make.tol de Antonio.Salmeron

Este programa se denomina Antonio.Salmeron pues es el constructor del sitio web AntonioSalmeron.com. La construcción del sitio web se realiza a partir de contenidos en formato de posts albergados en unos ficheros que se denominan agendas. Cada agenda contiene un conjunto de post, agrupados generalmente por temas. Estos ficheros de agendas se almacenan en un directorio que puede considerarse un banco de contenidos. Los post pueden pertenecer a múltiples clases, unas clases definidas en los posts y otras calculadas, como por ejemplo, las clase de los periodos de publicación de cada post. Este programa crea páginas Html de posts y de conjuntos de post por cada clase, a estas clases también se las denomina categorías.

El programa esta basado en un macro-expansor a doble nivel de Tol en Html, donde las semillas de Html contienen Tol embebido y los post para publicar también pueden contener Tol embebido, por lo que: a) dentro de la primera expansion, la de las semillas, b) se pueden realizar otras expansiones, que son las del lenguaje Tol contenido en los post. Hay 2 tipos de semillas para este sitio web: a) una para las páginas web del sitio AntonioSalmeron.com y b) otra para la elaboración de los curriculum viatae que es la misma para el curriculum en inglés y el curriculum en español.

Este programa emplea un directorio de agenda de posts, dentro de este directorio los posts se estructuran en varios ficheros que permiten organizarlos por su tipo de contenido y dentro de cada fichero por su fecha de publicación, los más recientes al principio. Con parte de los contenidos de los post este programa crea 4 documentos, 2 en español y 2 en ingles, en Html y en formato Pdf que son un curriculun vitae. A diferencia de otros sitios web construidos por Tol, en el sitio web AntonioSalmeron.com todos los post tienen el mismo nivel de importancia.

Con este programa: a) las páginas de artículos se generan con la opción art, b) las páginas con los artículos agrupados por categorías con la opción cat, c) las páginas Html por defecto del sitio (index.html) y de cada directorio, la página absoluta y la de control de errores con la opción rot, d) el mapa del sitio web en Xml con la opción xsm, que son las siglas de Xml site map y e) las opciones ftp, fup y snd permiten poner el contenido en online. Los comentarios del código de este programa están realizados utilizando unas veces el español, sin acentos, y otras veces el inglés.

Este programa sólo escribe los ficheros de páginas Html que son diferentes a los ya creados en ejecuciones anteriores de forma que no haya que enviar todo el conjunto de páginas sino las de modificadas de fecha más reciente que el último log de envío. Este control lo realiza la opcion fup, de ftp update, frente a la opción ftp que generar ficheros de mandatos de envío con todo el contenido del web. Nótese que las opciones fup y ftp solo generan ficheros de mandatos ftp de envío y luego se pueden ejecutar con la opcion snd.

Árbol de ficheros

Antonio.Salmeron construye las páginas y documentos del sitio web antoniosalmeron.con

← make.tol proceso principal de generación de contenidos del sitio web

- tol directorios de código fuente en lenguaje de programación Tol
  - cmm funciones comunes de textos, fechas, conjuntos, ficheros, etc.
    - ← txt.tol funciones de manejo de textos
    - ← dte.tol funciones de manejo de fechas en español
    - ← set.tol funciones de manejo de conjuntos
    - ← fil.tol funciones de gestión de ficheros
    - ← dir.tol funciones de gestión de directorios
    - ← tme.tol del macro-expansor simple de Tol en Html
    - ← htm.tol funciones básicas del lenguaje Html
    - ← ftp.tol funciones para generar mandatos para hacer Ftp
    - ← pdf.tol funciones para generar documentos en Pdf
    - ← xsm.tol para construir sitemaps en Xml
  - app funciones especificas de aplicacion para Antonio.Salmeron
    - ← pdb.tol funciones de manejo de los posts de una agenda
    - ← pht.tol funciones auxiliares para el Html de los post
  - ← inc.tol inclusión de los ficheros Tol básicos y de aplicación
- agenda directorio destinado a albergar los ficheros de agendas de posts
  - ← 03.arte.age ejemplo de 2 posts de contenido de poesía para publicar
- web directorio destinado a las paginas web generadas automáticamente
  - css directorio para ficheros de estilo
    - ← common.css fichero de estilo para las paginas Html del sitio web
  - seed directorio para ficheros semilla de Html con Tol embebido
    - ← seed.htm semilla de pagina Html con Tol embebido para el sitio web
    - ← seed.cv.htm semilla Html con Tol embebido para generar curriculum vitae
  - articulos directorio para paginas de artículos generadas por este programa
    - → sumasporparejas.html ejemplo de pagina Html generada automáticamente
  - categorias directorio para paginas de categorias generadas automáticamente
    - → logisticaytransporte.html ejemplo de pagina Html de categoria generada
  - → sitemap.xml mapa del sitio web generado en Xml de forma automática
- doc directorio de documentación del programa Antonio.Salmeron
  - → antoniosalmeron.grande.20111202.png ejemplo de página generada automáticamente por este programa
- → antonio_salmeron.pdf documento resumen de funciones del programa de generación Html

Declaraciones

Inclusiones

- `Set allInc`
  Inclusion de las funciones comunes y de aplicacion.

Constantes

- `Text makSep`
  Linea horizontal para separar fases de operacion.
- `Text CtrAge`
  All text files inside this directory.
- `Text CtrFil`
  Estadísticas de posts por clases.
- `Real CtrRec`
  Number of posts per recent page.
- `Real CtrTit`
  Number of posts per title.
- `Real CtrDes`
  Number of posts per description.
- `Set CatEsp`
  Categorias especiales.
- `Set CatGen`
  Categorias generales.
- `Set CatCur`
  Categorias curriculares.
- `Set CatTem`
  Categorias temporales.
- `Set CatMis`
  Categorias miscelaneas.
- `Set CatAre`
  Categorias de areas de aplicacion: funciones.
- `Set CatSec`
  Categorias de sectores de actividad.
- `Set CatTop`
  Categorias principales, top.
- `Set CatFor`
  Categorias dentro de la formacion.
- `Set CatExp`
  Categorias de la experiencia.
- `Set CatEmp`

Categorias de experiencia pero con empresa concreta.

- `Set CatCon`
  Categorias de contribución.
- `Set CatCom`
  Categorias de información complementaria.
- `Set CatAll`
  Todas las categorias.

Proceso

- `Text ctrExe`
  Argumento validado para la ejecucion del make.
- `Real makHlp`
  Es cierto si se ha visualizado la ayuda.

Lee la agenda

- `Set CtrPdb`
  Read all posts.

Genera el contenido

- `Real makCat`
  build all category pages.
- `Real makArt`
  Build all articles pages.
- `Real makRot`
  Build root absolute page.
- `Real makCvs`
  Build Cv html and pdf pages.
- `Real makXsm`
  Build Xml site map.
- `Real makFtp`
  Crear los ficheros de mandatos completos ftp o de actualizacion fup.
- `Real makSnd`
  Send files via ftp.

Set allInc

```
////////////////////////////////////////////////////////////////////////////
Set  allInc = Include("tol/inc.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Inclusion de las funciones comunes y de aplicacion.", allInc);
////////////////////////////////////////////////////////////////////////////
```

Estructuras de datos

```
Struct PdbSt   // Posts database
```

```
{
  Set  pstCla, // Conjunto de tipos de post
  Text pstNam, // Identificador del post, si no se usa se pone Pst+fecha
  Text pstTit, // Titulo del post
  Date pstDte, // Fecha de ocurrencia
  Date pstUpd, // Fecha de actualizacion
  Text pstTxt, // Texto del post es html
  Text pstLnk  // Enlaces del post es html
};
```

Constantes

Text makSep

```
////////////////////////////////////////////////////////////////////////////
Text makSep = "\n"+Repeat("_", 72)+"\n";
////////////////////////////////////////////////////////////////////////////
PutDescription("Linea horizontal para separar fases de operacion.",makSep);
////////////////////////////////////////////////////////////////////////////
```

Text CtrAge

```
////////////////////////////////////////////////////////////////////////////
Text CtrAge = "agenda";
////////////////////////////////////////////////////////////////////////////
PutDescription("All text files inside this directory.",CtrAge);
////////////////////////////////////////////////////////////////////////////
```

Text CtrFil

```
////////////////////////////////////////////////////////////////////////////
Text CtrFil = "poststat.csv";
////////////////////////////////////////////////////////////////////////////
PutDescription("Estadisticas de posts por clases.",CtrFil);
////////////////////////////////////////////////////////////////////////////
```

Real CtrRec

```
////////////////////////////////////////////////////////////////////////////
Real CtrRec = 15;
////////////////////////////////////////////////////////////////////////////
PutDescription("Number of posts per recent page.",CtrRec);
////////////////////////////////////////////////////////////////////////////
```

Real CtrTit

```
////////////////////////////////////////////////////////////////////////////
Real CtrTit =  4;
////////////////////////////////////////////////////////////////////////////
PutDescription("Number of posts per title.",CtrTit);
////////////////////////////////////////////////////////////////////////////
```

Real CtrDes

```
////////////////////////////////////////////////////////////////////////////
Real CtrDes =  8;
////////////////////////////////////////////////////////////////////////////
PutDescription("Number of posts per description.",CtrDes);
////////////////////////////////////////////////////////////////////////////
```

Set CatEsp

```
////////////////////////////////////////////////////////////////////////////
Set CatEsp = [["Reciente",
               "Completo"]];
////////////////////////////////////////////////////////////////////////////
PutDescription("Categorias especiales.",CatEsp);
////////////////////////////////////////////////////////////////////////////
```

Set CatGen

```
////////////////////////////////////////////////////////////////////////////
Set CatGen = [["Pintura y poesía",
               "Negocios",
               "Tecnología"]; //
////////////////////////////////////////////////////////////////////////////
PutDescription("Categorias generales.",CatGen);
////////////////////////////////////////////////////////////////////////////
```

Set CatCur

```
////////////////////////////////////////////////////////////////////////////
Set CatCur = [["Formación";
               "Experiencia";
               "Contribución";
               "Información complementaria"]];
////////////////////////////////////////////////////////////////////////////
PutDescription("Categorias curriculares.",CatCur);
////////////////////////////////////////////////////////////////////////////
```

Set CatTem

```
////////////////////////////////////////////////////////////////////////////
Set CatTem = [["2015-2019",
               "2010-2014",
               "2005-2009",
               "2000-2004",
               "1995-1999",
               "1990-1994",
               "1983-1989"]];
////////////////////////////////////////////////////////////////////////////
PutDescription("Categorias temporales.",CatTem);
////////////////////////////////////////////////////////////////////////////
```

Set CatMis

```
////////////////////////////////////////////////////////////////////////////
Set CatMis = [["Código";
               "Educación básica",
               "Web, post y vídeo"]];
////////////////////////////////////////////////////////////////////////////
PutDescription("Categorias miscelaneas.",CatMis);
```

```
////////////////////////////////////////////////////////////////////////////
```

Set CatAre

```
////////////////////////////////////////////////////////////////////////////
Set CatAre = [["Dirección y estrategia",
               "Economía y financiera",
               "Marketing y comercial",
               "Operaciones y logística",
               "Personal y formación"]];
////////////////////////////////////////////////////////////////////////////
PutDescription("Categorias de areas de aplicacion: funciones.",CatAre);
////////////////////////////////////////////////////////////////////////////
```

Set CatSec

```
////////////////////////////////////////////////////////////////////////////
Set CatSec = [["Banca y seguros",
               "Industria y energía",
               "Inmobiliario y edificación",
               "Legal y función pública",
               "Logística y transporte",
               "Prensa, editorial y medios"]];
////////////////////////////////////////////////////////////////////////////
PutDescription("Categorias de sectores de actividad.",CatSec);
////////////////////////////////////////////////////////////////////////////
```

