type="button" value="Clear Color History" onclick="ClearColorHistory()">
13473 <input id="LenColorHistory" class="CanvasParam" type="text" value="0">
13474 / Max.<input id="MaxColorHistory" class="CanvasParam" type="text" value="200">
13475 <div id="Color0" class="Color1" onclick="SelectThisColor()"></div>
13476 <div id="Color1" class="Color1" onclick="SelectThisColor()"></div>
13477 </div>
13478 </div>
13479 </div>
13480 <style>
13481 .LargeButton {
13482   font-size:14px !important;
13483   width:160px !important;
13484   color:#f00 !important;
13485 }
13486 .ColorComposition {
13487   color:#000000ff !important;
13488   font-size:16pt !important;
13489   padding:4px !important;
13490   border:2px solid #000000ff !important;
13491 }
13492 .Color1 {
13493   width:100% !important;
13494   height:10px !important;
13495   font-family:Courier New !important;
13496   font-size:10px !important;
13497   background-color:#000 !important;
13498 }
13499 .Color1:hover {
13500   border:1px solid #000;
13501 }
13502 .ColorHistory {
13503   resize:both;
13504   //overflow:scroll;
13505   width:100% !important;
13506   height:200px !important;
13507 }
13508 </style>
13509
13510 <script>
13511 var ColorID = 0;
13512 var LastColor = Color1;
13513 var HistLength = 0;
13514 function ClearColorHistory(){
13515   ColorHistory.innerHTML = '';
13516   cl = Color0.cloneNode();
13517   cl.id = 'Color1';
13518   LastColor = document.getElementById('Color1');
13519   ColorHistory.appendChild(cl);
13520   HistLength = 0;
13521   LenColorHistory.value = '0';
13522 }
13523 function OnWheelHex(){
13524   event.preventDefault();
13525   t = event.target;
13526   n = t.nodeName;
13527   i = t.id;
13528   p = t.parentNode;
13529   y = event.deltaY;
13530   inc = y; // scroll forward (up)
13531   inc /= 5;
13532
13533   val = parseFloat(t.value) + inc;
13534   val = val.toFixed(0);
13535   if (val < 0) val = 0;
13536   if (255 < val) val = 255;
13537   if (t.value != val || event.ctrlKey){
13538     t.value = val;
13539     if (ICPautoAddWheel.checked){
13540       showColor1Sample();
13541     }
13542   }
13543   return val;
13544 }
13545 function SelectThisColor(){
13546   ok = confirm('Pick this color ? '+event.target.innerHTML);
13547   if (ok){
13548     // add to Picked
13549   }
13550 }
13551
13552 var LastMotion = 0;
13553 function motionColor(){
13554   if (ICPmotion.checked){
13555     return;
13556   }
13557   d = new Date();
13558   if (d.getTime() - LastMotion < 100){
13559     return;
13560   }
13561   LastMotion = d.getTime();
13562
13563   t = CanvasTool_Color_Fore;
13564   R = childByName(t,'R');
13565   G = childByName(t,'G');
13566   B = childByName(t,'B');
13567   A = childByName(t,'A');
13568   // console.log('mouse motion '+event.x+', '+event.y+' target#'+it);
13569
13570   updated = false;
13571
13572   val = 255 * (event.x/window.innerWidth);
13573   val = val.toFixed(0);
13574   if (B.value != val || event.ctrlKey){
13575     B.value = val;
13576     updated = true;
13577   }
13578   val = 255 * (event.y/window.innerHeight);
13579   val = val.toFixed(0);
13580   if (G.value != val || event.ctrlKey){
13581     G.value = val;
13582     updated = true;
13583   }
13584   if (updated){
13585     showColor1Sample(t,ICPautoAddMotion.checked);
13586   }
13587 }
13588 function scrollColor(){
13589   if (ICPmotion.checked){
13590     return;
13591   }
13592   d = new Date();
13593   if (d.getTime() - LastMotion < 100){
13594     return;
13595   }
13596   LastMotion = d.getTime();
13597
13598   t = CanvasTool_Color_Fore;
13599   R = childByName(t,'R');
13600   G = childByName(t,'G');
13601   B = childByName(t,'B');
13602   A = childByName(t,'A');
13603
13604   updated = false;
13605
13606   y = gsh.getBoudingClientRect().top.toFixed(0)
13607   if (y < 0) y += -1;
13608   size = 10000;
13609   val = 255 * (y/size);
13610   val = val.toFixed(0);
13611   if (255 < val) val = 255;
13612   if (R.value != val || event.ctrlKey){
13613     R.value = val;
13614     updated = true;
13615   }
13616   if (updated){
13617     showColor1Sample(t,ICPautoAddMotion.checked);
13618   }
13619 }
13620
13621 function showColor1Sample(){
13622   t = event.target.parentNode;
13623   showColor1Sample1(t,ICPautoAddWheel);
13624 }
13625 function showColor1Sample1(t,add){
13626   name = t.getAttribute('data-name');
13627 }
13628

```

```

13629 R = childByName(t,'R').value;
13630 G = childByName(t,'G').value;
13631 B = childByName(t,'B').value;
13632 A = childByName(t,'A').value;
13633
13634 R = parseInt(R);
13635 G = parseInt(G);
13636 B = parseInt(B);
13637 A = parseInt(A);
13638
13639 R = to02x(R); //R.toString(16);
13640 G = to02x(G); //G.toString(16);
13641 B = to02x(B); //B.toString(16);
13642 A = to02x(A); //A.toString(16);
13643
13644 color = '#'+R+G+B+A;
13645 //console.log(name+' color='+color);
13646
13647 C = childByName(t,'C');
13648 C.value = color;
13649 S = childByName(t,'Sample');
13650 S.style.color = color;
13651
13652 ColorID += 1;
13653 cl = Color1;
13654 clid = cl.id;
13655 cl.id = "color_"+ColorID;
13656 cl.innerHTML = cl.id +
13657 + '<'+font color=black>'+color+'</'+font>'+<'+font color=white>'+color+'</'+font>';
13658 if( name == 'Fore' ){
13659 ColorComposition.style.setProperty('color',color,'important');
13660 cl.style.setProperty('color',color,'important');
13661 cl.style.setProperty('background-color',color,'important');
13662 }
13663 if( name == 'Fill' ){
13664 ColorComposition.style.setProperty('border-color',color,'important');
13665 cl.style.setProperty('color',color,'important');
13666 cl.style.setProperty('background-color',color,'important');
13667 }
13668 if( name == 'Back' ){
13669 ColorComposition.style.setProperty('background-color',color,'important');
13670 cl.style.setProperty('background-color',color,'important');
13671 }
13672 if( name == 'Border' ){
13673 ColorComposition.style.setProperty('border-color',color,'important');
13674 cl.style.setProperty('background-color',color,'important');
13675 cl.style.setProperty('border-color',color,'important');
13676 }
13677 ccl = cl.cloneNode(true);
13678 clid = clid;
13679
13680 if( add ){
13681 max = parseInt(MaxColorHistory.value);
13682 if( HistLength < max ){
13683 HistLength++;
13684 LenColorHistory.value = HistLength;
13685 ColorHistory.insertBefore(ccl,LastColor);
13686 LastColor = ccl;
13687 }
13688 if( max <= HistLength ){
13689 LenColorHistory.style.setProperty('background-color','#f00','important');
13690 }else{
13691 LenColorHistory.style.setProperty('background-color','#fff','important');
13692 }
13693 }
13694 }
13695
13696 function InstaColor_Setup1(){
13697 window.addEventListener('mousemove',motionColor);
13698 window.addEventListener('scroll',scrollColor);
13699 //window.addEventListener('click',motionColor); // click is generated for animation
13700
13701 fi = document.getElementById('FeaturesView');
13702 if( fi != null ){
13703 fi.appendChild(InstaColorPicker);
13704 //cs = document.getElementById('InstaColorSpan');
13705 //cs.appendChild(InstaColorPicker);
13706 //ci.hidden = false;
13707 //ci.open = true;
13708 }
13709 }
13710
13711 function InstaColor_Setup(){
13712 if( CascadedCanvasBook_Section.open ){
13713 InstaColor_Setup1();
13714 }
13715 CascadedCanvasBook_Summary.addEventListener('click',InstaColor_Setup1);
13716 }
13717 </script>
13718
13719 <h3>Colors</h3>
13720
13721 <div id="CanvasColors">
13722 <div id="CanvasTool_Color_0" class="CanvasTool" onchange="showColorSample()">
13723 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13724 <input class="CV_Button" type="button" value="Trans">
13725 <span data-name="I" class="ColorParam" type="text"><0-0</span>
13726 <input data-name="BW" class="CanvasParam" type="text" value="0">
13727 <input data-name="FC" class="ColorParam" type="text" value="#000000">
13728 <input data-name="FO" class="CanvasParam" type="text" value="0.0">
13729 <span data-name="FCS" class="ColorSample">xxxx</span>
13730 <span data-name="BCS" class="ColorSample">xxxx</span>
13731 <input data-name="BC" class="ColorParam" type="text" value="#000000">
13732 <input data-name="BO" class="CanvasParam" type="text" value="0.0">
13733 </div>
13734 <div id="CanvasTool_Color_1" class="CanvasTool" onchange="showColorSample()">
13735 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13736 <input class="CV_Button" type="button" value="Mono">
13737 <span data-name="I" class="ColorParam" type="text"><0-1</span>
13738 <input data-name="BW" class="CanvasParam" type="text" value="1">
13739 <input data-name="FC" class="ColorParam" type="text" value="#000000">
13740 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
13741 <span data-name="FCS" class="ColorSample">xxxx</span>
13742 <span data-name="BCS" class="ColorSample">xxxx</span>
13743 <input data-name="BC" class="ColorParam" type="text" value="#d0d0d0">
13744 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
13745 </div>
13746 <div id="CanvasTool_Color_2" class="CanvasTool" onchange="showColorSample()">
13747 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13748 <input class="CV_Button" type="button" value="Red">
13749 <span data-name="I" class="ColorParam" type="text"><0-2</span>
13750 <input data-name="BW" class="CanvasParam" type="text" value="1">
13751 <input data-name="FC" class="ColorParam" type="text" value="#e20202">
13752 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
13753 <span data-name="FCS" class="ColorSample">xxxx</span>
13754 <span data-name="BCS" class="ColorSample">xxxx</span>
13755 <input data-name="BC" class="ColorParam" type="text" value="#ffffff">
13756 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
13757 </div>
13758 <div id="CanvasTool_Color_3" class="CanvasTool" onchange="showColorSample()">
13759 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13760 <input class="CV_Button" type="button" value="Green">
13761 <span data-name="I" class="ColorParam" type="text"><0-3</span>
13762 <input data-name="BW" class="CanvasParam" type="text" value="1">
13763 <input data-name="FC" class="ColorParam" type="text" value="#20c820">
13764 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
13765 <span data-name="FCS" class="ColorSample">xxxx</span>
13766 <span data-name="BCS" class="ColorSample">xxxx</span>
13767 <input data-name="BC" class="ColorParam" type="text" value="#ffffff">
13768 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
13769 </div>
13770 <div id="CanvasTool_Color_4" class="CanvasTool" onchange="showColorSample()">
13771 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13772 <input class="CV_Button" type="button" value="Blue">
13773 <span data-name="I" class="ColorParam" type="text"><0-4</span>
13774 <input data-name="BW" class="CanvasParam" type="text" value="1">
13775 <input data-name="FC" class="ColorParam" type="text" value="#6080f0">
13776 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
13777 <span data-name="FCS" class="ColorSample">xxxx</span>
13778 <span data-name="BCS" class="ColorSample">xxxx</span>
13779 <input data-name="BC" class="ColorParam" type="text" value="#0000c0">
13780 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
13781 </div>
13782 <div id="CanvasTool_Color_5" class="CanvasTool" onchange="showColorSample()">
13783 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13784 <input class="CV_Button" type="button" value="Yellow">
13785 <span data-name="I" class="ColorParam" type="text"><0-5</span>
13786 <input data-name="BW" class="CanvasParam" type="text" value="1">
13787 <input data-name="FC" class="ColorParam" type="text" value="#fa600">
13788 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
13789 <span data-name="FCS" class="ColorSample">xxxx</span>
13790 <span data-name="BCS" class="ColorSample">xxxx</span>
13791 <input data-name="BC" class="ColorParam" type="text" value="#ffffff">
13792 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
13793 </div>
13794 </script>
13795 // https://developer.mozilla.org/en-US/docs/Web/CSS/color_value
13796 function genCSSColorStyle(color,opa){
13797 opa = parseFloat(opa);
13798 opa *= 0xFF;
13799 opa = opa.toFixed(0);
13800 if( 0xFF < opa ) opa = 0xFF;
13801 opa = parseInt(opa);
13802 opa = opa.toString(16);
13803 if( opa < 0x10 ) opa = '0'+opa;
13804 color += opa;
13805 return color;

```