Set CatTop

```
////////////////////////////////////////////////////////////////////////////
Set CatTop = CatEsp <<
             CatGen <<
             CatCur <<
             CatTem <<
             CatMis <<
             CatAre <<
             CatSec;
////////////////////////////////////////////////////////////////////////////
PutDescription("Categorias principales, top.",CatTop);
////////////////////////////////////////////////////////////////////////////
```

Set CatFor

```
////////////////////////////////////////////////////////////////////////////
Set CatFor = [["Titulación",
               "Curso recibido"]];
////////////////////////////////////////////////////////////////////////////
PutDescription("Categorias dentro de la formacion.",CatFor);
////////////////////////////////////////////////////////////////////////////
```

Set CatExp

```
////////////////////////////////////////////////////////////////////////////
Set CatExp = [["Profesional",
               "Empresarial",
               "Directiva",
               "Inicial"]];
////////////////////////////////////////////////////////////////////////////
PutDescription("Categorias de la experiencia.",CatExp);
```

```
////////////////////////////////////////////////////////////////////////
```

Set CatEmp

```
////////////////////////////////////////////////////////////////////////
Set CatEmp = [["ASolver",
               "Bayes",
               "EQ"]];
////////////////////////////////////////////////////////////////////////
PutDescription("Categorias de experiencia pero con empresa concreta.",CatEmp);
////////////////////////////////////////////////////////////////////////
```

Set CatCon

```
////////////////////////////////////////////////////////////////////////
Set CatCon = [["Publicación",
               "Curso impartido",
               "Ponencia",
               "Registro",
               "Referencia"]];
////////////////////////////////////////////////////////////////////////
PutDescription("Categorias de contribución.",CatCon);
////////////////////////////////////////////////////////////////////////
```

Set CatCom

```
////////////////////////////////////////////////////////////////////////
Set CatCom = [["Dato de interés",
               "Interés personal"]];
////////////////////////////////////////////////////////////////////////
PutDescription("Categorias de información complementaria.",CatCom);
////////////////////////////////////////////////////////////////////////
```

Set CatAll

```
////////////////////////////////////////////////////////////////////////
Set CatAll = CatTop <<
             CatFor <<
             CatExp <<
             CatEmp <<
             CatCon <<
             CatCom;
////////////////////////////////////////////////////////////////////////
PutDescription("Todas las categorias.",CatAll);
////////////////////////////////////////////////////////////////////////
```

Text ctrExe

```
////////////////////////////////////////////////////////////////////////
Text ctrExe = If(Not(ObjectExist("Text","ctrBat")), "hlp",
              If(ctrBat=="",                        "hlp",
                                                    ToLower(ctrBat)));
////////////////////////////////////////////////////////////////////////
PutDescription("Argumento validado para la ejecucion del make.", ctrExe);
////////////////////////////////////////////////////////////////////////
```

Real makHlp

```
////////////////////////////////////////////////////////////////////////
Real makHlp = If(ctrExe!="hlp", FALSE,
{
  Text WriteLn(
  makSep+"help:
  Usage: make [OPTION]
  Builds antoniosalmeron.com site
  OPTION
    cat: build all category pages
    art: build all articles pages
    rot: build root absolute page
    cvs: build cv pages: english, spanish and theirs Pdf files
    xsm: build xml site maps
    ftp: build ftp files (go to ftp dir and run manually)
    fup: build a file update protocol (newer than ftp.log file)
    snd: send file via ftp
    ___
    web: build local web: cat, art, rot, cvs, xms
    all: do all works: cat, art, rot, cvs, xms, fup y snd
    hlp: show this help");
  TRUE
});
////////////////////////////////////////////////////////////////////////
PutDescription("Es cierto si se ha visualizado la ayuda.", makHlp);
////////////////////////////////////////////////////////////////////////
```

Set CtrPdb

```
////////////////////////////////////////////////////////////////////////
Set CtrPdb = If(ctrExe <: [["all", "web", "cat", "art", "cvs"]],
                PdbRead(CtrAge),
                Empty);
////////////////////////////////////////////////////////////////////////
PutDescription("Read all posts.", CtrPdb);
////////////////////////////////////////////////////////////////////////
```

Real makCat

```
////////////////////////////////////////////////////////////////////////
Real makCat = If(Or(ctrExe=="all", ctrExe=="web", ctrExe=="cat"),
{
  Text WriteLn(makSep+"building all category pages...");
  Text WriteFile(CtrFil, "Posts; Class\n");

  Real CtrArt = FALSE; // Categories, not articles

  // Make category pages
  Set  cicCat = EvalSet(CatAll, Real(Text CtrPag)
  {
    Text filNam = PhtFileName(CtrPag); // Output file
    Text WriteLn("["+CtrPag+"|"+filNam+"]");

    Set  SelPdb = // Selected post from database for this page
    {
      Set selSet =
      If(CtrPag=="Completo",
         CtrPdb,
      If(CtrPag=="Reciente",
         PdbFirstN(CtrPdb, CtrRec, Real(Set pdbObj) { TRUE }),
         Select(CtrPdb, Real(Set pdbObj) { CtrPag <: pdbObj->pstCla })));

      Real selCrd = Card(selSet);
      Text txtCrd = FormatReal(selCrd,"%.0lf");
      Text WriteLn("  "+txtCrd+" posts selected");
      Text AppendFile(CtrFil, txtCrd+"; "+CtrPag+"\n");
```

```
      // If none then the most recent
      If(selCrd, selSet, PdbFirstN(CtrPdb, CtrRec, Real(Set pdbObj) { TRUE }))
    };

    TmeFile("web/seed/seed.htm", "web/"+filNam)
  });

  Card(cicCat)
},FALSE);
////////////////////////////////////////////////////////////////////////////
PutDescription("build all category pages.",makCat);
////////////////////////////////////////////////////////////////////////////
```

Real makArt

```
////////////////////////////////////////////////////////////////////////////
Real makArt = If(Or(ctrExe=="all", ctrExe=="web", ctrExe=="art"),
{
  Text WriteLn(makSep+"building all article pages...");

  Real CtrArt = TRUE; // Articles

  // Make article pages
  Set  cicArt = For(1, Card(CtrPdb), Real(Real pstNum)
  {
    Set  SelPdb = SetSubCicle(CtrPdb, pstNum, CtrRec);
    Text CtrPag = TxtOutside2Tag(CtrPdb[pstNum]->pstTit,"<",">");

    Text filNam = PhtFileName(CtrPag); // Output file
    Text WriteLn("["+CtrPag+"|"+filNam+"]");
    TmeFile("web/seed/seed.htm", "web/"+filNam)
  });

  Card(cicArt)
},FALSE);
////////////////////////////////////////////////////////////////////////////
PutDescription("Build all articles pages.", makArt);
////////////////////////////////////////////////////////////////////////////
```

Real makRot

```
////////////////////////////////////////////////////////////////////////////
Real makRot = If(Or(ctrExe=="all", ctrExe=="web", ctrExe=="rot"),
{
  Text WriteLn(makSep+"building root absolute page...");

  Text WriteLn("-> homepage index.html");
  Text inxHtm = ReadFile("web/categorias/index.html");
  Text inxRot = Replace(inxHtm,"\"../", "\"");
  Text WriteFile("web/index.html", inxRot);

  Text WriteLn("-> absolute index");
  Text absSav = Replace(inxHtm, " href=\"http",
                                " _XX_=\"_XX_"); // Save external references
  Text absRep = ReplaceTable(absSav,
  [[
     [[ " src=\"../",          " src=\"http://www.antoniosalmeron.com/"]],
     [[" href=\"../",         " href=\"http://www.antoniosalmeron.com/"]]
  ]],1);
  Text absRec = Replace(absRep, " _XX_=\"_XX_",
                                " href=\"http"); // Recall external references

  Real FilWriteIfDiff("web/absoluto.html", absRec);

  Text WriteLn("-> common directory page, utf-8");
  FilCopy("../common/dir.html", "web/categorias/comxidir.html", TRUE)
```

```
},FALSE);
////////////////////////////////////////////////////////////////////////////
PutDescription("Build root absolute page.", makRot);
////////////////////////////////////////////////////////////////////////////
```

Real makCvs

```
////////////////////////////////////////////////////////////////////////////
Real makCvs = If(Or(ctrExe=="all", ctrExe=="web", ctrExe=="cvs"),
{
  Text e2(Text eng, Text spa) { If(CtrEng, eng, spa) };

  Text tx(Real int) { FormatReal(int, "%.0lf") };

  Text inSet(Text claTxt, Real newOld) // Si True new->old, si false old->new
  {
    Text iniTag = "<li class=\"" + If(CtrEng, "eng", "spa") + "\">";
    Text endTag = "</li>";
    Set  selSet = Select(CtrPdb, Real(Set objPst) { claTxt<:objPst->pstCla });
    Set  ordSet = If(newOld, selSet, SetReverse(selSet));
    Set  txtSet = EvalSet(ordSet, Text(Set objPst)
    {
      Text txtCit = TxtBetween2Tag(objPst->pstTxt, iniTag, endTag, TRUE);
      If(txtCit == "", "", "  <li>\n    "+txtCit+"\n  </li>\n")
    });
    "\n"+SetSum(txtSet)+"\n"
  };

  Text liSet(Text claTxt) { inSet(claTxt, TRUE)  };
  Text liRev(Text claTxt) { inSet(claTxt, FALSE) };

  Text WriteLn(makSep+"building Cv html and pdf pages...");
  Text sedPth = "web/seed/seed.cv.htm";

  Real engBld =
  {
    Real CtrEng = TRUE;
    Text htmPth = "web/curriculum/antoniosalmeroncveng.html";
    Text pdfPth = Replace(htmPth, ".html", ".pdf");

    Real htmBld = TmeFile(sedPth, htmPth);
    Real pdfBld = PdfBuild(htmPth, pdfPth);
    htmBld + pdfBld
  };

  Real spaBld =
  {
    Real CtrEng = FALSE;
    Text htmPth = "web/curriculum/antoniosalmeroncvspa.html";
    Text pdfPth = Replace(htmPth, ".html", ".pdf");

    Real htmBld = TmeFile(sedPth, htmPth);
    Real pdfBld = PdfBuild(htmPth, pdfPth);
    htmBld + pdfBld
  };

  engBld + spaBld
},FALSE);
////////////////////////////////////////////////////////////////////////////
PutDescription("Build Cv html and pdf pages.",makCvs);
////////////////////////////////////////////////////////////////////////////
```

Real makXsm

```
////////////////////////////////////////////////////////////////////////////
Real makXsm = If(Or(ctrExe=="all", ctrExe=="web", ctrExe=="xsm"),
{
  Text WriteLn(makSep+"building xml site map...");
```

```
  XsmDir("web/sitemap.xml", "web", "http://www.antoniosalmeron.com/")
},FALSE);
////////////////////////////////////////////////////////////////////////////
PutDescription("Build Xml site map.", makXsm);
////////////////////////////////////////////////////////////////////////////
```

Real makFtp

```
////////////////////////////////////////////////////////////////////////////
Real makFtp = If(ctrExe <: [["ftp", "fup", "all"]],
{
  Text msgTxt = If(ctrExe=="ftp", "ftp (all files)", "fup (update)");
  Text absPth = Replace(GetSourcePath(ctrExe),"/make.tol","");     // Absoluto
  Text locPth = absPth+"/web";                    // Ha de ser una ruta absoluta
  Text webNam = GetFileName(absPth);             // Nombre del directorio actual
  Text dtePth = If(ctrExe=="ftp", "", "ftp/"+webNam+".log");       // Relativo

  Text WriteLn(makSep+webNam+": building "+msgTxt+" from "+locPth+"...");
  FtpAll(
    webNam,                       // Web name
    "www.antoniosalmeron.com", // Host remoto
    locPth,                       // Directorio local
    dtePth,                       // Fichero señal de fecha de actualizacion
    [[//dir        extension    type    binary or ascii
      ["",         "html",      "html", FALSE]],  // ASCII
      ["",         "ico",       "ico",  TRUE]],   // Binary favicon.ico
      ["",         "xml",       "xml",  FALSE]],  // ASCII  site map, rss
      ["",         "css",       "css",  FALSE]],  // ASCII
      ["src",      "js",        "js",   FALSE]],  // ASCII  javascript
      ["",         "gif",       "gif",  TRUE]],   // Binary
      ["",         "jpg",       "jpg",  TRUE]],   // Binary
      ["",         "png",       "png",  TRUE]],   // Binary
      ["",         "xls",       "xls",  TRUE]],   // Binary
      ["",         "pdf",       "pdf",  TRUE]]    // Binary
    ]])
},FALSE);
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Crear los ficheros de mandatos completos ftp o de actualizacion fup.",
makFtp);
////////////////////////////////////////////////////////////////////////////
```

Real makSnd

```
////////////////////////////////////////////////////////////////////////////
Real makSnd = If(Or(ctrExe=="all", ctrExe=="snd"),
{
  Text WriteLn(makSep+"sending files using ftp...");
  System(W("ftp/Antonio.Salmeron.bat"))
},FALSE);
////////////////////////////////////////////////////////////////////////////
PutDescription("Send files via ftp.",makSnd);
////////////////////////////////////////////////////////////////////////////
```

txt.tol de Antonio.Salmeron

Text functions.

Declaraciones

Funciones de nombre corto

- `Text Q(Text txtVal)`
  Retorna un texto entre dobles comillas. Equivalente a la funcion Tol Qt().
- `Text W(Text txtVal)`
  Retorna un camino en formato Unix convertido a formato Windows/DOS.
- `Text R(Text txtLst)`
  Retorna un texto elegido al azar de entre los tokens (|) de otro texto.
- `Text F(Anything anyVal)`
  Retorna numeros, fechas, textos, conjuntos como un texto de formato simple.

Funciones

- `Set Txt2Set(Text txtInp, Text sepTok)`
  Retorna un conjunto a partir de un texto txtInp, troceandolo por un separador sepTok y dependiendo de ctrFun puede, en este orde, si C compactar, si N eliminar los texto nulos, si U retornar elementos unicos y si S retornar el set ordenado.
- `Set TxtTokenizer(Text txtInp, Text tagBrk)`
  Retorna un conjunto de textos resultado de cortar el texto de entrada por un unico tag tagBrk no incluyendo el tag tagBrk dentro de los textos. Se soporta en la funcion Tol Tokenizer() que rompe por un unico caracter. Usa como caracter interno de corte el 7 (bell), esperando que no aparezca.
- `Set TxtForChr(Text inpTxt, Code funChr)`
  Retorna el conjunto resultado de aplicar la funcion funChr(Text oneChr) a todos los caracteres del texto de entrada txtInp. Retorna un Set a imagen de las funciones basicas For() y EvalSet().
- `Text TxtBetween2Tag(Text inpTxt, Text tagIni, Text tagEnd, Real cmpFlg)`
  Retorna un subtexto entre la primera ocurrencia de tagIni y tagEnd. Si tagIni o tagEnd no aparecen retorna la tira vacia. Si cmpFlg es cierto entonces aplica la funcion Compact() al texto que retorna. Por ejemplo: TxtBetween2Tag('a b [[c]] d [[e]] f', '[[', ']]', TRUE) retorna 'c'.
- `Text TxtOutside2Tag(Text txtInp, Text tagIni, Text tagEnd)`
  Returns all the texts outside 2 tags (tagIni and tagEnd) in txt. For example: <aaa(::)bbb(:::)ccc>, <(>, <)> -> <aaabbbccc>.
- `Text TxtOutHtmTag(Text htmTxt)`
  Retorna todos el texto fuera de los tags de html.
- `Text TxtOutHtmScr(Text htmTxt)`
  Retorna todos el texto fuera de los tags de html y de los scripts. Sirve para extraer texto limpio del que sacar palabras clave.

- `Text TxtReplaceSecuence(Text txtInp, Set repTab)`

  Retorna un texto resultado de aplicar la tabla de reemplazamientos repTab al texto de entrada txtIno. Es una version de ReplaceTable(txtInp, repTab, 1) que garantiza la secuencia de los reemplazamientos del primero al ultimo de los reemplazamientos. Es una funcion recursiva. Cada reemplazamiento solo se efectua una vez.

Funciones de nombre corto

Text Q()

```
////////////////////////////////////////////////////////////////////////////
Text Q(Text txtVal) // Text
////////////////////////////////////////////////////////////////////////////
{ Char(34) + txtVal + Char(34) };
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un texto entre dobles comillas. Equivalente a la funcion Tol Qt().",
Q);
////////////////////////////////////////////////////////////////////////////
```

Text W()

```
////////////////////////////////////////////////////////////////////////////
Text W(Text txtVal) // Text
////////////////////////////////////////////////////////////////////////////
{ Replace(txtVal, "/", "\\") };
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un camino en formato Unix convertido a formato Windows/DOS.",
W);
////////////////////////////////////////////////////////////////////////////
```

Text R()

```
////////////////////////////////////////////////////////////////////////////
Text R(Text txtLst) // List of texts separated with |
////////////////////////////////////////////////////////////////////////////
{ SetGetRand(Txt2Set(txtLst, "|")) };
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un texto elegido al azar de entre los tokens (|) de otro texto.",
R);
////////////////////////////////////////////////////////////////////////////
```

Text F()

```
////////////////////////////////////////////////////////////////////////////
Text F(Anything anyVal)
////////////////////////////////////////////////////////////////////////////
{
  Text graVal = Grammar(anyVal);
  Case
  (
    graVal=="Text", anyVal, // Ya es texto

    graVal=="Real", If(EQ(anyVal, Round(anyVal)),
                       FormatReal(anyVal, "%.0lf"),  // Entero sin decimales
                       FormatReal(anyVal, "%.2lf")), // 2 decimales
```

```
    graVal=="Date", If(Hour(anyVal),
                       FormatDate(anyVal, "%c%Y/%m/%d %h:%i:%s"), // Tiempo
                       FormatDate(anyVal, "%c%Y/%m/%d")),         // Fecha

    graVal=="Set",
    {
      Real crdSet = Card(anyVal);
      Case(
      EQ(crdSet,0), "[]",
      EQ(crdSet,1), "["+F(anyVal[1])+"]",
      TRUE,         "["+F(anyVal[1])+SetSum(For(2,crdSet,Text(Real setPos)
                  { "|"+F(anyVal[setPos]) }))+"]")
    },

    TRUE,           "Not basic type"
  )
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna numeros, fechas, textos, conjuntos como un texto de formato simple.",
F);
////////////////////////////////////////////////////////////////////////////
```

Set Txt2Set()

```
////////////////////////////////////////////////////////////////////////////
Set Txt2Set(Text txtInp, // Texto de entrada
            Text sepTok) // Elemento separador
////////////////////////////////////////////////////////////////////////////
{
  Set setSep = TxtTokenizer(txtInp, sepTok);
  Set setCmp = EvalSet(setSep, Text(Text eleTxt) { Compact(eleTxt) });
  setCmp
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un conjunto a partir de un texto txtInp, troceandolo por un separador
sepTok y dependiendo de ctrFun puede, en este orde, si C compactar, si N
eliminar los texto nulos, si U retornar elementos unicos y si S retornar el
set ordenado.",
Txt2Set);
////////////////////////////////////////////////////////////////////////////
```

Set TxtTokenizer()

```
////////////////////////////////////////////////////////////////////////////
Set TxtTokenizer(Text txtInp, // Texto de entrada
                 Text tagBrk) // Tag por el que se corta
////////////////////////////////////////////////////////////////////////////
{ Tokenizer(Replace(txtInp, tagBrk, Char(7)), Char(7)) };
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un conjunto de textos resultado de cortar el texto de entrada por un
unico tag tagBrk no incluyendo el tag tagBrk dentro de los textos.
Se soporta en la funcion Tol Tokenizer() que rompe por un unico caracter.
Usa como caracter interno de corte el 7 (bell), esperando que no aparezca.",
TxtTokenizer);
////////////////////////////////////////////////////////////////////////////
```

Set TxtForChr()

```
////////////////////////////////////////////////////////////////////////////
Set TxtForChr(Text inpTxt, // Texto de entrada
              Code funChr) // Funcion tipo Anything(Text oneChr)
```

```
////////////////////////////////////////////////////////////////////////////
{
  Real lenTxt = TextLength(inpTxt);
  For(1, lenTxt, Anything(Real posTxt) { funChr(Sub(inpTxt,posTxt,posTxt)) })
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna el conjunto resultado de aplicar la funcion funChr(Text oneChr)
a todos los caracteres del texto de entrada txtInp.
Retorna un Set a imagen de las funciones basicas For() y EvalSet().",
TxtForChr);
////////////////////////////////////////////////////////////////////////////
```

Text TxtBetween2Tag()

```
////////////////////////////////////////////////////////////////////////////
Text TxtBetween2Tag(Text inpTxt, // Texto de entrada
                    Text tagIni, // Tag inicial
                    Text tagEnd, // Tag final
                    Real cmpFlg) // Si true aplica Compact()
////////////////////////////////////////////////////////////////////////////
{
  Real posIni = TextFind(inpTxt, tagIni);
  Text result = If(LE(posIni,0), "",
  {
    Real lenIni = TextLength(tagIni);
    Real posSub = posIni + lenIni;
    Real posEnd = TextFind(inpTxt, tagEnd, posSub);
    If(LE(posEnd, 0), "", Sub(inpTxt, posSub, posEnd-1))
  });
  If(cmpFlg, Compact(result), result)
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un subtexto entre la primera ocurrencia de tagIni y tagEnd.
Si tagIni o tagEnd no aparecen retorna la tira vacia.
Si cmpFlg es cierto entonces aplica la funcion Compact() al texto que retorna.
Por ejemplo:
  TxtBetween2Tag('a b [[ c ]] d [[ e ]] f', '[[', ']]', TRUE)
  retorna 'c'.",
TxtBetween2Tag);
////////////////////////////////////////////////////////////////////////////
```

Text TxtOutside2Tag()

```
////////////////////////////////////////////////////////////////////////////
Text TxtOutside2Tag(Text txtInp, // Input text
                    Text tagIni, // Initial tag
                    Text tagEnd) // End tag)
////////////////////////////////////////////////////////////////////////////
{
  Set  txtSet = TxtTokenizer(tagIni + tagEnd + txtInp, tagIni);
  Set  txtCic = EvalSet(txtSet, Text(Text txtTok)
  { TxtBetween2Tag(txtTok + tagIni, tagEnd, tagIni, FALSE) });
  SetSum(txtCic)
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns all the texts outside 2 tags (tagIni and tagEnd) in txt.
For example: <aaa(::)bbb(:::)ccc>, <(>, <)> -> <aaabbbccc>.",
TxtOutside2Tag);
////////////////////////////////////////////////////////////////////////////
```