```

1380 function CV_GenColorStyle(cole,forFill){
1381   if( forFill ){
1382     col = childByName(cole, 'FC').value;
1383     opa = childByName(cole, 'FO').value;
1384   }else{
1385     col = childByName(cole, 'BC').value;
1386     opa = childByName(cole, 'BO').value;
1387   }
1388   color = genCSSColorStyle(col,opa);
1389   return color;
1390 }
1391 function xxxshowColorSample(){
1392   t = event.target;
1393   p = t.parentNode;
1394   alert('showColorSample '+event.target.nodeName+'/'+#'+p.id);
1395 }
1396 function showColorSample(){
1397   var csv = CanvasColors.children;
1398   //console.log('colors'+csv.length);
1399   for( i = 0; i < csv.length; i++){
1400     fc = childByName(csv[i], 'FC').value;
1401     bc = childByName(csv[i], 'BC').value;
1402     //console.log('colors'+csv[i].id+' fc='+fc+' bc='+bc);
1403     fcs = childByName(csv[i], 'FCS');
1404     fcs.style.color = fc;
1405     fcs.style.borderColor = fc;
1406     fcs.style.backgroundColor = bc;
1407   }
1408   bcs = childByName(csv[i], 'BCS');
1409   bcs.style.color = bc;
1410   bcs.style.borderColor = fc;
1411   bcs.style.backgroundColor = fc;
1412 }
1413 CV_redrawParts();
1414 }
1415 </script>
1416 <style>
1417 .ColorSample {
1418   color:#fff;
1419   background-color:#ff0;
1420   border:1px solid #000;
1421   margin:0px;
1422   padding:0px;
1423   width:12pt !important;
1424   height:12pt !important;
1425 }
1426 .ColorParam {
1427   color:#000 !important;
1428   font-family:Courier New !important;
1429   font-size:9pt !important;
1430   padding:2px !important;
1431   line-height:1.1 !important;
1432   height:14pt !important;
1433   width:55pt !important;
1434   text-align:left !important;
1435   display:inline !important;
1436   vertical-align:middle !important;
1437 }
1438 .CanvasParam {
1439   color:#000 !important;
1440   font-family:Courier New, Monospace !important;
1441   font-size:9pt !important;
1442   padding:2px !important;
1443   line-height:1.1 !important;
1444   height:14pt !important;
1445   width:30pt !important;
1446   text-align:right !important;
1447   display:inline !important;
1448   vertical-align:middle !important;
1449 }
1450 .CV_Button {
1451   padding:2pt !important;
1452   line-height:1.1 !important;
1453   border:2px inset #bbb !important;
1454   font-size:9pt !important;
1455   font-weight:normal !important;
1456   font-family:Georgia !important;
1457   border-radius:3px !important;
1458   color:#ddd; background-color:#66a !important;
1459   width:50pt;
1460 }
1461 .xxHtmlCodeviewText {
1462   font-size:9pt;
1463   font-family:Courier New;
1464   white-space:pre;
1465 }
1466 .xxCV_Button {
1467   font-family:Arial, Monospace, Courier New;
1468   color:#000;
1469   font-size:9pt;
1470   line-height:1.2;
1471   width:50pt;
1472 }
1473 </style>
1474 <div id="Parts">
1475 <div id="ApperToCanvas" class="CanvasTool" draggable="true">
1476   <input data-name="Resize" class="CanvasParam" type="text" value="100" omwheel="OnWheelResize()" %>
1477   <input data-name="Zoom" class="CanvasParam" type="text" value="100" omwheel="OnWheelZoom()" %>
1478   <input id="RedrawImmediate" type="checkbox" value="Redraw" checked="">Redraw Immediate
1479 </div>
1480 <span id="CanvasTools" class="CanvasTools" draggable="true" contenteditable="">
1481 <div id="CanvasTool_Clear" class="CanvasTool">
1482   <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
1483   <input data-name="rdr" class="CV_Button" type="button" value="Rect" onclick="CV_drawRect()">
1484   <span data-name="evid" class="CanvasParam" type="text" value="CI-0</span>
1485   <input type="checkbox" value="Fill" checked="">
1486   <input data-name="X" class="CanvasParam" type="text" value="0" omwheel="OnWheelIntRedraw()">
1487   <input data-name="Y" class="CanvasParam" type="text" value="0" omwheel="OnWheelIntRedraw()">
1488   <input data-name="Z" class="CanvasParam" type="text" value="1" omwheel="OnWheelIntRedraw()">
1489   <input data-name="W" class="CanvasParam" type="text" value="1000" omwheel="OnWheelIntRedraw()">
1490   <input data-name="R" class="CanvasParam" type="text" value="1000" omwheel="OnWheelIntRedraw()">
1491   <input data-name="R" class="CanvasParam" type="text" value="0" omwheel="OnWheelIntRedraw()">
1492   <input data-name="C" class="CanvasParam" type="text" value="0">
1493 </div>
1494 <div id="CanvasTool_Rect" class="CanvasTool">
1495   <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
1496   <input data-name="rdr" class="CV_Button" type="button" value="Rect" onclick="CV_drawRect()">
1497   <span data-name="evid" class="CanvasParam" type="text" value="RE-0</span>
1498   <input type="checkbox" value="Fill" checked="">
1499   <input data-name="X" class="CanvasParam" type="text" value="40" omwheel="OnWheelIntRedraw()">
1500   <input data-name="Y" class="CanvasParam" type="text" value="60" omwheel="OnWheelIntRedraw()">
1501   <input data-name="Z" class="CanvasParam" type="text" value="2" omwheel="OnWheelIntRedraw()">
1502   <input data-name="W" class="CanvasParam" type="text" value="120" omwheel="OnWheelIntRedraw()">
1503   <input data-name="R" class="CanvasParam" type="text" value="80" omwheel="OnWheelIntRedraw()">
1504   <input data-name="R" class="CanvasParam" type="text" value="0" omwheel="OnWheelIntRedraw()">
1505   <input data-name="C" class="CanvasParam" type="text" value="1">
1506 </div>
1507 <div id="CanvasTool_Circle" class="CanvasTool">
1508   <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
1509   <input data-name="rdr" class="CV_Button" type="button" value="Circle" onclick="CV_drawCircle()">
1510   <span data-name="evid" class="CanvasParam" type="text" value="CI-0</span>
1511   <input type="checkbox" value="Fill" checked="">
1512   <input data-name="X" class="CanvasParam" type="text" value="100" omwheel="OnWheelIntRedraw()">
1513   <input data-name="Y" class="CanvasParam" type="text" value="100" omwheel="OnWheelIntRedraw()">
1514   <input data-name="Z" class="CanvasParam" type="text" value="3" omwheel="OnWheelIntRedraw()">
1515   <input data-name="W" class="CanvasParam" type="text" value="24" omwheel="OnWheelIntRedraw()">
1516   <input data-name="R" class="CanvasParam" type="text" value="0" omwheel="OnWheelIntRedraw()">
1517   <input data-name="E" class="CanvasParam" type="text" value="360" omwheel="OnWheelIntRedraw()">
1518   <input data-name="C" class="CanvasParam" type="text" value="2">
1519 </div>
1520 <div id="CanvasTool_Packman" class="CanvasTool">
1521   <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
1522   <input data-name="rdr" class="CV_Button" type="button" value="Packman" onclick="CV_drawPackman()">
1523   <span data-name="evid" class="CanvasParam" type="text" value="PA-0</span>
1524   <input type="checkbox" value="Fill" checked="">
1525   <input data-name="X" class="CanvasParam" type="text" value="240" omwheel="OnWheelIntRedraw()">
1526   <input data-name="Y" class="CanvasParam" type="text" value="130" omwheel="OnWheelIntRedraw()">
1527   <input data-name="Z" class="CanvasParam" type="text" value="6" omwheel="OnWheelIntRedraw()">
1528   <input data-name="R" class="CanvasParam" type="text" value="45" omwheel="OnWheelIntRedraw()">
1529   <input data-name="R" class="CanvasParam" type="text" value="0" omwheel="OnWheelIntRedraw()">
1530   <input data-name="B" class="CanvasParam" type="text" value="30" omwheel="OnWheelIntRedraw()">
1531   <input data-name="C" class="CanvasParam" type="text" value="5">
1532 </div>
1533 <div id="CanvasTool_Ellipse" class="CanvasTool">
1534   <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
1535   <input data-name="rdr" class="CV_Button" type="button" value="Ellipse" onclick="CV_drawEllipse()">
1536   <span data-name="evid" class="CanvasParam" type="text" value="EL-0</span>
1537   <input type="checkbox" value="Fill" checked="">
1538   <input data-name="X" class="CanvasParam" type="text" value="40" omwheel="OnWheelIntRedraw()">
1539   <input data-name="Y" class="CanvasParam" type="text" value="30" omwheel="OnWheelIntRedraw()">
1540   <input data-name="Z" class="CanvasParam" type="text" value="2" omwheel="OnWheelIntRedraw()">
1541   <input data-name="R" class="CanvasParam" type="text" value="30" omwheel="OnWheelIntRedraw()">
1542   <input data-name="RY" class="CanvasParam" type="text" value="20" omwheel="OnWheelIntRedraw()">
1543   <input data-name="R" class="CanvasParam" type="text" value="0" omwheel="OnWheelIntRedraw()">
1544   <input data-name="S" class="CanvasParam" type="text" value="0" omwheel="OnWheelIntRedraw()">
1545   <input data-name="E" class="CanvasParam" type="text" value="360" omwheel="OnWheelIntRedraw()">
1546   <input data-name="C" class="CanvasParam" type="text" value="4">
1547 </div>
1548 <div id="CanvasTool_Baloon" class="CanvasTool">
1549   <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
1550   <input data-name="rdr" class="CV_Button" type="button" value="Baloon" onclick="CV_drawBaloon()">
1551   <span data-name="evid" class="CanvasParam" type="text" value="BA-0</span>
1552   <input type="checkbox" value="Fill" checked="">

```