Text TxtOutHtmTag()

```
////////////////////////////////////////////////////////////////////////
Text TxtOutHtmTag(Text htmTxt)
////////////////////////////////////////////////////////////////////////
{ TxtOutside2Tag(htmTxt, "<", ">") };
////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna todos el texto fuera de los tags de html.",
TxtOutHtmTag);
////////////////////////////////////////////////////////////////////////
```

Text TxtOutHtmScr()

```
////////////////////////////////////////////////////////////////////////
Text TxtOutHtmScr(Text htmTxt)
////////////////////////////////////////////////////////////////////////
{ TxtOutHtmTag(TxtOutside2Tag(htmTxt, "<script", "</script>")) };
////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna todos el texto fuera de los tags de html y de los scripts.
Sirve para extraer texto limpio del que sacar palabras clave.",
TxtOutHtmScr);
////////////////////////////////////////////////////////////////////////
```

Text TxtReplaceSecuence()

```
////////////////////////////////////////////////////////////////////////
Text TxtReplaceSecuence(Text txtInp, // Texto de entrada
                        Set  repTab) // Tabla de reemplazamientos
////////////////////////////////////////////////////////////////////////
{
  Real crdTab = Card(repTab); // Numero de cambios a realizar
  If(!crdTab, txtInp, // Nada que hacer
  {
    Text txtNew = Replace(txtInp, repTab[1][1], repTab[1][2]);  // Un cambio
    TxtReplaceSecuence(txtNew, SetLastN(repTab,crdTab-1)) // Resto de cambios
  })
};
////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un texto resultado de aplicar la tabla de reemplazamientos repTab al
texto de entrada txtIno.
Es una version de ReplaceTable(txtInp, repTab, 1) que garantiza la secuencia
de los reemplazamientos del primero al ultimo de los reemplazamientos.
Es una funcion recursiva.
Cada reemplazamiento solo se efectua una vez.",
TxtReplaceSecuence);
////////////////////////////////////////////////////////////////////////
```

dte.tol de Antonio.Salmeron

Date functions.

Declaraciones

set.tol de Antonio.Salmeron

Set functions.

Declaraciones

Funciones

- `Text Set2Txt(Set valSet, Text iniTxt, Text endTxt, Text sepTxt, Text sepLst, Text txtDet, Text datFmt, Text dteDet, Text dteFmt)`
  Returns a text like a list with all the elements of valSet converted in a text format (elements types: Text, Real or Date). Arguments: - iniTxt initial text, for example: '(', '[[', '', etc. - endTxt end text, for example ')', ']]', '', etc. - sepTxt elements separator, for example '; ', ',', '', etc. - sepLst two last elements separator, for example, ' & ', ' and ', etc., you can specify the same as sepTxt - txtDet text delimiters, for exmple, quotes for TOL, single quote for SQL, nothing, etc. - datFmt real numbers format, for explame, '%.0lf' for integers, if none then uses the default TOL real number format. - dteDet date delimiters, for example, simgle quote for SQL. - dteFmt date format, for example, '%c%Y%m%d', if none then uses the default TOL dates format. Only works with TOL types Text, Real or Date, when find a Set type then works in a recursive way with the same arguments.
- `Set Set2Keyword(Set valSet, Real minChr, Real a2zOrd, Real maxKey)`
  Returns a set with all the elements of valSet converted in a text format, ordered and without repetitions. Remove all word with LE(TextLength(), minChr). Returns the maxKey elements more occurrencies.
- `Text Set2TxtCommaAmp(Set valSet)`
  Return a text list with all the elements of valSet converted in a text format with commas and & in Html style. For example Set2TxtCommaAmp([['a','b','c','d']]) returns: a, b, c & d -> in html -> a, b, c & d.
- `Set SetFirstN(Set setInp, Real numEle)`
  Retorna un subconjunto de un conjunto con los primeros numEle elementos. Si el conjunto tiene menos de numEle elementos los retorna todos. Si numEle es 0 o negativo retorna el conjunto vacio.
- `Set SetFirstNByFun(Set set, Real num, Code funSor)`
  Returns a subset with the first num elements of a set ordered by funSor. If the set does not have num elements returns the set ordered by funSor.
- `Set SetLastN(Set setInp, Real numEle)`
  Retorna un subconjunto de un conjunto con los ultimos numEle elementos. Si el conjunto tiene menos de numEle elementos los retorna todos.

- `Anything SetGetRand(Set setInp)`
  Retorna un elemento al azar del conjunto de entrada setInp. Si setInp es Empty retorna FALSE.
- `Set SetReverse(Set setInp)`
  Retorna el conjunto de entrada con su orden invertido.
- `Set SetSubCicle(Set setInp, Real iniPos, Real lenRet)`
  Retorna un subconjunto de un conjunto con los primeros numEle elementos. Si el conjunto tiene menos de numEle elementos los extrae por el principio. Por ejemplo:
  SetSubCicle([a,b,c,d],3,8) -> [c,d,a,b,c,d,a,b]

Text Set2Txt()

```
////////////////////////////////////////////////////////////////////////////
Text Set2Txt(Set  valSet, // Set of elements
             Text iniTxt, // Initial text for list
             Text endTxt, // End text for list
             Text sepTxt, // Element separators
             Text sepLst, // 2 last elements separator
             Text txtDet, // Delimiter for texts
             Text datFmt, // Format for real numbers
             Text dteDet, // Delimiter for dates
             Text dteFmt) // Format for dates
////////////////////////////////////////////////////////////////////////////
{
  Real card = Card(valSet);
  Text body =
    If(EQ(card,0), "",
    If(EQ(card,1), F(valSet[1]),
    If(EQ(card,2), F(valSet[1])+sepLst+F(valSet[2]),
    {
      Set txtVal = For(2,card,Text(Real p)
      {
        If(EQ(p,card),sepLst,sepTxt) + F(valSet[p])
      });
      F(valSet[1]) + BinGroup("+",txtVal)
    })));
  iniTxt+body+endTxt
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns a text like a list with all the elements of valSet converted in a
text format (elements types: Text, Real or Date).
Arguments:
 - iniTxt initial text, for example: '(', '[[', '', etc.
 - endTxt end text, for example ')', ']]', '', etc.
 - sepTxt elements separator, for example '; ', ', ', '', etc.
 - sepLst two last elements separator, for example, ' & ', ' and ', etc.,
   you can specify the same as sepTxt
 - txtDet text delimiters, for exmple, quotes for TOL, single quote for
   SQL, nothing, etc.
 - datFmt real numbers format, for explame, '%.0lf' for integers, if none
   then uses the default TOL real number format.
 - dteDet date delimiters, for example, simgle quote for SQL.
 - dteFmt date format, for example, '%c%Y%m%d', if none
   then uses the default TOL dates format.
Only works with TOL types Text, Real or Date, when find a Set type then
works in a recursive way with the same arguments.",
Set2Txt);
////////////////////////////////////////////////////////////////////////////
```

Set Set2Keyword()

```
////////////////////////////////////////////////////////////////////////////
Set Set2Keyword(Set  valSet, // Set of texts
                Real minChr, // Minimum number of chars
                Real a2zOrd, // If true ordered az else by ocurrences
                Real maxKey) // Maximun number of keywords
////////////////////////////////////////////////////////////////////////////
{
  Set cmmWrd =
  [[
     "ambos",
     "aunque",
     "avppm",
     "barcelona",
     "carlos",
     "cinco",
     "cuando",
     "cuatro",
     "con-q",
     "dentro",
     "desde",
     "donde",
     "entre",
     "están",
     "estas", "estos",
     "había",
     "junto",
     "luego",
     "madrid",
     "misma", "mismo",
     "mucho",
     "número",
     "otro", "otros",
     "página", "páginas",
     "partir", "permite",
     "primer", "primera",
     "propio",
     "puede", "pueden",
     "segunda", "segundo", "segundos",
     "siguiente", "siguientes",
     "sobre",
     "también",
     "tiene", "tienen",
     "través",
     "utilizando", "utilizar",
     "verse",
     "zulú"
  ]];

  // Convert all to text
  Text lstTxt = ToLower(Set2Txt(valSet,"", "", " ", " ", "", "", "", ""));

  Text htmNot = TxtOutHtmScr(lstTxt);

  Text lstCls = Compact(ReplaceTable(htmNot, // _ must NOT be changed
    [[ [[Char(34), " "]], [["'",      " "]],
       [[".",      " "]], [[":",      " "]],
       [[";",      " "]], [[",",      " "]],
       [["[",      " "]], [["]",      " "]],
       [["(",      " "]], [[")",      " "]],
/*     [["<",      " "]], [[">",      " "]],    Sobra */
       [["«",      " "]], [["»",      " "]], // Titulos de libros
       [["¡",      " "]], [["!",      " "]],
       [["¿",      " "]], [["?",      " "]],
       [["/",      " "]], [["|",      " "]],
       [["&amp",   " "]],
       [[" ", " "]],
       [[" none ", " "]],
       [[" the ",  " "]],
       [[" and ",  " "]],
       [[" or ",   " "]],
       [[" & ",    " "]],
       [[" y ",    " "]],
       [[" o ",    " "]]
```

```
    ]]));

// Split in words
  Set  setTxt = Tokenizer(lstCls, " ");

// Select word with more than minChr characters that locks like words
  Set  setSel = Select(setTxt, Real(Text wrdTxt)
  {
    Real wrdLen = TextLength(wrdTxt);
    Text iniLet = Sub(wrdTxt, 1,      1);
    Text lstLet = Sub(wrdTxt, wrdLen, wrdLen);
    And(GT(wrdLen, minChr), iniLet >= "a", iniLet <= "z",
                            lstLet >= "a", lstLet <= "z")
  });

// Classify words
  Set  setCla = Classify(setSel, Real(Text a, Text b)
                         { Compare(ToLower(a),ToLower(b)) });

// Make a frecuencies table [ word, number of occurrencies ]
  Set  tabFrq = EvalSet(setCla, Set(Set cla)
                         { [[ cla[1], Card(cla) ]] });

// Sort (with most ocurrencies first)
  Set  srtFrq = Sort(tabFrq, Real(Set a, Set b)
                      { Compare(Real(b[2]), Real(a[2])) });

// Project by word column (the first column)
  Set  keySet = EvalSet(srtFrq, Text(Set a) { Text(a[1]) });

// Minus common words
  Set  keyMin = keySet - cmmWrd;

// Select the first maxKey or Card(keySet) or all if maxKey=0
  Set  keyFst = If(maxKey, SetFirstN(keyMin, maxKey), keyMin);

// Sort if is needed
  If(! a2zOrd, keyFst,
     Sort(keyFst, Real(Text a, Text b)
       { Compare(ToLower(a),ToLower(b)) })) // Order
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns a set with all the elements of valSet converted in a
text format, ordered and without repetitions.
Remove all word with LE(TextLength(), minChr).
Returns the maxKey elements more occurrencies.",
Set2Keyword);
////////////////////////////////////////////////////////////////////////////
```

Text Set2TxtCommaAmp()

```
////////////////////////////////////////////////////////////////////////////
Text Set2TxtCommaAmp(Set  valSet) // Set of elements
////////////////////////////////////////////////////////////////////////////
{ Set2Txt(valSet, "", "", ", ", " &amp;amp; ", "", "", "", "") };
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Return a text list with all the elements of valSet converted in a
text format with commas and & in Html style.
For example Set2TxtCommaAmp([['a','b','c','d']]) returns:
 a, b, c &amp;amp; d -> in html -> a, b, c & d.",
Set2TxtCommaAmp);
////////////////////////////////////////////////////////////////////////////
```

Set SetFirstN()

```
////////////////////////////////////////////////////////////////////////////
```

```
Set SetFirstN(Set  setInp, // Set de entrada
              Real numEle) // Numero de elementos a retornar
////////////////////////////////////////////////////////////////////////////
{
  If(LE(numEle, 0), Empty,
     For(1, Min(Card(setInp), numEle), Anything(Real setPos)
         { setInp[setPos] }))
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un subconjunto de un conjunto con los primeros numEle elementos.
Si el conjunto tiene menos de numEle elementos los retorna todos.
Si numEle es 0 o negativo retorna el conjunto vacio.",
SetFirstN);
////////////////////////////////////////////////////////////////////////////
```

Set SetFirstNByFun()

```
////////////////////////////////////////////////////////////////////////////
Set SetFirstNByFun(Set set, Real num, Code funSor)
////////////////////////////////////////////////////////////////////////////
{ SetFirstN(Sort(set, funSor), num) };
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns a subset with the first num elements of a set ordered by funSor.
If the set does not have num elements returns the set ordered by funSor.",
SetFirstNByFun);
////////////////////////////////////////////////////////////////////////////
```

Set SetLastN()

```
////////////////////////////////////////////////////////////////////////////
Set SetLastN(Set  setInp, // Set de entrada
             Real numEle) // Numero de elementos a retornar
////////////////////////////////////////////////////////////////////////////
{
  If(LE(numEle, 0), Empty,
     For(Max(1, 1+Card(setInp)-numEle), Card(setInp), Anything(Real setPos)
         { setInp[setPos] }))
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un subconjunto de un conjunto con los ultimos numEle elementos.
Si el conjunto tiene menos de numEle elementos los retorna todos.",
SetLastN);
////////////////////////////////////////////////////////////////////////////
```

Anything SetGetRand()

```
////////////////////////////////////////////////////////////////////////////
Anything SetGetRand(Set setInp)
////////////////////////////////////////////////////////////////////////////
{
  Real setCrd = Card(setInp);
  If(LE(setCrd, 0), FALSE,
  {
    Real     rndPos = Min(setCrd, Max(1, Round(Rand(0, setCrd)+0.5)));
    Anything rndVal = setInp[rndPos];
    rndVal
  })
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un elemento al azar del conjunto de entrada setInp.
Si setInp es Empty retorna FALSE.",
```

```
SetGetRand);
////////////////////////////////////////////////////////////////////////////
```

Set SetReverse()

```
////////////////////////////////////////////////////////////////////////////
Set SetReverse(Set  setInp) // Set de entrada
////////////////////////////////////////////////////////////////////////////
{
  Real numEle = Card(setInp);
  If(LE(numEle, 0), Empty,
     For(1, numEle, Anything(Real setPos) { setInp[numEle+1-setPos] }))
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna el conjunto de entrada con su orden invertido.",
SetReverse);
////////////////////////////////////////////////////////////////////////////
```

Set SetSubCicle()

```
////////////////////////////////////////////////////////////////////////////
Set SetSubCicle(Set  setInp, // Conjunto de elementos
                Real iniPos, // Posicion inicial
                Real lenRet) // Numero de elementos a retornar
////////////////////////////////////////////////////////////////////////////
{
  Real modCic(Real setPos, Real crdSet)
  {
    Real modPos = setPos % crdSet;
    If(LE(modPos, 0), crdSet, modPos)
  };

  Real crdSet = Card(setInp);

  For(0, lenRet-1, Anything(Real setPos)
  {
    Real posCic = modCic(iniPos + setPos, crdSet);
    setInp[posCic]
  })
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un subconjunto de un conjunto con los primeros numEle elementos.
Si el conjunto tiene menos de numEle elementos los extrae por el principio.
Por ejemplo: SetSubCicle([a,b,c,d],3,8) -> [c,d,a,b,c,d,a,b]",
SetSubCicle);
////////////////////////////////////////////////////////////////////////////
```

fil.tol de Antonio.Salmeron

Funciones de ficheros.

Declaraciones

Funciones

- `Real FilWriteIfDiff(Text filPth, Text txtNew)`
  Escribe txtOut en filPth si filPth es un nuevo fichero o si su contenido actual es diferente de txtOut y en otro caso no escribe nada. Retorna true si lo ha escrito y false si no ha sido necesario hacerlo. Evita que los ficheros adquiran una fecha de actualizacion reciente cuando su contenido no ha cambiando realmente, evitando, por ejemplo, retransmisiones por ftp de fichero que realmente no han cambiado.
- `Real FilCheckExtension(Text filPth, Text txtExt, Real casSen)`
  Retorna cierto si el fichero de ruta filPth tiene la extension txtExt. Si casSen es cierto distingue entre mayusculas de minusculas.
- `Real FilCopy(Text oldPth, Text newPth, Real overwrite)`
  Returns TRUE in the file oldNam can be copied over newNam.

Real FilWriteIfDiff()

```
////////////////////////////////////////////////////////////////////////////
Real FilWriteIfDiff(Text filPth, // Fichero de salida
                    Text txtNew) // Texto para escribir
////////////////////////////////////////////////////////////////////////////
{
  If(! FileExist(filPth),        { Text WriteFile(filPth, txtNew); TRUE },
  If(txtNew != ReadFile(filPth), { Text WriteFile(filPth, txtNew); TRUE },
                                    FALSE ))
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Escribe txtOut en filPth si filPth es un nuevo fichero o si su contenido
actual es diferente de txtOut y en otro caso no escribe nada.
Retorna true si lo ha escrito y false si no ha sido necesario hacerlo.
Evita que los ficheros adquiran una fecha de actualizacion reciente
cuando su contenido no ha cambiando realmente, evitando, por ejemplo,
retransmisiones por ftp de fichero que realmente no han cambiado.",
FilWriteIfDiff);
////////////////////////////////////////////////////////////////////////////
```

Real FilCheckExtension()

```
////////////////////////////////////////////////////////////////////////////
Real FilCheckExtension(Text filPth, // Ruta del fichero
                       Text txtExt, // Extension para comprobar
                       Real casSen) // Si distingue mayus/minus en extension
////////////////////////////////////////////////////////////////////////////
{
  Text filExt = If(casSen, GetFileExtension(filPth),
                    ToLower(GetFileExtension(filPth)));
  Text chkExt = If(casSen, txtExt,
                    ToLower(txtExt));
  filExt==chkExt
};
////////////////////////////////////////////////////////////////////////////
```

```
PutDescription(
"Retorna cierto si el fichero de ruta filPth tiene la extension txtExt.
Si casSen es cierto distingue entre mayusculas de minusculas.",
FilCheckExtension);
////////////////////////////////////////////////////////////////////////////
```

Real FilCopy()

```
////////////////////////////////////////////////////////////////////////////
Real FilCopy(Text oldPth,    // Ruta del fichero origen
             Text newPth,    // Ruta del fichero destino
             Real overwrite) // Si cierto sobre-escribe
////////////////////////////////////////////////////////////////////////////
{
  Text cpyTxt = "copy "+oldPth+" "+newPth;
  Text cpyCmd = Replace(cpyTxt, "/", "\\");
  Case(
    Not(FileExist(oldPth)),     FALSE,           // Original doesn't exist
    oldPth==newPth,             FALSE,           // Auto-copy ?
    Not(FileExist(newPth)),     System(cpyCmd),  // Execute copy
    overwrite,                  System(cpyCmd),  // Copy and overwrite
                                FALSE)           // Don't overwrite
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns TRUE in the file oldNam can be copied over newNam.",
FilCopy);
////////////////////////////////////////////////////////////////////////////
```

dir.tol de Antonio.Salmeron

Directories functions.

Declaraciones

Funciones

- `Set DirExtAll(Text dirPth, Text chkExt, Real toLowe, Real casSen)`
  Returns a set of relative paths of files with the extension chkExt that are inside the directory dirPth and inside all of its directories. If toLowe then all names are changed to lower case. If casSen then are case sensitive in the extension match. Is a version of DirAll() function that only check extensions and not uses TextMatch() that writes warnings at Tol 2.0.1.
- `Text DirReadFiles(Text dirPth, Text chkExt, Text filSep)`
  Returns as a text the contents of all files in dirPth that theirs file names match with file pattern filPat. In this text, the contenst of each file will be separated with filSep.

Set DirExtAll()

```
////////////////////////////////////////////////////////////////////////////
Set DirExtAll(Text dirPth, // Directory path
              Text chkExt, // File extension
              Real toLowe, // If true all paths are changed to lower case
              Real casSen) // If true the extension match is case sensitive
////////////////////////////////////////////////////////////////////////////
{
  If(Not(DirExist(dirPth)), Empty,
  {
    Set  getDir = GetDir(dirPth);
    Set  filSet = getDir[1];
    Set  dirSet = getDir[2];

    Set  filFnd = EvalSet(filSet, Text(Text filNam)
    {
      If(!FilCheckExtension(filNam, chkExt, casSen), "",
         If(toLowe, ToLower(dirPth+"/"+filNam), dirPth+"/"+filNam))
    });

    Set filSel = Select(filFnd, Real(Text filNam) { filNam != "" });

    Set  dirFnd = EvalSet(dirSet, Set(Text subDir)
      { DirExtAll(dirPth+"/"+subDir, chkExt, toLowe, casSen) });

    Real dirCar = Card(dirFnd);

    If(EQ(dirCar,0), filSel,
    If(EQ(dirCar,1), filSel << dirFnd[1],
                     filSel << BinGroup("+", dirFnd)))
  })
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns a set of relative paths of files with the extension chkExt that
are inside the directory dirPth and inside all of its directories.
If toLowe then all names are changed to lower case.
If casSen then are case sensitive in the extension match.
Is a version of DirAll() function that only check extensions and not uses
TextMatch() that writes warnings at Tol 2.0.1.",
DirExtAll);
////////////////////////////////////////////////////////////////////////////
```

Text DirReadFiles()

```
////////////////////////////////////////////////////////////////////////////
Text DirReadFiles(Text dirPth, // Directory path
                  Text chkExt, // File extension
                  Text filSep) // File separator
////////////////////////////////////////////////////////////////////////////
{
  Set pthSet = DirExtAll(dirPth, chkExt, FALSE, TRUE);
  Set txtSet = EvalSet(pthSet, Text(Text filPth) { ReadFile(filPth) });
  Set2Txt(txtSet, "", "", filSep, filSep, "", "", "", "")
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns as a text the contents of all files in dirPth that theirs file names
match with file pattern filPat.
In this text, the contenst of each file will be separated with filSep.",
DirReadFiles);
////////////////////////////////////////////////////////////////////////////
```

tme.tol de Antonio.Salmeron

Macro-expansor potenciado de Tol para ser inyectado en Html o en otros lenguajes de programacion.

Por defecto utiliza tags que combinan < y { el inicial y el final } y > como Php utiliza <? y ?> y Asp <% y %>.

Esta es la version potenciada que permite inyectar Tol dentro de semillas Html del tipo seed.htm, este codigo Tol embebido se expande en el codigo Html de las agendas de post u otras cosas, que a su vez pueden contener codigo Tol embebido que se puede volver a expandir. Existe una version de este codigo que no admite esta capacidad.

Esta es una version especialmente depurada de este codigo e incluye la modificacion TxtReplaceSecuence(codTxt, repTab) en vez de la clasica ReplaceTable(codTxt, repTab, 1).

Declaraciones

Constantes

- `Text TmeIni`
  Default initial tag. Other tags can be used as <% like Asp or <? like Php.
- `Text TmeEnd`
  Default ending tag. Other tags can be used as %> like Asp or ?> like Php.
- `Text TmeSep`
  For internal use. Lets use Tokenizer() replacing initial tags by TmeSep.
- `Text TmeEmpty`
  All code must return some text. If there are only definitions and replace with nothing can be ended with an empty string of with a TmeEmpty.