```

13983 <input data-name="X" class="CanvasParam" type="text" value="280" onwheel="OnWheelIntRedraw()">
13984 <input data-name="Y" class="CanvasParam" type="text" value="40" onwheel="OnWheelIntRedraw()">
13985 <input data-name="Z" class="CanvasParam" type="text" value="5" onwheel="OnWheelIntRedraw()">
13986 <input data-name="RX" class="CanvasParam" type="text" value="50" onwheel="OnWheelIntRedraw()">
13987 <input data-name="RY" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()">
13988 <input data-name="RO" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
13989 <input data-name="S" class="CanvasParam" type="text" value="120" onwheel="OnWheelIntRedraw()">
13990 <input data-name="E" class="CanvasParam" type="text" value="100" onwheel="OnWheelIntRedraw()">
13991 <input data-name="C" class="CanvasParam" type="text" value="4">
13992 </div>
13993
13994 <div id="CanvasTool_Piechart" class="CanvasTool">
13995 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13996 <input data-name="rdr" class="CV_Button" type="button" value="Piechart" onclick="CV_drawPiechart()">
13997 <span data-name="evid" class="CanvasParam" type="text" value="BA-0c/span>
13998 <input data-name="F" type="checkbox" value="Fill" checked="">
13999 <input data-name="X" class="CanvasParam" type="text" value="350" onwheel="OnWheelIntRedraw()">
14000 <input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()">
14001 <input data-name="Z" class="CanvasParam" type="text" value="7" onwheel="OnWheelIntRedraw()">
14002 <input data-name="RX" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
14003 <input data-name="RY" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
14004 <input data-name="RO" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
14005 <input data-name="S" class="CanvasParam" type="text" value="210" onwheel="OnWheelIntRedraw()">
14006 <input data-name="E" class="CanvasParam" type="text" value="180" onwheel="OnWheelIntRedraw()">
14007 <input data-name="C" class="CanvasParam" type="text" value="3">
14008 </div>
14009
14010 <div id="CanvasTool_XArc1" class="CanvasTool">
14011 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
14012 <input data-name="rdr" class="CV_Button" type="button" value="XArc" onclick="CV_drawXArc()">
14013 <span data-name="evid" class="CanvasParam" type="text" value="XA-0c/span>
14014 <input data-name="F" type="checkbox" value="Fill" checked="">
14015 <input data-name="X" class="CanvasParam" type="text" value="460" onwheel="OnWheelIntRedraw()">
14016 <input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()">
14017 <input data-name="Z" class="CanvasParam" type="text" value="6" onwheel="OnWheelIntRedraw()">
14018 <input data-name="RX" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
14019 <input data-name="RY" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
14020 <input data-name="RO" class="CanvasParam" type="text" value="150" onwheel="OnWheelIntRedraw()">
14021 <input data-name="S" class="CanvasParam" type="text" value="210" onwheel="OnWheelIntRedraw()">
14022 <input data-name="E" class="CanvasParam" type="text" value="180" onwheel="OnWheelIntRedraw()">
14023 <input data-name="C" class="CanvasParam" type="text" value="4">
14024 </div>
14025
14026 <div id="CanvasTool_XArc2" class="CanvasTool">
14027 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
14028 <input data-name="rdr" class="CV_Button" type="button" value="XArc" onclick="CV_drawXArc()">
14029 <span data-name="evid" class="CanvasParam" type="text" value="XA-1c/span>
14030 <input data-name="F" type="checkbox" value="Fill" checked="">
14031 <input data-name="X" class="CanvasParam" type="text" value="580" onwheel="OnWheelIntRedraw()">
14032 <input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()">
14033 <input data-name="Z" class="CanvasParam" type="text" value="8" onwheel="OnWheelIntRedraw()">
14034 <input data-name="RX" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
14035 <input data-name="RY" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
14036 <input data-name="RO" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
14037 <input data-name="S" class="CanvasParam" type="text" value="210" onwheel="OnWheelIntRedraw()">
14038 <input data-name="E" class="CanvasParam" type="text" value="180" onwheel="OnWheelIntRedraw()">
14039 <input data-name="C" class="CanvasParam" type="text" value="2">
14040 </div>
14041
14042 <canvas id="CV_partsCanvas" data-name="canvas" class="CS_Canvas" width="740" height="200"></canvas>
14043 </span>
14044 </script>
14045
14046 function CV_redrawParts(){
14047 // search Z-Index and sort
14048 var parts = CanvasTools.children;
14049 //console.log("parts="+parts.length);
14050 np = [];
14051 for( i = 0; i < parts.length; i++ ){
14052 p = parts[i];
14053 z = childByName(p,'Z');
14054 if( z != null ){
14055 //console.log("P'+p.id+' z="+z+'.value);
14056 np.push([z.value,p]);
14057 }
14058 np.sort(function(n1,n2){ return n1[0] - n2[0]; });
14059 CV_clearRect();
14060 for( i = 0; i < np.length; i++ ){
14061 p = np[i][1];
14062 redraw = childByName(p,'rdr');
14063 //console.log("Redraw Z="+np[i][0]+' #' +np[i][1].id+' redraw'+redraw.onclick);
14064 if( redraw != null ){
14065 redraw.click();
14066 }
14067 }
14068 }
14069
14070 function DrawingCanvas_Setup(){
14071 showColorSample();
14072 CV_redrawParts();
14073 }
14074
14075 function OnWheelZoom(){
14076 val = OnWheelInt();
14077 canvas = CV_partsCanvas;
14078 canvas.style.zoom = val + '%';
14079 CV_redrawParts();
14080 }
14081
14082 function OnWheelResize(){
14083 val = OnWheelInt();
14084 canvas = CV_partsCanvas;
14085 if( !canvas.hasAttribute('data-width') ){
14086 w = canvas.width;
14087 h = canvas.height;
14088 canvas.setAttribute('data-width',w);
14089 canvas.setAttribute('data-height',h);
14090 sw = canvas.getAttribute('data-width');
14091 sh = canvas.getAttribute('data-height');
14092 console.log("Zoom save original w="+sw+',h='+h+' sw="+sw+', sh="+sh);
14093 }
14094 w = canvas.getAttribute('data-width');
14095 h = canvas.getAttribute('data-height');
14096 console.log("Zoom got original size w="+w+' h="+h);
14097 nw = w * (val/100.0);
14098 nh = h * (val/100.0);
14099 //console.log("Zoom nw="+nw+' nh="+nh);
14100 CV_partsCanvas.width = nw;
14101 CV_partsCanvas.height = nh;
14102 CV_redrawParts();
14103 }
14104
14105 function OnWheelIntRedraw(){
14106 OnWheelInt();
14107 t = event.target;
14108 n = t.nodeName;
14109 i = t.id;
14110 p = t.parentNode;
14111 y = event.deltaY.toFixed(0);
14112 //console.log("OnWheelIntRedraw '+y+' '+n+'#'+i+' '+t.value+' #' +p.id);
14113 if( true ){
14114 CV_redrawParts();
14115 }else{
14116 if( RedrawImmediate.checked ){
14117 CV_clearRect();
14118 //if( p.id == 'CanvasTool_Circle' ){ CV_drawCircle(CanvasTool_Circle); }
14119 //if( p.id == 'CanvasTool_Rect' ){ CV_drawRect(CanvasTool_Rect); }
14120 CV_drawCircle(CanvasTool_Circle);
14121 CV_drawRect(CanvasTool_Rect);
14122 }
14123 }
14124 }
14125
14126 function CV_setCtxStyle(ctx,p){
14127 C = childByName(p,'C').value;
14128 c = document.getElementById('CanvasTool_Color_'+C);
14129 ctx.fillStyle = CV_GenColorStyle(c,true);
14130 ctx.strokeStyle = CV_GenColorStyle(c,false);
14131 return ctx;
14132 }
14133
14134 function CV_clearRect(){
14135 cv = document.getElementById('CV_partsCanvas');
14136 ctx = cv.getContext('2d');
14137 ctx.clearRect(0,0,cv.width,cv.height);
14138 }
14139
14140 function CV_drawRect(rect){
14141 canvas = document.getElementById('CV_partsCanvas');
14142 ctx = canvas.getContext('2d');
14143 ctx = CV_setCtxStyle(ctx,p);
14144 p = rect;
14145 F = childByName(p,'F').checked;
14146 X = childByName(p,'X').value;
14147 Y = childByName(p,'Y').value;
14148 Z = childByName(p,'Z').value;
14149 p.style.zIndex = Z;
14150 W = childByName(p,'W').value;
14151 H = childByName(p,'H').value;
14152 if( F ){
14153 ctx.fillRect(X,Y,W,H);
14154 }else{
14155 ctx.strokeRect(X,Y,W,H);
14156 }
14157 saveDrawing();
14158 }
14159
14160 function CV_drawRect(rect){
14161 CV_drawRect(CanvasTool_Rect);

```