Funciones

- `Set TmeGetMacros(Text tagIni, Text tagEnd, Text codTxt)`
  Returns a set with all codes inside the initial en the ending tags.
- `Text TmeFormat(Anything retVal)`
  Intenta retornar un texto a partir de otros tipo, es una funcion equivalente a la funcion de nombre corto de textos F().
- `Set TmeEvalMacros(Set codSet)`
  Returns a set with the text results of all macros. This secuential evaluation does not let to share functions and variables between macros.
- `Set TmeShareMacros(Set codSet, Real codPos)`
  Returns a set with the text results of all macros. This recursive evaluation lets to share definitions, functions and variables between macros.
- `Text TmeSubMacros(Text tagIni, Text tagEnd, Text codTxt, Real share)`

Returns the result of full macro expansion. The original remmark with the old code ReplaceTable(codTxt, repTab, 1) was: Be aware if you uses duplicated macros and assumes something about your order definitions. Now the function TxtReplaceSecuence(codTxt, repTab) make the replacements in a strick order. Inside this code always is called with share=TRUE.

- `Text TmePst(Text codTxt)`
  Easy way to call TmeSubMacros() with the default values and behaviour. Returns the text generated.
- `Text TmeExpandFile(Text tagIni, Text tagEnd, Text inpFil, Text outFil)`
  Returns the result of full macro expansion in inpFil. Update the file remmark from classic seed.htm to the output file name outFil. If outFil is not empty then writes the result in outFil.
- `Real TmeFile(Text inpFil, Text outFil)`
  Easy way to call TmeExpandFile() with the default values and behaviour. Returns true if some text was generated.

Constantes

Text TmeIni

```
////////////////////////////////////////////////////////////////////////////
Text TmeIni = "<"+"{";
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Default initial tag. Other tags can be used as <% like Asp or <? like Php.",
TmeIni);
////////////////////////////////////////////////////////////////////////////
```

Text TmeEnd

```
////////////////////////////////////////////////////////////////////////////
Text TmeEnd = "}"+">";
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Default ending tag. Other tags can be used as %> like Asp or ?> like Php.",
TmeIni);
////////////////////////////////////////////////////////////////////////////
```

Text TmeSep

```
////////////////////////////////////////////////////////////////////////////
Text TmeSep = Char(7);
////////////////////////////////////////////////////////////////////////////
PutDescription(
"For internal use. Lets use Tokenizer() replacing initial tags by TmeSep.",
TmeIni);
////////////////////////////////////////////////////////////////////////////
```

Text TmeEmpty

```
////////////////////////////////////////////////////////////////////////////
Text TmeEmpty = "";
```

```
////////////////////////////////////////////////////////////////////////////
PutDescription(
"All code must return some text. If there are only definitions and replace
with nothing can be ended with an empty string of with a TmeEmpty.",
TmeEmpty);
////////////////////////////////////////////////////////////////////////////
```

Set TmeGetMacros()

```
////////////////////////////////////////////////////////////////////////////
Set TmeGetMacros(Text tagIni, // Initial tag
                 Text tagEnd, // Ending tag
                 Text codTxt) // Other programing language code
////////////////////////////////////////////////////////////////////////////
{
  Text repTxt = Replace(codTxt, tagIni, TmeSep);
  Set  linSet = Tokenizer(repTxt, TmeSep);
  Real lenSet = Card(linSet);
  If(lenSet <= 1, Empty, // There are not macros
  {
    Set For(2, lenSet, Text(Real posLin)
    {
      Text linTxt = linSet[posLin];
      Real posEnd = TextFind(linTxt, tagEnd, 1);
      If(posEnd <= 1, "", // Without ending (0) or not code inside (1)
                      Sub(linTxt, 1,posEnd-1))
    })
  })
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns a set with all codes inside the initial en the ending tags.",
TmeGetMacros);
////////////////////////////////////////////////////////////////////////////
```

Text TmeFormat()

```
////////////////////////////////////////////////////////////////////////////
Text TmeFormat(Anything retVal)
////////////////////////////////////////////////////////////////////////////
{ F(retVal) };
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Intenta retornar un texto a partir de otros tipo, es una funcion equivalente
a la funcion de nombre corto de textos F().",
TmeFormat);
////////////////////////////////////////////////////////////////////////////
```

Set TmeEvalMacros()

```
////////////////////////////////////////////////////////////////////////////
Set TmeEvalMacros(Set codSet) // A set with Tol code that returs text
////////////////////////////////////////////////////////////////////////////
{
  EvalSet(codSet, Text(Text codEle)
  {                                  // All the evaluations at the same level,
    Anything retVal = Eval(codEle); // can't share the definitions
    TmeFormat(retVal)                // Try to convert to text
  })
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns a set with the text results of all macros. This secuential evaluation
does not let to share functions and variables between macros.",
TmeEvalMacros);
```

```
////////////////////////////////////////////////////////////////////////////
```

Set TmeShareMacros()

```
////////////////////////////////////////////////////////////////////////////
Set TmeShareMacros(Set  codSet, // Set of tol codes
                   Real codPos) // Position inside codSet (recursion counter)
////////////////////////////////////////////////////////////////////////////
{
  Real lstPos = Card(codSet);
  If(codPos > lstPos, Empty, // Nothing to do
  {                                  // Evaluations at different stack levels,
    Anything retVal = Eval(codSet[codPos]); // can inherit definitions
    Text txtFmt = TmeFormat(retVal);        // Try to convert to text
    [[ txtFmt ]] << TmeShareMacros(codSet, codPos+1) // Recursion
  })
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns a set with the text results of all macros. This recursive evaluation
lets to share definitions, functions and variables between macros.",
TmeShareMacros);
////////////////////////////////////////////////////////////////////////////
```

Text TmeSubMacros()

```
////////////////////////////////////////////////////////////////////////////
Text TmeSubMacros(Text tagIni, // Initial tag
                  Text tagEnd, // Ending tag
                  Text codTxt, // Other programing language code
                  Real share)  // If true share definitions between macros
////////////////////////////////////////////////////////////////////////////
{
  Set macSet = TmeGetMacros(tagIni, tagEnd, codTxt);
  Set txtSet = If(share, TmeShareMacros(macSet, 1), TmeEvalMacros(macSet));

  Set repTab = For(1, Card(macSet), Set(Real macPos)
               { [[ Text(tagIni+macSet[macPos]+tagEnd), txtSet[macPos] ]] });
  TxtReplaceSecuence(codTxt, repTab) // ReplaceTable(codTxt, repTab, 1)
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns the result of full macro expansion.
The original remmark with the old code ReplaceTable(codTxt, repTab, 1) was:
Be aware if you uses duplicated macros and assumes something about your order
definitions. Now the function TxtReplaceSecuence(codTxt, repTab) make the
replacements in a strick order.
Inside this code always is called with share=TRUE.",
TmeSubMacros);
////////////////////////////////////////////////////////////////////////////
```

Text TmePst()

```
////////////////////////////////////////////////////////////////////////////
Text TmePst(Text codTxt) // Other programing language code
////////////////////////////////////////////////////////////////////////////
{ TmeSubMacros(TmeIni, TmeEnd, codTxt, TRUE) };
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Easy way to call TmeSubMacros() with the default values and behaviour.
Returns the text generated.",
TmePst);
////////////////////////////////////////////////////////////////////////////
```

Text TmeExpandFile()

```
////////////////////////////////////////////////////////////////////////////
Text TmeExpandFile(Text tagIni, // Initial tag
                   Text tagEnd, // Ending tag
                   Text inpFil, // Input file
                   Text outFil) // Output file
////////////////////////////////////////////////////////////////////////////
{
  Text codTxt = ReadFile(inpFil);
  Text outTxt = TmeSubMacros(tagIni, tagEnd, codTxt, TRUE); // Sharing

  Text outRep = Replace(outTxt, "// FILE    : seed.htm",
                                "// FILE    : "+GetFileName(outFil));

  Real wriFil = If(outFil != "", FilWriteIfDiff(outFil, outRep), FALSE);
  outTxt
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns the result of full macro expansion in inpFil.
Update the file remmark from classic seed.htm to the output file name outFil.
If outFil is not empty then writes the result in outFil.",
TmeExpandFile);
////////////////////////////////////////////////////////////////////////////
```

Real TmeFile()

```
////////////////////////////////////////////////////////////////////////////
Real TmeFile(Text inpFil, // Input file
             Text outFil) // Output file
////////////////////////////////////////////////////////////////////////////
{ If(TmeExpandFile(TmeIni, TmeEnd, inpFil, outFil) != "", TRUE, FALSE) };
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Easy way to call TmeExpandFile() with the default values and behaviour.
Returns true if some text was generated.",
TmeFile);
////////////////////////////////////////////////////////////////////////////


////////////////////////////////////////////////////////////////////////////
Text TmePut(Text codTol) // Tol programing language code
////////////////////////////////////////////////////////////////////////////
{ TmeIni+" "+codTol+" "+TmeEnd };
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns a Tol code inside Tme brackets.",
TmePut);
////////////////////////////////////////////////////////////////////////////
```

htm.tol de Antonio.Salmeron

Funciones de basicas de Html (HyperText Markup Language).

Declaraciones

Funciones

- `Text Htm2Ide(Text prgNam, Text lblTxt, Real makLow)`
  Retorna una etiqueta de texto convertida en identificador Html, por ejemplo, para utilizarse en un <a name>. Un idetificador ha de empezar por letras A..Z o a..z y seguir por letras, numeros 0..9 o subrayador, menos, dos puntos o punto. Esta funcion solo conserva las letras y numeros en su posicion y cualquier otro caracter lo cambia por el subrayador. Dependiendo del parametro makLow lo pasa todo a minusculas. Si no empieza por letra le antepone una X.

Text Htm2Ide()

```
////////////////////////////////////////////////////////////////////////////
Text Htm2Ide(Text prgNam, // Nombre del programa
             Text lblTxt, // Etiqueta de entrada
             Real makLow) // A minusculas
////////////////////////////////////////////////////////////////////////////
{
  Text lblInp = ReplaceTable(prgNam+"_"+lblTxt, [[
    ["á","a"]],["é","e"]],["í","i"]],["ó","o"]],["ú","u"]],["ü","u"]],
    ["Á","A"]],["É","E"]],["Í","I"]],["Ó","O"]],["Ú","U"]],["Ü","U"]],
    ["ñ","n"]],["Ñ","N"]]
  ]], 1);

  Set  setChr = TxtForChr(lblInp, Text(Text oneChr)
  {
    Case(
      And(oneChr >= "0", oneChr <= "9"), oneChr,
      And(oneChr >= "A", oneChr <= "Z"), If(makLow, ToLower(oneChr), oneChr),
      And(oneChr >= "a", oneChr <= "z"), oneChr,
      TRUE,                              "_")
  });
  Text txtSum = If(Card(setChr), SetSum(setChr), "");

  If(txtSum=="",  "",           // No hay identificador
  If(txtSum<"A",  "X"+txtSum,   // No ha de empezar por numero
  If(txtSum=="_", "X"+txtSum,   // No ha de empezar por _
                  txtSum)))  // Es correcto
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna una etiqueta de texto convertida en identificador Html, por ejemplo,
para utilizarse en un <a name></a>.
Un idetificador ha de empezar por letras A..Z o a..z y seguir por letras,
numeros 0..9 o subrayador, menos, dos puntos o punto.
Esta funcion solo conserva las letras y numeros en su posicion y cualquier
otro caracter lo cambia por el subrayador.
Dependiendo del parametro makLow lo pasa todo a minusculas.
Si no empieza por letra le antepone una X.",
Htm2Ide);
////////////////////////////////////////////////////////////////////////////
```

ftp.tol de Antonio.Salmeron

Funciones para realizar Ftp (versión independiente del web). No realiza el Ftp sino que crea todos los ficheros de mandatos que permiten realizarlo de forma automatica.

Declaraciones

Funciones

- `Real FtpDir(Text hstDir, Text locDir, Text extPat, Text cmdFil, Real toLowe, Real casSen, Text dtePth)`
  Agrega de formar recursiva a un fichero de nombre cmdFil las instrucciones necesarias para enviar todos los ficheros con extension extPat de un directorio local (locDir) y de sus subdirectorios. Hay que proporcionarle el directorio remoto (hstDir) bajo el cual se requiere hacer la copia. Si toLowe es minuscula todos los nombres de ficheros y subdirectorios son enviados en minusculas, esto es clasico para sistemas Unix o Linux. Si casSen es TRUE el math de fichero se realizar sensible a mayusculas y minusculas. Si se le proporciona el nombre de un fichero que indica una fecha dtePth, entonces solo transmite los ficheros mas recientes que dicho fichero, este fichero señal de fecha puede ser el fichero de log del ftp.
- `Real FtpBuild(Text hstDir, Text locDir, Text extPat, Text cmdFil, Text hstNam, Real toLowe, Real casSen, Real binCtr, Text dtePth)`
  Crea un fichero de nombre cmdFil con las instrucciones necesarias para enviar todos los ficheros de extension extPat de un directorio local (locDir) y de sus subdirectorios. Hay que proporcionar el nombre del host y el nombre del directorio remoto (hstDir) bajo el cual se requiere hacer la copia: locDir hstDir / | \ -> / | \ a b c a b c Si toLowe es minuscula todos los nombres de ficheros y subdirectorios son enviados en minusculas, esto es clasico para sistemas Unix o Linux. Si casSen es TRUE el math de fichero se realizar sensible a mayusculas y minusculas. Sin binCtr es TRUE la transmision es en modo binario y en otro caso en Ascii. Si se le proporciona el nombre de un fichero que indica una fecha dtePth, entonces solo transmite los ficheros mas recientes que dicho fichero, este fichero señal de fecha puede ser el fichero de log del ftp. En Windows se ejecuta como ftp -n -s:cmdFil. Retorna el numero de ordenes de transmision dadas.
- `Real FtpAll(Text webNam, Text hstDom, Text locDir, Text dtePth, Set ftpSet)`
  Funcion principal que crea todos los mandatos para la realizacion del ftp. Realiza un ciclo creando ficheros de mandatos para todos los elementos de la tabla ftpSet.

Real FtpDir()

```
////////////////////////////////////////////////////////////////////////
Real FtpDir(Text hstDir, // Directorio remoto
            Text locDir, // Directorio local
            Text extPat, // Patron de extension de fichero
            Text cmdFil, // Fichero de salida con mandatos ftp
            Real toLowe, // Envio de nombre en minusculas
            Real casSen, // Equiparacion sensible a mayus/minusculas
            Text dtePth) // File path to obtain a update date
////////////////////////////////////////////////////////////////////////
```

```
{
  Set  getDir   = GetDir(locDir);
  Set  filSet   = getDir[1];
  Set  dirSet   = getDir[2];

  Set  filSnd  = EvalSet(filSet, Text(Text filNam)
  {
    If(!FilCheckExtension(filNam, extPat, casSen), "", // Does not match
    {
      Real filNew = If(dtePth=="", TRUE, // There is not a file to compare
                       FileTime(locDir+"/"+filNam) >= FileTime(dtePth));
      If(! filNew, "", // Is not a recent update file
      {
        Text filLow = If(toLowe, ToLower(filNam), filNam);
        "send "+filNam+" "+filLow+"\n"
      })
    })
  });

  Text filCmd = If(Card(filSnd)==0, "", BinGroup("+",filSnd));
  Text allCmd = If(filCmd=="", "",
  {
    Text WriteLn("ftp> "+locDir);

    Text AppendFile(cmdFil, "mkdir "+hstDir+"\n");
    Text AppendFile(cmdFil, "cd "+hstDir+"\n");
    Text AppendFile(cmdFil, "lcd "+locDir+"\n");
    Text AppendFile(cmdFil, filCmd)
  });

  Set  dirSnd  = EvalSet(dirSet, Real(Text subDir)
  {
    Text dirName = If(toLowe, ToLower(subDir), subDir);
    FtpDir(hstDir+"/"+dirName, locDir+"/"+dirName, extPat,
           cmdFil, toLowe,  casSen, dtePth)
  });

  Card(filSnd)+Card(dirSnd)
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Agrega de formar recursiva a un fichero de nombre cmdFil las instrucciones
necesarias para enviar todos los ficheros con extension extPat de un
directorio local (locDir) y de sus subdirectorios.
Hay que proporcionarle el directorio remoto (hstDir) bajo el cual se requiere
hacer la copia.
Si toLowe es minuscula todos los nombres de ficheros y subdirectorios son
enviados en minusculas, esto es clasico para sistemas Unix o Linux.
Si casSen es TRUE el math de fichero se realizar sensible a mayusculas y
minusculas.
Si se le proporciona el nombre de un fichero que indica una fecha dtePth,
entonces solo transmite los ficheros mas recientes que dicho fichero,
este fichero señal de fecha puede ser el fichero de log del ftp.",
FtpDir);
////////////////////////////////////////////////////////////////////////////
```

Real FtpBuild()

```
////////////////////////////////////////////////////////////////////////////
Real FtpBuild(Text hstDir, // Directorio remoto
              Text locDir, // Directorio local
              Text extPat, // Patron de extension de fichero
              Text cmdFil, // Fichero de salida con mandatos ftp
              Text hstNam, // Nombre del host remoto
              Real toLowe, // Envio de ficheros en minusculas
              Real casSen, // Equiparacion sensible a mayusculas/minusculas
              Real binCtr, // Si TRUE envio en binario, si FALSE en Ascii
              Text dtePth) // Fichero de señal de fecha de actualizacion
////////////////////////////////////////////////////////////////////////////
{
  Text WriteLn("ftp> call as: ftp -n -s:"+cmdFil);
  Text WriteLn("ftp> searching...");
```

```
  Text WriteFile (cmdFil, "open "+hstNam+"\n");
  Text AppendFile(cmdFil, "user "+FtpHUs+" "+FtpHPa+"\n");
  Text AppendFile(cmdFil, "cd "+hstDir+"\n");
  Text AppendFile(cmdFil, "lcd "+locDir+"\n");

  Text If(binCtr, AppendFile(cmdFil, "binary"+"\n"),
                  AppendFile(cmdFil, "ascii" +"\n"));

  Set  getDir   = GetDir(locDir);
  Set  filSet   = getDir[1];
  Set  dirSet   = getDir[2];

  Set  filSnd  = EvalSet(filSet, Text(Text filNam)
  {
    If(!FilCheckExtension(filNam, extPat, casSen), "", // Doesn't match
    {
      Real filNew = If(dtePth=="", TRUE, // There is not a file to compare
                       FileTime(locDir+"/"+filNam) >= FileTime(dtePth));
      If(! filNew, "", // Is not a recent update file
      {
        Text filLow = If(toLowe, ToLower(filNam), filNam);
        AppendFile(cmdFil,"send "+filNam+" "+filLow+"\n")
      })
    })
  });

  Set  dirSnd  = EvalSet(dirSet, Real(Text subDir)
  {
    Text dirNam = If(toLowe, ToLower(subDir), subDir);
    FtpDir(hstDir+"/"+dirNam, locDir+"/"+dirNam, extPat,
           cmdFil, toLowe,  casSen, dtePth)
  });

  Text AppendFile(cmdFil,"close"+"\n");
  Text AppendFile(cmdFil,"quit"+"\n");
  Text WriteLn("ftp> done");

  Card(filSnd)+Card(dirSnd)
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Crea un fichero de nombre cmdFil con las instrucciones necesarias para enviar
todos los ficheros de extension extPat de un directorio local (locDir)
y de sus subdirectorios.
Hay que proporcionar el nombre del host y el nombre del directorio remoto
(hstDir) bajo el cual se requiere hacer la copia:
         locDir          hstDir
       /   |   \  ->  /   |   \
      a    b    c    a    b    c
Si toLowe es minuscula todos los nombres de ficheros y subdirectorios son
enviados en minusculas, esto es clasico para sistemas Unix o Linux.
Si casSen es TRUE el math de fichero se realizar sensible a mayusculas y
minusculas.
Sin binCtr es TRUE la transmision es en modo binario y en otro caso en Ascii.
Si se le proporciona el nombre de un fichero que indica una fecha dtePth,
entonces solo transmite los ficheros mas recientes que dicho fichero, este
fichero señal de fecha puede ser el fichero de log del ftp.
En Windows se ejecuta como ftp -n -s:cmdFil.
Retorna el numero de ordenes de transmision dadas.",
FtpBuild);
////////////////////////////////////////////////////////////////////////////
```

Real FtpAll()

```
////////////////////////////////////////////////////////////////////////////
Real FtpAll(Text webNam, // Web name
            Text hstDom, // Dominio del host remoto
            Text locDir, // Directorio local
            Text dtePth, // Fichero de señal de fecha de actualizacion
            Set  ftpSet) // Tabla de transporte ftp
////////////////////////////////////////////////////////////////////////////
```

```
{
  Text cmdFtp = "ftp/"+webNam+".bat";
  Text WriteFile(cmdFtp, W("cd "+Replace(locDir,"/web","/ftp")+"\n")+
                         "copy "+webNam+".bat "+webNam+".log\n");

  Real ftp(Text dirNam, Text extPat, Text filTyp, Real binCtr)
  {
    Text cmdFil = webNam + filTyp + ".ftp";
    Real bldFtp = FtpBuild(
                    If(dirNam!="", FtpHDi+"/"+dirNam, FtpHDi),
                    If(dirNam!="", locDir+"/"+dirNam, locDir),
                    extPat,
                    "ftp/"+cmdFil,
                    hstDom,                  // Host domain
                    FALSE,                   // To lower all
                    TRUE,                    // Case sensitive
                    binCtr ,                 // binary or Ascii
                    dtePth);                 // File to obtain a update date

    Text AppendFile(cmdFtp, "ftp -n -s:"+cmdFil+" >> "+webNam+".log\n");
    bldFtp
  };

  Set ftpCic = EvalSet(ftpSet, Real(Set ftpReg)
  {
    ftp(ftpReg[1], ftpReg[2], ftpReg[3], ftpReg[4])
  });

  Card(ftpCic)
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Funcion principal que crea todos los mandatos para la realizacion del ftp.
Realiza un ciclo creando ficheros de mandatos para todos los elementos de
la tabla ftpSet.",
FtpAll);
////////////////////////////////////////////////////////////////////////////
```

pdf.tol de Antonio.Salmeron

Pdf functions.

Declaraciones

Funciones

- `Real PdfBuild(Text filInp, Text filOut)`
  Returns true if can create a pdf file filOut from an html file filInp.