```

14160 }
14161 function CV_drawCircle1(circle){
14162   canvas = document.getElementById('CV_partsCanvas');
14163   ctx = canvas.getContext('2d');
14164   ctx = CV_setCtxStyle(ctx,p);
14165
14166   p = circle;
14167   F = childByName(p,'F').checked;
14168   X = childByName(p,'X').value;
14169   Y = childByName(p,'Y').value;
14170   Z = childByName(p,'Z').value;
14171   p.style.zIndex = Z;
14172   R = childByName(p,'R').value;
14173   S = childByName(p,'S').value;
14174   E = childByName(p,'E').value;
14175
14176   //console.log('Circle'+X+', '+Y+' F='+F);
14177   ctx.beginPath();
14178   SA = (S / 180) * Math.PI;
14179   EA = (E / 180) * Math.PI;
14180   ctx.arc(X,Y,R,SA,EA);
14181   if( F ){
14182     ctx.fill();
14183   }else{
14184     ctx.stroke();
14185   }
14186   saveDrawing();
14187 }
14188 function CV_drawCircle(){
14189   CV_drawCircle1(CanvasTool_Circle);
14190 }
14191 function CV_drawEllipsel(circle){
14192   canvas = document.getElementById('CV_partsCanvas');
14193   ctx = canvas.getContext('2d');
14194   ctx = CV_setCtxStyle(ctx,p);
14195
14196   p = circle;
14197   X = childByName(p,'X').value;
14198   Y = childByName(p,'Y').value;
14199   Z = childByName(p,'Z').value;
14200   RX = childByName(p,'RX').value;
14201   RY = childByName(p,'RY').value;
14202   p.style.zIndex = Z;
14203   RO = childByName(p,'RO').value;
14204   S = childByName(p,'S').value;
14205   E = childByName(p,'E').value;
14206
14207   ctx.beginPath();
14208   SA = (S / 180) * Math.PI;
14209   EA = (E / 180) * Math.PI;
14210   ROA = (RO / 180) * Math.PI;
14211   ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
14212
14213   F = childByName(p,'F').checked;
14214
14215   if( F ){
14216     ctx.fill();
14217   }
14218   // if( S ){
14219   {
14220     ctx.stroke();
14221   }
14222   saveDrawing();
14223 }
14224 function CV_drawEllipse(){
14225   CV_drawEllipsel(CanvasTool_Ellipse);
14226 }
14227 function CV_drawBalloon1(baloon){
14228   canvas = document.getElementById('CV_partsCanvas');
14229   ctx = canvas.getContext('2d');
14230   ctx = CV_setCtxStyle(ctx,p);
14231
14232   p = baloon;
14233   F = childByName(p,'F').checked;
14234   X = childByName(p,'X').value;
14235   Y = childByName(p,'Y').value;
14236   Z = childByName(p,'Z').value;
14237   RX = childByName(p,'RX').value;
14238   RY = childByName(p,'RY').value;
14239   p.style.zIndex = Z;
14240   RO = childByName(p,'RO').value;
14241   S = childByName(p,'S').value;
14242   E = childByName(p,'E').value;
14243
14244   //console.log('Ellipse'+X+', '+Y+' F='+F);
14245   ctx.beginPath();
14246
14247   SA = (S / 180) * Math.PI;
14248   EA = (E / 180) * Math.PI;
14249   ROA = (RO / 180) * Math.PI;
14250   ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
14251
14252   PX = parseInt(X);
14253   PY = parseInt(Y);
14254   //console.log('Ellipse A '+PX+', '+PY+' F='+F);
14255   FX = 25;
14256   FY = 40;
14257   //console.log('Ellipse B '+PX+', '+PY+' F='+F);
14258   ctx.lineTo(PX,PY);
14259   ctx.closePath();
14260
14261   if( F ){
14262     ctx.fill();
14263   }else{
14264     ctx.stroke();
14265   }
14266   saveDrawing();
14267 }
14268 function CV_drawBalloon(){
14269   CV_drawBalloon1(CanvasTool_Balloon);
14270 }
14271 function CV_drawPackman1(circle){
14272   canvas = document.getElementById('CV_partsCanvas');
14273   ctx = canvas.getContext('2d');
14274   ctx = CV_setCtxStyle(ctx,p);
14275
14276   p = circle;
14277   F = childByName(p,'F').checked;
14278   X = childByName(p,'X').value;
14279   Y = childByName(p,'Y').value;
14280   Z = childByName(p,'Z').value;
14281   p.style.zIndex = Z;
14282   R = childByName(p,'R').value;
14283   S = childByName(p,'S').value;
14284   E = childByName(p,'E').value;
14285
14286   //console.log('Packman'+X+', '+Y+' F='+F);
14287   ctx.beginPath();
14288   SA = (S / 180) * Math.PI;
14289   //SA += 0.15 * Math.PI;
14290   EA = SA + Math.PI; //(E / 180) * Math.PI;
14291   ctx.arc(X,Y,R,SA,EA);
14292   if( F ){ ctx.fill(); }else{ ctx.stroke(); }
14293
14294   ctx.beginPath();
14295   E = 180 - E;
14296   SA += (E/180) * Math.PI;
14297   EA += (E/180) * Math.PI;
14298   ctx.arc(X,Y,R,SA,EA);
14299   if( F ){ ctx.fill(); }else{ ctx.stroke(); }
14300
14301   saveDrawing();
14302 }
14303 function CV_drawPackman(){
14304   CV_drawPackman1(CanvasTool_Packman);
14305 }
14306 function CV_drawPiechart1(baloon){
14307   canvas = document.getElementById('CV_partsCanvas');
14308   ctx = canvas.getContext('2d');
14309   ctx = CV_setCtxStyle(ctx,p);
14310
14311   p = baloon;
14312   F = childByName(p,'F').checked;
14313   X = childByName(p,'X').value;
14314   Y = childByName(p,'Y').value;
14315   Z = childByName(p,'Z').value;
14316   RX = childByName(p,'RX').value;
14317   RY = childByName(p,'RY').value;
14318   p.style.zIndex = Z;
14319   RO = childByName(p,'RO').value;
14320   S = childByName(p,'S').value;
14321   E = childByName(p,'E').value;
14322
14323   //console.log('Ellipse'+X+', '+Y+' F='+F);
14324   ctx.beginPath();
14325
14326   SA = (S / 180) * Math.PI;
14327   EA = (E / 180) * Math.PI;
14328   ROA = (RO / 180) * Math.PI;
14329   ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
14330
14331   PX = parseInt(X);
14332   PY = parseInt(Y);
14333   //console.log('Ellipse A '+PX+', '+PY+' F='+F);
14334   //FX = 25;
14335   //PY = 40;
14336   //console.log('Ellipse B '+PX+', '+PY+' F='+F);

```

```

14337 ctx.lineTo(PX,PY);
14338 ctx.closePath();
14339
14340 if( F ){
14341   ctx.fill();
14342 }else{
14343   ctx.stroke();
14344 }
14345 saveDrawing();
14346 }
14347
14348 function CV_drawPiechart(){
14349   CV_drawPiechart(CanvasTool_Piechart);
14350 }
14351
14352 function CV_drawArc1(baloon){
14353   canvas = document.getElementById('CV_partsCanvas');
14354   ctx = canvas.getContext('2d');
14355   ctx = CV_setCtxStyle(ctx,p);
14356
14357   p = baloon;
14358   F = childByName(p,'F').checked;
14359   X = childByName(p,'X').value;
14360   Y = childByName(p,'Y').value;
14361   Z = childByName(p,'Z').value;
14362   RX = childByName(p,'RX').value;
14363   RY = childByName(p,'RY').value;
14364   p.style.zIndex = Z;
14365   RO = childByName(p,'RO').value;
14366   S = childByName(p,'S').value;
14367   E = childByName(p,'E').value;
14368
14369   //console.log('Ellipse'+X+', '+Y+' F='+F);
14370   ctx.beginPath();
14371
14372   if( true ){
14373     d = new Date();
14374     ms = d.getTime();
14375     id = (ms % 300) / 10;
14376     //S = parseFloat(F) * id/2;
14377     E = parseFloat(E) - id;
14378     xd = (ms % 10000) / 5;
14379     if( id <= xd ){
14380       xd = 2000 - xd;
14381     }
14382     xd *= 0.8;
14383     X = parseFloat(X) + xd - 600;
14384     //console.log('Ellipse S='+S+', E='+E+' id='+id);
14385
14386     SA = (S / 180) * Math.PI;
14387     EA = (E / 180) * Math.PI;
14388     ROA = (RO / 180) * Math.PI;
14389     ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
14390
14391     PX = parseInt(X);
14392     PY = parseInt(Y);
14393
14394     //console.log('Ellipse A '+PX+', '+PY+' F='+F);
14395     //console.log('Ellipse B '+PX+', '+PY+' F='+F);
14396
14397     ctx.lineTo(PX,PY);
14398     ctx.closePath();
14399
14400     if( F ){
14401       ctx.fill();
14402     }else{
14403       ctx.stroke();
14404     }
14405     saveDrawing();
14406   }
14407
14408   function CV_drawArc1(){
14409     t = event.target;
14410     CV_drawArc1(t.parentNode);
14411   }
14412   var AnimateIvl = window.setInterval(CV_redrawParts,30);
14413 }
14414 </script>
14415
14416 <h3>Animation</h3>
14417 <span id="Animation" class="CanvasTool" draggable="true" contenteditable="">
14418   <input class="CV_Button" type="button" value="Clone" onclick="cloneParent()">
14419   <input class="CV_Button" type="button" value="Clear" onclick="CV_clearRect()">
14420   <span data-name="evid" class="CanvasParam" type="text" value="ANIMA-0"></span>
14421   <input data-name="T" class="ColorParam" type="text" value="Rotate">
14422   <input data-name="r" class="CanvasParam" type="text" value="30" onwheel="OnWheelInt()">ms
14423   <input data-name="DURA" class="CanvasParam" type="text" value="10" onwheel="OnWheelInt()">s
14424 </span>
14425
14426 <h3>Canvas</h3>
14427 <span id="AppendToCanvas" class="CanvasTool" draggable="true">
14428   <input class="CV_Button" type="button" value="Append"> the above part
14429 </span>
14430 <span id="CanvasWrapTemplate" class="CanvasWrap" draggable="true">
14431   <input class="CV_Button" type="button" value="Clone" onclick="cloneParent()">
14432   <input class="CV_Button" type="button" value="Clear" onclick="CV_clearRect()">
14433   <span data-name="width" class="CanvasParam" type="text" value="700">
14434   <input data-name="height" class="CanvasParam" type="text" onchange="CS_setSize()" value="400">
14435   <input data-name="zoom" class="CanvasParam" type="text" onchange="CS_setSize()" value="100" onwheel="OnWheelCanvasZoom()">
14436   <input class="CV_Button" type="button" value="Remove" onclick="removeParent()"><br>
14437   <canvas id="DrawingCanvas" data-name="canvas" class="CS_Canvas" width="700" height="400"></canvas>
14438 </span>
14439 </script>
14440 function OnWheelCanvasZoom(){
14441   val = OnWheelInt();
14442   DrawingCanvas.style.zoom = val + '%';
14443   //CanvasWrapTemplate.style.width = (700*(val/100.0)+20) + 'px';
14444   CanvasWrapTemplate.style.height = (400*(val/100.0)+30) + 'px';
14445 }
14446 </script>
14447
14448 <h3>CanvasBook</h3>
14449 <div id="CanvasBook"></div>
14450 <br>
14451 <style>
14452 .CanvasBook {
14453   overflow:scroll;
14454 }
14455 .CBPanel {
14456 }
14457 .CanvasTool {
14458   font-size:9pt;
14459   font-family:Courier New;
14460   color:#000 !important;
14461   xborder:1px solid #aaa;
14462   background-color:rgba(127,127,127,0.5);
14463   width:740px;
14464   white-space:nowrap;
14465   xheight:30;
14466   margin:4px;
14467   padding-left:4px;
14468   padding-right:4px;
14469   overflow:auto;
14470   display:inline-block;
14471   resize:both;
14472   vertical-align:top;
14473   zoom:1.0;
14474 }
14475 .CanvasWrap {
14476   font-size:9pt;
14477   font-family:Courier New;
14478   color:#000 !important;
14479   border:1px solid #aaa;
14480   background-color:rgba(200,200,200,0.2);
14481   width:740px;
14482   height:430px;
14483   margin:4px;
14484   padding-left:4px;
14485   padding-right:4px;
14486   overflow:auto;
14487   display:inline-block;
14488   resize:both;
14489   vertical-align:top;
14490   zoom:1.0;
14491 }
14492 .CS_Panel {
14493   padding:3px;
14494   vertical-align:middle;
14495 }
14496 .CS_Canvas {
14497   border:1px dashed #fcc;
14498   resize:both;
14499   zoom:1.0;
14500 }
14501 </style>
14502 <script>
14503 var CanvasID = 0;
14504 function removeParent(){
14505   e = event.target;
14506   p = e.parentNode;
14507   if( p == CanvasWrapTemplate ){
14508     return;
14509   }
14510   pp = p.parentNode;
14511   alert('removeParent #' + pp.id + '/' + p.id + '/' + # + e.id);
14512   p.parentNode.removeChild(p);
14513 }

```

```

14514 function cloneParent(){
14515     b = event.target;
14516     w = b.parentNode;
14517     CanvasID += 1;
14518     //pp = w.parentNode;
14519     cw = w.cloneNode(true);
14520     cw.id = 'Canvas' + CanvasID;
14521     childByName(cw, 'cvid').innerHTML = CanvasID;
14522     childByName(cw, 'cvid').value = CanvasID;
14523     CanvasBook.appendChild(cw);
14524 }
14525 function CS_resize(){
14526     e = event.target;
14527     p = e.parentNode;
14528     console.log('resize '+e.nodeName+' '+p.nodeName);
14529     c = childByName(p, 'canvas');
14530     w = childByName(p, 'width').value;
14531     h = childByName(p, 'height').value;
14532     console.log('resize '+c.nodeName+' '+w+' '+h);
14533     c.width = w;
14534     c.height = h;
14535     p.style.width = w + 'px';
14536     p.style.height = h + 'px';
14537     console.log("c='"+c+"' +w+'/'+'+c.width+' '+h+'/'+'+c.height);
14538 }
14539 function CS_newFunc(){
14540     cvt = CanvasWrapTemplate;
14541     cw = CanvasWrapTemplate.cloneNode(true);//needs an argument, otherwise 'functiuon notfound'
14542     CanvasID += 1;
14543     cw.id = 'Canvas' + CanvasID;
14544     //childByName(cw, 'cvid').contentEditable = false;
14545     childByName(cw, 'cvid').innerHTML = CanvasID;
14546     childByName(cw, 'cvid').value = CanvasID;
14547     childByName(cw, 'width').value = w = childByName(cvt, 'width').value;
14548     childByName(cw, 'height').value = h = childByName(cvt, 'height').value;
14549     cw.style.width = w + 'px';
14550     //ncv = document.createElement('canvas');
14551     ncv = childByName(cw, 'canvas');
14552     //ncv.setAttribute('class', 'CS_Canvas');
14553     //ncv.setAttribute('data-name', 'canvas');
14554     ncv.width = w;
14555     ncv.height = h;
14556     //cvt.replaceChild(ncv, childByName(cw, 'canvas'));
14557     CanvasBook.appendChild(cw);
14558 }
14559 //CS_new.addEventListener('click', CS_newFunc);
14560 </script>
14561
14562 <input id="CanvasBook_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14563 <input id="CanvasBook_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14564 <input id="CanvasBook_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14565 <span id="CanvasBook_WorkCodeView"></span>
14566 <script id="CanvasBook_WorkScript">
14567 function CanvasBook_openWorkCodeView(){
14568     function CanvasBook_showWorkCode(){
14569         showHtmlCode(CanvasBook_WorkCodeView, CascadedCanvasBook_WorkCodeSpan);
14570     }
14571     CanvasBook_WorkCodeViewOpen.addEventListener('click', CanvasBook_showWorkCode);
14572 }
14573 CanvasBook_openWorkCodeView(); // should be invoked by an event
14574 </script>
14575 </details>
14576 <!-- CanvasBook_WorkCodeSpan -->
14577 </span>
14578 <!-- Work -->
14579
14580
14581
14582 <!-- Work { -->
14583 </span id="SVG_WorkCodeSpan" open=""><a href="#SVG">SVG</a></span></span>
14584 </span id="SVGFacesSection"><summary id="SVGFacesSummary">SVG Getting Started</summary>
14585 <!-- SVG // 2020-1108 SatoxITS -->
14586 <h2>Getting started SVG</h2>
14587
14588 <div>
14589 <svg id="xSVG_01" class="SVG100">
14590 <circle cx="50" cy="50" r="40" stroke="white" stroke-width="0" fill="yellow"></circle>
14591 <circle cx="35" cy="35" r="05" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14592 <circle cx="65" cy="35" r="05" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14593 <circle cx="50" cy="60" r="20" stroke="black" stroke-width="2" fill="#00000000"></circle>
14594 </svg>
14595
14596 <svg id="xSVG_02" class="SVG100" viewBox="0 0 100 100">
14597 <circle cx="50" cy="50" r="40" stroke="white" stroke-width="0" fill="yellow"></circle>
14598 <circle cx="35" cy="35" r="05" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14599 <circle cx="65" cy="35" r="05" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14600 <circle cx="50" cy="60" r="20" stroke="black" stroke-width="2" fill="#00000000"></circle>
14601 </svg>
14602
14603 <svg id="xSVG_03" class="SVG100" viewBox="0 0 100 100">
14604 <circle cx="50" cy="50" r="40" stroke="white" stroke-width="0" fill="yellow"></circle>
14605 <circle cx="35" cy="35" r="05" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14606 <circle cx="65" cy="35" r="05" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14607 <path fill="orange" d="M 25 50 A 10 10 0 0 0 75 50 Z"></path>
14608 </svg>
14609
14610 <svg id="xSVG_04" class="SVG100" viewBox="0 0 100 100">
14611 <circle cx="50" cy="50" r="40" stroke="white" stroke-width="0" fill="yellow"></circle>
14612 <circle cx="35" cy="35" r="05" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14613 <circle cx="65" cy="35" r="05" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14614 <path fill="orange" d="M 25 50 A 10 10 0 0 0 75 50 Z"></path>
14615 </svg>
14616
14617 <svg id="xSVG_05" class="SVG100" viewBox="0 0 100 100">
14618 <circle cx="50" cy="50" r="40" stroke="black" stroke-width="1" fill="ffffff00"></circle>
14619 <circle cx="38" cy="42" r="04" stroke="black" stroke-width="1" fill="#00000000"></circle>
14620 <circle cx="62" cy="42" r="04" stroke="black" stroke-width="1" fill="#00000000"></circle>
14621 <path stroke="black" d="M 20 50 A 10 10 0 0 80 50" fill="ffffff00"></path>
14622 </div>
14623
14624 <style>
14625 SVG100 {
14626     width:100px;
14627     height:100px;
14628 }
14629 </style>
14630 <script>
14631 var svg_deg = 0;
14632 var SVGFacesInterval = 0;
14633 function rotateSVGFaces(){
14634     if( SVGFacesSection.open != true ){
14635         clearInterval(SVGFacesInterval);
14636         SVGFacesInterval = 0;
14637         return;
14638     }
14639     //ms = new Date().getTime();
14640     //deg = (ms.toFixed(0)/10) % 360;
14641     svg_deg += 2;
14642     xSVG_04.style.transform = 'rotate('+svg_deg+'deg)';
14643     xSVG_05.style.transform = 'rotate('+svg_deg+'deg)';
14644 }
14645 function SVGFaces_Setup(){
14646     function startStopSVGFaces(){
14647         //if( SVGFacesSection.open == true ){ // not yet open when clicked
14648             if( SVGFacesInterval == 0 ){
14649                 SVGFacesInterval = window.setInterval(rotateSVGFaces,100);
14650             }
14651         }
14652     }
14653     SVGFacesSummary.addEventListener('click', startStopSVGFaces);
14654     document.addEventListener('load', startStopSVGFaces);
14655 }
14656 SVGFaces_Setup();
14657 </script>
14658
14659 <input id="SVG_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14660 <input id="SVG_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14661 <input id="SVG_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14662 <span id="SVG_WorkCodeView"></span>
14663 <script id="SVG_WorkScript">
14664 function SVG_openWorkCodeView(){
14665     function SVG_showWorkCode(){
14666         showHtmlCode(SVG_WorkCodeView, SVG_WorkCodeSpan);
14667     }
14668     SVG_WorkCodeViewOpen.addEventListener('click', SVG_showWorkCode);
14669 }
14670 SVG_openWorkCodeView(); // should be invoked by an event
14671 </script>
14672 </details>
14673 <!-- SVG_WorkCodeSpan -->
14674 </span>
14675 <!-- Work -->
14676
14677
14678
14679 <!-- Work { -->
14680 </span id="XYyes_WorkCodeSpan">
14681 </span id="XYyesSection"><summary id="XYyesSummary">XYyes</summary>
14682 <a href="#XYyes">XYyes</a></span></span>
14683 <!-- XYyes // 2020-1113 SatoxITS -->
14684 <h2>XYyes</h2>
14685
14686 <style>
14687 SVG100b { width:100px; height:100px; }
14688 </style>

```

```

1469 </style>
1469 <div id="XYeyes1">
1469 <svg class="SVG100b" viewBox="0 0 100 100">
1469 <circle cx="50" cy="50" r="40" stroke="black" stroke-width="1" fill="ffff00ff"></circle>
1469 <circle cx="38" cy="42" r="4" stroke="black" stroke-width="1" fill="#000000ff"></circle>
1469 <circle cx="62" cy="42" r="4" stroke="black" stroke-width="1" fill="#000000ff"></circle>
1469 <path stroke="black" d="M 20 50 A 10 10 0 0 0 80 50" fill="fffff00"></path>
1469 </svg>
1469 <svg class="SVG100b" viewBox="0 0 100 100">
1470 <circle cx="50" cy="50" r="40" fill="#f00000ff"></circle>
1470 <circle cx="50" cy="50" r="20" fill="ffffffff"></circle>
1470 <circle cx="50" cy="50" r="10" fill="#0000ffff"></circle>
1470 </svg>
1470 <svg class="SVG100b" viewBox="0 0 100 100">
1470 <circle cx="50" cy="50" r="40" fill="#d00000ff"></circle>
1470 <circle cx="66" cy="50" r="20" fill="ffffffff"></circle>
1470 <circle cx="76" cy="50" r="10" fill="#0000ffff"></circle>
1470 </svg>
1470 <svg class="SVG100b" viewBox="0 0 100 100">
1471 <circle cx="50" cy="50" r="40" fill="#d00000ff"></circle>
1471 <circle cx="72" cy="34" r="10" fill="#0000ffff"></circle>
1471 </svg>
1471 </div>
1471 <div>
1471 <input id="XYeyes_0_0" class="SVG100b" viewBox="0 0 100 100">
1471 <circle cx="50" cy="50" r="40" fill="f00000ff"></circle>
1471 <circle cx="50" cy="34" r="20" fill="ffffffff"></circle>
1471 <circle cx="50" cy="24" r="10" fill="#0000ffff"></circle>
1471 </input>
1472 <input id="XYeyes_0_1" class="SVG100b" viewBox="0 0 100 100">
1472 <circle cx="50" cy="50" r="40" fill="f00000ff"></circle>
1472 <circle cx="50" cy="34" r="20" fill="ffffffff"></circle>
1472 <circle cx="50" cy="24" r="10" fill="#0000ffff"></circle>
1472 </input>
1472 <div class="XYZvalues">
1472 <input id="XYeyes_0_Move" class="XYZcheckbox" type="checkbox" checked="">Move
1472 <input id="XYeyes_0_Cont" class="XYZcheckbox" type="checkbox" checked="">Cont
1473 <input id="XYeyes_0_Degree" class="XYZvalue1" type="text" value="0"></input>
1473 <input id="XYeyes_0_dist0" class="XYZvalue1" type="text" value="0"></input>
1473 <input id="XYeyes_0_dist1" class="XYZvalue1" type="text" value="0"></input>
1473 </div>
1473 </div>
1473 <script>
1473 function XYeyes_Setup0(){
1473 function lookat(eye,pos,x,y){
1473 x0 = eye.getBoundingClientRect().left.toFixed(0);
1473 y0 = eye.getBoundingClientRect().top.toFixed(0);
1473 pos.value = x0 + ',' + y0;
1473 }
1473 var rad = 0;
1473 function rotate(){
1473 if( XYeyesSection.open != true ){
1473 clearInterval(XYinterval);
1473 console.log('interval cleared');
1473 return;
1473 }
1473 if( !XYeyes_0_Move.checked ){
1473 return;
1473 }
1473 if( XYeyes_0_Cont.checked ){
1473 rad += 2;
1473 }else{
1473 rad += 12;
1473 }
1473 XYeyes_0_Degree.value = rad % 360;
1473 XYeyes_0_0.style.transform = 'rotate('+rad+'deg)';
1473 XYeyes_0_1.style.transform = 'rotate('+rad+'deg)';
1473 lookat(XYeyes_0_0,XYeyes_0_dist0,x,y);
1473 lookat(XYeyes_0_1,XYeyes_0_dist1,x,y);
1473 }
1473 function startStopXYeyes(){
1473 if( XYeyesSection.open == true ){
1473 XYinterval = window.setInterval(rotate,100);
1473 }
1473 }
1473 XYeyesSummary.addEventListener('click',startStopXYeyes);
1473 document.addEventListener('load',startStopXYeyes);
1473 }
1473 XYeyes_Setup0();
1473 </script>
1473 <div id="XYeyes_1" class="XYeyes">
1473 <h3>XY eyes</h3>
1473 <svg id="XYeyes_1_x0" class="SVG100e" viewBox="0 0 100 100">
1473 <circle cx="50" cy="50" r="40" stroke="black" stroke-width="6" fill="fffff00"></circle>
1473 <circle id="XYeyes_1_x0p" cx="50" cy="30" r="20" fill="#000000ff"></circle>
1473 </svg>
1473 <svg id="XYeyes_1_x1" class="SVG100f" viewBox="0 0 100 100">
1473 <circle cx="50" cy="50" r="40" stroke="black" stroke-width="6" fill="fffff00"></circle>
1473 <circle id="XYeyes_1_x1p" cx="50" cy="30" r="20" fill="#000000ff"></circle>
1473 </svg>
1473 <svg id="XYeyes_1_0" class="SVG100c" viewBox="0 0 100 100">
1473 <circle cx="50" cy="50" r="40" fill="f00000ff"></circle>
1473 <circle cx="50" cy="34" r="20" fill="ffffffff"></circle>
1473 <circle cx="50" cy="24" r="10" fill="#0000ffff"></circle>
1473 </svg>
1473 <svg id="XYeyes_1_1" class="SVG100c" viewBox="0 0 100 100">
1473 <circle data-name="ball" cx="50" cy="50" r="40" fill="f00000ff"></circle>
1473 <circle id="XYeyes_1_1_whyeye" data-name="whyeye" cx="50" cy="34" r="20" fill="ffffffff"></circle>
1473 <circle id="XYeyes_1_1_bleye" data-name="bleye" cx="50" cy="24" r="10" fill="#000000ff"></circle>
1473 </svg>
1473 <svg id="XYeyes_1_2" class="SVG100c" viewBox="0 0 100 100">
1473 <circle data-name="ball" cx="50" cy="50" r="40" fill="f00000ff"></circle>
1473 <circle id="XYeyes_1_2_whyeye" data-name="whyeye" cx="50" cy="34" r="20" fill="ffffffff"></circle>
1473 <circle id="XYeyes_1_2_bleye" data-name="bleye" cx="50" cy="24" r="10" fill="#000000ff"></circle>
1473 </svg>
1480 <div class="XYZvalues">
1480 <input id="XYeyes_1_Move" class="XYZcheckbox" type="checkbox" checked="">Move
1480 <input id="XYeyes_1_Cont" class="XYZcheckbox" type="checkbox" checked="">Cont
1480 <input id="XYeyes_1_MousePos" class="XYZvalue1" type="text" value="0"></input>
1480 <input id="XYeyes_1_dist0" class="XYZvalue1" type="text" value="0"></input>
1480 <input id="XYeyes_1_distx1" class="XYZvalue1" type="text" value="0"></input>
1480 <input id="XYeyes_1_dist0" class="XYZvalue1" type="text" value="0"></input>
1480 <input id="XYeyes_1_dist1" class="XYZvalue1" type="text" value="0"></input>
1480 <input id="XYeyes_1_dist2" class="XYZvalue1" type="text" value="0"></input>
1480 </div>
1481 </div>
1481 <style>
1481 .SVG100e { width:100px; height:100px; display:inline; }
1481 .SVG100e { width:100px; height:100px; display:inline; }
1481 .SVG100f { width:100px; height:100px; position:relative; left:-20px; display:inline; }
1481 .XYeyes {
1481 text-align:left;
1481 }
1481 .XYZcheckbox {
1481 width:20px;
1481 height:20px;
1481 }
1481 .XYZvalues {
1481 z-index:2;
1481 display:block;
1481 background-color:#eee;
1481 height:100px;
1481 text-align:left;
1481 }
1481 .XYZvalue1 {
1481 font-size:10pt !important;
1481 width:70pt !important;
1481 line-height:1.1 !important;
1481 margin:1px !important;
1481 display:inline !important;
1481 text-align:right;
1481 padding:1px !important;
1481 }
1481 </style>
1481 <script>
1481 var XYmid = Math.random();
1481 var XYsent = 0;
1481 var XYrecv = 0;
1481 function XYeyes_recvMousePosition(msg){
1481 argv = msg.split(' ');
1481 if( argv[0] == XYmid ){
1481 //console.log('XYeyes echo '+msg);
1481 }else{
1481 XYrecv += 1;
1481 //console.log('XYeyes recv '+XYrecv+' '+msg);
1481 sx = argv[1];
1481 sy = argv[2];
1481 x = sx - window.screenX;
1481 y = sy - window.screenY;
1481 XYeyes_1_MousePos.value = 'R ' + x + ',' + y;
1481 XYlookat(XYeyes_1_x0,XYeyes_1_dist0,x,y);
1481 XYlookat(XYeyes_1_x1,XYeyes_1_dist1,x,y);
1481 }
1481 }
1481 var xye_gjlinked = false;
1481 function XYeyes_sendMousePosition(event){
1481 if( xye_gjlinked == false ){
1481 GJ_Join();
1481 xye_gjlinked = true;
1481 }
1481 x = event.x;

```

```

14868 y = event.y;
14869 sx = window.screenX + x;
14870 sy = window.screenY + y;
14871 msg = 'MOUSEPOSITION '+XYmid+', '+sx+', '+sy;
14872 GJ_BcastMessage(msg);
14873 XYsent += 1;
14874 //console.log('XYes sent '+XYsent+' '+msg);
14875 }
14876 function XYlookat(eye,pos,x,y){
14877 x0 = eye.getBoundingClientRect().left.toFixed(0);
14878 y0 = eye.getBoundingClientRect().top.toFixed(0);
14879
14880 dx = x - x0;
14881 dx = 65;
14882 dy = y - y0;
14883 dy -= 75;
14884
14885 degx = Math.acos(dx/Math.sqrt(dx*dx+dy*dy));
14886 degx *= 180 / Math.PI;
14887 degx = degx.toFixed(0);
14888 degx = parseInt(degx);
14889
14890 if( dx <= 0 && dy <= 0 ){
14891   deg1 = -45;
14892   degx = -(degx - 90);
14893 }else
14894 if( dx <= 0 && 0 < dy ){
14895   deg1 = -135;
14896   degx = degx + 90;
14897 }else
14898 if( 0 < dx && dy <= 0 ){
14899   deg1 = 45;
14900   degx = 90 - degx;
14901 }else{
14902   deg1 = 135;
14903   degx = degx + 90;
14904 }
14905 if( XYeses_1.Cont.checked ){
14906   deg1 = degx;
14907 }
14908 eye.style.transform = 'rotate('+deg1+'deg)';
14909 //pos.value = dx+', '+dy+', '+deg1+' '+degx;
14910 pos.value = dx+', '+dy+', '+deg1;
14911 }
14912 }
14913 function XYeses_Setup1(){
14914   function motion(){
14915     XYeses_sendMousePosition(event);
14916     if( !XYeses_1.Move.checked ){
14917       return;
14918     }
14919     x = event.x;
14920     y = event.y;
14921     XYeses_1.MousePos.value = x + ', ' + y;
14922     XYlookat(XYeses_1_0,XYeses_1_dist0,x,y);
14923     XYlookat(XYeses_1_1,XYeses_1_dist1,x,y);
14924     XYlookat(XYeses_1_2,XYeses_1_dist2,x,y);
14925     XYlookat(XYeses_1_x0,XYeses_1_distx0,x,y);
14926     XYlookat(XYeses_1_x1,XYeses_1_distx1,x,y);
14927   }
14928   window.addEventListener('mousemove',motion);
14929   window.addEventListener('mouseover',motion);
14930   window.addEventListener('focusin',motion);
14931   window.addEventListener('focusout',motion);
14932   fi = document.getElementById('FeaturesView');
14933   if( fi != null ){
14934     fi.appendChild(XYeses_1);
14935   }
14936 }
14937 XYeses_Setup1();
14938 </script>
14939
14940 <input id="XYeses_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14941 <input id="XYeses_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14942 <input id="XYeses_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14943 <span id="XYeses_WorkCodeView"></span>
14944 <script id="XYeses_WorkScript">
14945 function XYeses_openWorkCodeView(){
14946   function XYeses_showWorkCode(){
14947     showHtmlCode(XYeses_WorkCodeView,XYeses_WorkCodeSpan);
14948   }
14949   XYeses_WorkCodeViewOpen.addEventListener('click',XYeses_showWorkCode);
14950 }
14951 XYeses_openWorkCodeView(); // should be invoked by an event
14952 </script>
14953 </details>
14954 <!-- XYeses_WorkCodeSpan -->
14955 <!-- ===== Work } ===== -->
14956 </span id="X3DOM_WorkCodeSpan"><a href="#X3DOM">X3DOM</a></a name="X3DOM">X3DOM</a>
14957 </span id="X3DOM_Summary"><summary id="X3DOM_Summary">X3DOM Getting Started</summary>
14958 <!-- ===== X3DOM // 2020-1111 SatoxITS { -->
14959 <h2>X3D-ROFL</h2>
14960 <!-- X3dom JavaScript and CSS BEGIN -->
14961 <h3><a href="/gshell/x3dom-1.8.2-dev/">x3dom-1.8.2-dev</a></h3>
14962
14963 <div>
14964   <input id="X3dRofl_ViewPosition_Value" class="X3DOM_Param" type="text" value="0 0 0">
14965   <input id="X3dRofl_ViewOrientation_Value" class="X3DOM_Param" type="text" value="0 0 0">
14966 </div>
14967 <div>
14968   <input id="X3dRofl_BoxPosition" class="X3DOM_Param" type="text" value="0 0 0 0">
14969   <input id="X3dRofl_BoxRotation_Value" class="X3DOM_Param" type="text" value="0 0 0 0">
14970 </div>
14971 <span id="x3domInclude">
14972 <script type="text/javascript" src="/gshell/x3dom-1.8.2-dev/x3dom.js"></script>
14973 <link rel="stylesheet" type="text/css" href="/gshell/x3dom-1.8.2-dev/x3dom.css">
14974 </span>
14975 <!-- X3dom JavaScript and CSS JEND -->
14976
14977 <div class="x3dRofl">
14978 <x3d id="Box1View" class="x3dRoflScene" width="600px" height="300px" draggable="false">
14979 <scene id="Box1Scene">
14980 <viewpoint id="Box1ViewPoint" position="0 0 7 4"></viewpoint>
14981 <transform id="Box1Trans" translation="1 0 5 5" rotation="1 1 1 1">
14982 <shape id="xxBox1Box">
14983 <appearance>
14984 <!--
14985 <script>putTextureTag("Box1Texture");</script>
14986 <imageTexture id="Box1Texture" url="GshellInside00.png"></imageTexture>
14987 -->
14988 <imageTexture id="Box1Texture" url="WD-WallPaper03.png"></imageTexture>
14989 <material id="Box1Material" diffusecolor="0 1 0"></material>
14990 </appearance>
14991 <box id="xbox1Box" size="1 1 2"></box>
14992 </shape>
14993 </transform>
14994 </scene>
14995 </x3d>
14996 </div>
14997 <style>
14998 .x3dRofl {
14999   background-color:#e0f0e0;
15000   position:relative;
15001   display:block;
15002   overflow:visible !important;
15003   resize:both !important;
15004 }
15005 .x3dRoflScene {
15006   z-index:100;
15007   background-color:#e0fff0;
15008   position:relative;
15009   display:block;
15010   overflow:visible !important;
15011   resize:both !important;
15012   xzoom:2.0;
15013   yzoom:2.0;
15014   xtransform:scale(2.0);
15015 }
15016 .X3DOM_Param {
15017   color:#00 !important;
15018   font-family:Courier New, Monospace !important;
15019   font-size:9pt !important;
15020   padding:2px !important;
15021   line-height:1.1 !important;
15022   height:14pt !important;
15023   width:120pt !important;
15024   text-align:left !important;
15025   display:inline !important;
15026   vertical-align:middle !important;
15027 }
15028 </style>
15029 </script>

```

```

15045//<transform translation="0 0 0">
15046//<transform id="Box1Box" translation="0 0 0">
15047//</transform>
15048function showVP(){
15049// console.log('color0:'+Box1Material.getAttribute('diffusecolor'));
15050 m = document.getElementById('Box1Material');
15051 //console.log('color1:'+Box1Material.getFieldValue('diffusecolor'));
15052// console.log('color1:'+m.getFieldValue('diffusecolor'));
15053 //console.log('rotation:'+Box1Box.getFieldValue('rotation'));
15054// console.log('Box1TransRotation:'+Box1Trans.getFieldValue('rotation'));
15055
15056 e = document.getElementById('Box1View');
15057// console.log("Box1View:"+e.runtime.properties());
15058 e = document.getElementById('X3dRof1_ViewPoint');
15059// console.log("Box1Box:"+e.outerHTML);
15060
15061 //e.runtime.showAll();
15062 //e.runtime.resetView();
15063
15064 //console.log('Transform'+X3dom.runtime.getCurrentTransform(Box1View));
15065 //console.log('Transform'+Box1View.runtime.getCurrentTransform());
15066 //console.log('Transform'+Box1Scene.runtime.getCurrentTransform());
15067 //console.log('VPosition'+X3dRof1_ViewPoint.position);
15068// console.log('VPosition'+X3dRof1_ViewPoint.getFieldValue('position'));
15069// console.log('VOrientations'+X3dRof1_ViewPoint.getFieldValue('orientation'));
15070//console.log('VPositionProp'+Box1View.runtime.properties());
15071 //console.log('VPositionProp'+X3dRof1_ViewPoint.runtime.properties());
15072 //sid = document.getElementById('Box1Scene');
15073 //console.log('Box1Scene'+sid.runtime.properties());
15074 //X3dRof1_ViewPoint.value = X3dRof1_ViewPoint.position;
15075
15076 X3dRof1_ViewPosition_Value.value = X3dRof1_ViewPoint.getFieldValue('position');
15077 X3dRof1_ViewOrientation_Value.value = X3dRof1_ViewPoint.getFieldValue('orientation');
15078 X3dRof1_BoxPosition_Value = Box1Trans.getFieldValue('translation');
15079 X3dRof1_BoxRotation_Value.value = Box1Trans.getFieldValue('rotation');
15080
15081 box1 = document.getElementById('Box1');
15082 //X3dRof1_ViewOrientation_Value.value = box1.runtime.viewpoint();
15083
15084 //window.setInterval(showVP,500);
15085 window.addEventListener('load',showVP);
15086
15087 function newVP(){
15088 console.log('ViewPoint changed:'+event);
15089
15090 vp = document.getElementById('X3dRof1_ViewPoint');
15091 vp.addEventListener('viewpointchanged',newVP,false);
15092 X3dom.runtime.ready = function(){
15093 console.log('-- X3DOM ready');
15094 }
15095 //X3dom.runtime.debug(true);
15096
15097 var svg_deg = 0;
15098 function rotateX3DOM(){
15099 svg_deg += 2;
15100 }
15101 function X3DOM_Setup1(){
15102 X3dRof1_ViewPoint.setAttribute('position','0.8 0.7 5');
15103 X3dRof1_ViewPoint.setAttribute('position','0.8 0.7 4');
15104 Box1Material.setAttribute('diffusecolor','0 1 0');
15105 Box1Trans.setAttribute('translation','1.0 0.5 0.5');
15106 Box1Trans.setAttribute('rotation','1 1 1');
15107 }
15108 function X3DOM_Setup(){
15109 if( X3DOM_Section.open == true ){
15110 X3DOM_Setup1();
15111 }
15112 }
15113 X3DOM_Summary.addEventListener('click',X3DOM_Setup);
15114 X3DOM_Setup();
15115 </script>
15116
15117 <h3>Canvas Smiley onto X3DOM</h3>
15118 <canvas id="Smilly_1" width="150" height="150"></canvas>
15119 <script>
15120 function draw_smiley() {
15121 var canvas = Smilly_1;
15122 if( canvas.getContext() ){
15123 var ctx = canvas.getContext('2d');
15124 ctx.beginPath();
15125 ctx.arc(75, 75, 50, 0, Math.PI * 2, true); // Outer circle
15126 ctx.fillStyle = 'rgba(255,240,0,0.5)';
15127 ctx.fill();
15128
15129 ctx.beginPath();
15130 ctx.moveTo(110, 75);
15131 ctx.arc(75, 75, 35, 0, Math.PI, false); // Mouth (clockwise)
15132 ctx.moveTo(65, 65);
15133 ctx.stroke();
15134
15135 ctx.beginPath();
15136 ctx.arc(60, 65, 5, 0, Math.PI * 2, true); // Left eye
15137 ctx.moveTo(95, 65);
15138 ctx.arc(90, 65, 5, 0, Math.PI * 2, true); // Right eye
15139 ctx.stroke();
15140 ctx.fillStyle = 'rgba(0,0,0,0.8)';
15141 ctx.fill();
15142 }
15143 }
15144 function getSmillyUrl(){
15145 draw_smiley();
15146 url = Smilly_1.toDataURL("image/png");
15147 return url;
15148 }
15149 function putTextureTag(id){
15150 url = getSmillyUrl();
15151 document.write('<'+'ImageTexture'
15152 + ' id="'+id
15153 + ' url="'+url+'">'+'ImageTexture'+'');
15154 }
15155 </script>
15156
15157 <x3d width="150px" height="150px">
15158 <scene><shape>
15159 <appearance><ImageTexture url="WD-WallPaper03.png" /></appearance>
15160 <sphere DEF="obj" radius="3.3" />
15161 </shape></scene>
15162 </x3d>
15163
15164 <x3d width="150px" height="150px">
15165 <scene><shape>
15166 <appearance>
15167 <!--
15168 <ImageTexture url="http://im3/gshell/skype-rofl-texture-02.png" />
15169 -->
15170 <script>putTextureTag("Box1Texture");</script>
15171 </appearance>
15172 <sphere DEF="obj" radius="3.3" />
15173 </shape></scene>
15174 </x3d>
15175
15176 <x3d width="150px" height="150px">
15177 <scene>
15178 <shape>
15179 <appearance><ImageTexture url="WD-WallPaper03.png" /></appearance>
15180 <sphere DEF="obj" radius="3.3" />
15181 </shape>
15182 <transform translation="1 1 0">
15183 <shape>
15184 <appearance>
15185 <script>putTextureTag("Box1Texture");</script>
15186 <!--
15187 <ImageTexture url="http://im3/gshell/skype-rofl-texture-02.png" />
15188 -->
15189 </appearance>
15190 <sphere DEF="obj" radius="3.3" />
15191 </shape>
15192 </translation>
15193 </scene>
15194 </x3d>
15195 <style>
15196 x3d {
15197 display:inline;
15198 }
15199 </style>
15200
15201 <h3>Rofl by Unicode 1F923</h3>
15202 <style>
15203 .largefont {font-size:32px !important; text-align:center !important; }
15204 .iconfont {font-size:64px !important; text-align:center !important; }
15205 .hugefont {font-size:160px !important; text-align:center !important; }
15206 </style>
15207 &#x1f923; = &#x1f923;
15208 <big>&#x1f923;</big>
15209 <big><big>&#x1f923;</big></big>
15210 <span class="largefont">&#x1f923;</span>
15211 <span class="iconfont">&#x1f923;</span>
15212 <span class="hugefont">&#x1f923;</span>
15213 <br>
15214
15215 <input id="X3DOM_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
15216 <input id="X3DOM_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
15217 <input id="X3DOM_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
15218 <span id="X3DOM_WorkCodeView"></span>
15219 <script id="X3DOM_WorkCodeView">
15220 function X3DOM_openWorkCodeView(){
15221 function X3DOM_showWorkCode(){

```

```

15222     showHtmlCode(X3DOM_WorkCodeView,X3DOM_WorkCodeSpan);
15223   }
15224   X3DOM_WorkCodeViewOpen.addEventListener('click',X3DOM_showWorkCode);
15225 }
15226 X3DOM_openWorkCodeView(); // should be invoked by an event
15227 </script>
15228 </details>
15229 <!-- 3DROFL_WorkCodeSpan } -->
15230 *//</span>
15231 <!-- ===== Work } ===== -->
15232 </!-- ===== Work } ===== -->
15233 </!-- ===== Work { ===== -->
15234 <span id="Template_WorkCodeSpan">
15235 /
15236 <details><summary>Work Template</summary>
15237 <!-- ----- Template of Work// 2020-0928 SatoxITS { -->
15238 <h2>Template of Work</h2>
15239
15240 <style>
15241 </style>
15242 <script>
15243 </script>
15244
15245 <input id="Template_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
15246 <input id="Template_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
15247 <input id="Template_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
15248 <span id="Template_WorkCodeView"></span>
15249 <script id="Template_WorkScript">
15250 function Template_openWorkCodeView(){
15251   function Template_showWorkCode(){
15252     showHtmlCode(Template_WorkCodeView,Template_WorkCodeSpan);
15253   }
15254   Template_WorkCodeViewOpen.addEventListener('click',Template_showWorkCode);
15255 }
15256 Template_openWorkCodeView(); // should be invoked by an event
15257 </script>
15258 </details>
15259 <!-- Template_WorkCodeSpan } -->
15260 *//</span>
15261 <!-- ===== Work } ===== -->
15262 </!-- ===== Work { ===== -->
15263 <span id="OriginalSource_WorkCodeSpan">
15264 /
15265 <details open=""><summary>Original Source</summary>
15266 <!-- ----- OriginalSource // 2020-1009 SatoxITS { -->
15267 <h2>Original Source of GShell</h2>
15268 <input id="OriginalSource_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
15269 <input id="OriginalSource_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
15270 <input id="OriginalSource_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
15271 <span id="OriginalSource_TextElement"></span>
15272 <span id="OriginalSource_WorkCodeView"></span>
15273 <script id="OriginalSource_WorkScript">
15274 function OriginalSource_openWorkCodeView(){
15275   function OriginalSource_showWorkCode(){
15276     //showHtmlCode(OriginalSource_WorkCodeView,OriginalSource_WorkCodeSpan);
15277     //OriginalSource_TextElement = OriginalSourceNode;
15278     //console.log('src='\n'+OriginalSourceNode.outerHTML);
15279     showHtmlCodeX(OriginalSource_WorkCodeView,OriginalSourceNode,
15280       '\n',
15281       '\n',true);
15282   }
15283   OriginalSource_WorkCodeViewOpen.addEventListener('click',OriginalSource_showWorkCode);
15284 }
15285 //OriginalSourceNode = document.documentElement.cloneNode();
15286 //OriginalSourceNode = gsh.cloneNode(true); //=====
15287 //console.log('src='\n'+document.documentElement.outerHTML);
15288 //console.log('src='\n'+gsh.outerHTML);
15289 //console.log('src='\n'+OriginalSourceNode.innerHTML);
15290 OriginalSource_openWorkCodeView(); // should be invoked by an event
15291 //showNodeAsHtmlSource(OriginalSource_WorkCodeView,OriginalSourceNode);
15292 function SaveOriginalNode(){
15293   if( false ){
15294     m0 = performance.memory;
15295     mu0 = m0.usedJSHeapSize;
15296     console.log('-- heap bef clone: '
15297       +m0.usedJSHeapSize+' '+m0.totalHeapSize+' '+m0.jsHeapSizeLimit);
15298   }
15299   OriginalSourceNode = gsh.cloneNode(true);
15300   if( false ){
15301     m1 = performance.memory;
15302     mu1 = m1.usedJSHeapSize;
15303     mu = mu1 - mu0;
15304     //alert('-- clone: used heap '+mu0+' -> '+mu1+' = '+mu+' bytes';
15305     console.log('-- heap aft clone: '
15306       +m1.usedJSHeapSize+' '+m1.totalHeapSize+' '+m1.jsHeapSizeLimit);
15307     //OriginalSourceNode = document.documentElement.cloneNode(true);
15308   }
15309 }
15310 function Gsh_setupPage(){
15311   GshSetImages();
15312   //Indexer_afterLoaded();
15313   //GShell_initKeyCommands();
15314   //GKConsole_initConsole();
15315   GJConsole_initFactory();
15316   GJLink_init();
15317   InterFrameComm_init();
15318   Gshell_initTopbar();
15319   //WirtualDesktop_init();
15320   Banner_init();
15321   // Rff_Setup();
15322   Shading_Setup();
15323   window.setInterval(ShowResourceUsage,1000);
15324   //document.addEventListener('keydown',jgshCommand); // should be applied later?
15325   Pointillism_Setup();
15326   FontList_Setup();
15327   showFooter();
15328   GshInsideIcomSetup();
15329   SightClass_Setup();
15330   //spawnPackmonGo();
15331   //PackmonGo_Setup(null);
15332   DrawingCanvas_Setup();
15333   InstaColor_Setup();
15334 }
15335 function OnLoad(){
15336   SaveOriginalNode();
15337   Gsh_setupPage();
15338 }
15339 document.addEventListener('load',Gsh_setupPage);
15340 </script>
15341 </details>
15342 <!-- OriginalSource_WorkCodeSpan } -->
15343 *//</span>
15344 <!-- ===== Work } ===== -->
15345 </!-- ===== Work { ===== -->
15346 </div>
15347 <br><script>OnLoad();</script></span>
15348 </div>

```