Real PdfBuild()

```
////////////////////////////////////////////////////////////////////////////
Real PdfBuild(Text filInp, // Input html file
              Text filOut) // Output pdf file
////////////////////////////////////////////////////////////////////////////
{
  If(Not(FileExist(filInp)), FALSE, // Nothing to do
  {
    Text absPth = Ois.AutoPath("."); // Ruta absoluta al punto de ejecucion

    Text dosInp = W(absPth+"/"+filInp); // Rutas con \ de Dos
    Text dosOut = W(absPth+"/"+filOut);

    Text margin = " /margin-left   42 "+
                  " /margin-right  42 "+
                  " /margin-top    42 "+
                  " /margin-bottom 42 ";

    Text cmdTxt = PdfExe+" "+                    // Executable pdf convertor
                  dosInp+" "+                    // Input file
                  dosOut+" "+                    // Output file
                  "/jpeg 100 "+                  // Compresion
                  "/psize A4" +                  // Author
                  margin;

//  Text WriteLn(cmdTxt);
    Text WriteLn("  Pdf: "+filOut);

    System(cmdTxt) // Execute html to pdf conversion
  })
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns true if can create a pdf file filOut from an html file filInp.",
PdfBuild);
////////////////////////////////////////////////////////////////////////////
```

xsm.tol de Antonio.Salmeron

Funciones para realizar una descripcion Xsm del sitemap de un sitio web utilizando las especificaciones de Google. Pone la misma prioridad para todas las paginas, ya que la prioridad es un valor relativo entre las paginas del sitio web, no frente a otros webs. Xsm son las siglas de Xsm Site Map, originalmente a las funciones de esta libreria se las identifico por Xml, pero se cambio por ser Xml un termino muy general.

Declaraciones

Variables de control

- `Text XsmPri`
  Prioridad, valor relativo entre paginas.
- `Text XsmFrq`
  Frecuencia de actualizacion del sitio web.
- `Set XsmTyp`
  Tipos de ficheros que se transmiten.
- `Text XsmExc`
  Directorio en el que no se busca, el de semillas.

Constantes

- `Text XsmHea`
  Semilla para la cebecera del sitemap en Xml.
- `Text XsmEnd`
  Texto final de un sitemap, etiqueta de cierre Xml.
- `Text XsmUrl`
  Semilla para la url de un fichero Xml de sitemap.

Funciones

- `Set XsmDateRepTab(Date dteFil)`
  Retorna una tabla de reemplazamientos para las fechas.
- `Real XsmDir(Text xsmFil, Text dirPth, Text urlDom)`
  Crea un sitemap con el contenido de un directorio, retorna el numero de ficheros includos en el sitemap.

Variables de control

Text XsmPri

```
////////////////////////////////////////////////////////////////////////////
Text XsmPri = "0.5"; // Da igual 1 que 0 es relativo entre las paginas
////////////////////////////////////////////////////////////////////////////
PutDescription("Prioridad, valor relativo entre paginas.", XsmPri);
////////////////////////////////////////////////////////////////////////////
```

Text XsmFrq

```
////////////////////////////////////////////////////////////////////////////
Text XsmFrq = "weekly"; // daily, weekly, monthly, ...
////////////////////////////////////////////////////////////////////////////
PutDescription("Frecuencia de actualizacion del sitio web.", XsmFrq);
////////////////////////////////////////////////////////////////////////////
```

Set XsmTyp

```
////////////////////////////////////////////////////////////////////////////
```

```
Set  XsmTyp = [["html", "pdf"]]; // Tipos Htm y Pdf
////////////////////////////////////////////////////////////////////////
PutDescription("Tipos de ficheros que se transmiten.", XsmTyp);
////////////////////////////////////////////////////////////////////////
```

Text XsmExc

```
////////////////////////////////////////////////////////////////////////
Text XsmExc = "/seed/"; // Exclusion
////////////////////////////////////////////////////////////////////////
PutDescription("Directorio en el que no se busca, el de semillas.", XsmExc);
////////////////////////////////////////////////////////////////////////
```

Constantes

Text XsmHea

```
////////////////////////////////////////////////////////////////////////
Text XsmHea =
"<?xml version='1.0' encoding='UTF-8'?>
<urlset xmlns='http://www.google.com/schemas/sitemap/0.84'>
<url>
  <loc>DOM</loc>
  <priority>"+XsmPri+"</priority>
  <lastmod>YEA-MTH-DAYTHOU:MIN:SEC+00:00</lastmod>
  <changefreq>"+XsmFrq+"</changefreq>
</url>";
////////////////////////////////////////////////////////////////////////
PutDescription("Semilla para la cebecera del sitemap en Xml.", XsmHea);
////////////////////////////////////////////////////////////////////////
```

Text XsmEnd

```
////////////////////////////////////////////////////////////////////////
Text XsmEnd = "
</urlset>"; // Parece que no se quiere el último salto de línea
////////////////////////////////////////////////////////////////////////
PutDescription("Texto final de un sitemap, etiqueta de cierre Xml.", XsmEnd);
////////////////////////////////////////////////////////////////////////
```

Text XsmUrl

```
////////////////////////////////////////////////////////////////////////
Text XsmUrl = "
<url>
  <loc>URL</loc>
  <priority>"+XsmPri+"</priority>
  <lastmod>YEA-MTH-DAYTHOU:MIN:SEC+00:00</lastmod>
  <changefreq>"+XsmFrq+"</changefreq>
</url>";
////////////////////////////////////////////////////////////////////////
PutDescription("Semilla para la url de un fichero Xml de sitemap.", XsmUrl);
////////////////////////////////////////////////////////////////////////
```

Set XsmDateRepTab()

```
////////////////////////////////////////////////////////////////////////
```

```
Set XsmDateRepTab(Date dteFil) // Fecha con segundo de un fichero
////////////////////////////////////////////////////////////////////////////
{
  //                            123456789.123456789.
  //                            y2005m07d20h17i40s35.00
  Text dteTxt = FormatDate(dteFil, "%cy%Ym%md%d%uh%hi%is%s");
  [[ SetOfText("YEA", Sub(dteTxt,  2,  5)),
     SetOfText("MTH", Sub(dteTxt,  7,  8)),
     SetOfText("DAY", Sub(dteTxt, 10, 11)),
     SetOfText("HOU", Sub(dteTxt, 13, 14)),
     SetOfText("MIN", Sub(dteTxt, 16, 17)),
     SetOfText("SEC", Sub(dteTxt, 19, 20)),
     SetOfText("'",   Char(34)           )
  ]]
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna una tabla de reemplazamientos para las fechas.",
XsmDateRepTab);
////////////////////////////////////////////////////////////////////////////
```

Real XsmDir()

```
////////////////////////////////////////////////////////////////////////////
Real XsmDir(Text xsmFil, // Fichero de salida
            Text dirPth, // Directorio a explorar
            Text urlDom) // Dominio con /, ie. http://www.omrforms.es/
////////////////////////////////////////////////////////////////////////////
{
  Text WriteLn("Output file: "+xsmFil+"\n"+
               "Input path:  "+dirPth+"\n"+
               "Domain:      "+urlDom);

  Set  setDir = EvalSet(XsmTyp, Set(Text filExt) // htm, html, pdf,...
  {
    Text WriteLn("Get files for: "+filExt);
    DirExtAll(dirPth, filExt, TRUE, TRUE)
  });
  Set  getDir = BinGroup("<<", setDir);

  Text WriteLn("Init Xml site map");
  Text WriteFile(xsmFil, ReplaceTable(XsmHea, // Domain and date
                 [[ [["DOM",urlDom]] ]] << XsmDateRepTab(Now)));

  Text WriteLn("Exclusions, not "+XsmExc);
  Set  getSel = Select(getDir, Real(Text pth) { !TextFind(pth, XsmExc) });

  Set  filCic = EvalSet(getSel, Real(Text filPth)
  {
    Text filUrl = Replace(filPth, "web/", urlDom);
    Date filDte = FileTime(filPth);
    Set  repTab = XsmDateRepTab(filDte) <<
    [[
       [[ "URL", filUrl ]]
    ]];
    Text AppendFile(xsmFil, ReplaceTable(XsmUrl, repTab));
    TRUE
  });

  Text AppendFile(xsmFil, XsmEnd);
  Text WriteLn("Xml site map: "+F(Card(filCic))+" pages");

  Card(filCic)
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Crea un sitemap con el contenido de un directorio, retorna el numero de
ficheros includos en el sitemap.",
XsmDir);
////////////////////////////////////////////////////////////////////////////
```

pdb.tol de Antonio.Salmeron

Posts database functions.

Declaraciones

Constantes

- `Text PdbSep`
  Post separator inside the agendas.

Funciones

- `Set PdbRead(Text inpDir)`
  Reads and returns a post database.
- `Set PdbFirstN(Set inpSet, Real maxNum, Code funSel)`
  Returns the maxNum recents posts selected using the function funSel.
- `Real PdbAppend(Text pstTxt)`
  Guarda el contenido de un post en un fichero. Esta funcion solo se emplea cuando es necesario convertir una agenda grande en varias pequeñas. A los ficheros se les pone extension txt y no age, pero poder revisarlos antes de la siguiente carga.

Constantes

Text PdbSep

```
////////////////////////////////////////////////////////////////////////////
Text PdbSep = Repeat("_",78);
////////////////////////////////////////////////////////////////////////////
PutDescription("Post separator inside the agendas.", PdbSep);
////////////////////////////////////////////////////////////////////////////
```

Set PdbRead()

```
////////////////////////////////////////////////////////////////////////////
Set PdbRead(Text inpDir)
////////////////////////////////////////////////////////////////////////////
{
  Real err(Text msg){ Text WriteLn("\nERROR: "+msg+"\n"); FALSE }; // Function

  Text filSep = Char(7);

  Text WriteLn("Reading "+inpDir+"...");
  Text inpAll = DirReadFiles(inpDir, "age", filSep);

  Text inpTxt = Replace(inpAll, PdbSep+"\n",filSep);
  Set  inpSet = Tokenizer(inpTxt,filSep);

  Set  inpTab = EvalSet(inpSet, Set(Text inf)
  {
    // Read
    Set  pstCla = Txt2Set(TxtBetween2Tag(inf,"<Pst.Cla>", "\n<", TRUE),";");
    Text pstNam =         TxtBetween2Tag(inf,"<Pst.Nam>", "\n<", TRUE);
    Text pstTit =         TxtBetween2Tag(inf,"<Pst.Tit>", "\n<", TRUE);
    Date pstDte =    Eval(TxtBetween2Tag(inf,"<Pst.Dte>", "\n<", TRUE));
```

```
    Date pstUpd =    Eval(TxtBetween2Tag(inf,"<Pst.Upd>", "\n<", TRUE));
    Text pstTxt =         TxtBetween2Tag(inf,"<Pst.Txt>", "\n<", FALSE);
    Text pstLnk =         TxtBetween2Tag(inf,"<Pst.Lnk>", "\n<", FALSE);

    // Check classes
    Set  chkCla = EvalSet(pstCla, Real(Text claNam) // Category class
    { If(claNam <: CatAll, TRUE, err("Class: "+pstNam+" "+claNam)) });

    Real chkPer = // Check periods and dates
    {
      Text chkLst = pstCla[Card(pstCla)]; // The period must be the last
      Set  chkSet = Tokenizer(chkLst, "-");
      Text chkIni = chkSet[1];
      Text chkEnd = chkSet[2];
      Text chkYea = FormatReal(Year(pstDte),"%4.0lf");
      If(And(chkYea >= chkIni, chkYea <= chkEnd), TRUE,
         err(pstNam+"Period and date: "+chkIni+" ?<= "+chkYea+" ?<= "+chkEnd))
    };

    Real chkNam = // Check names and dates
    {
      Text yeaNam = Sub(TxtBetween2Tag(pstNam, ",", "]", TRUE),1,4);
      Text yeaDte = FormatReal(Year(pstDte),"%4.0lf");
      If(yeaNam == yeaDte, TRUE,
         err("Name and date: "+pstNam+" ? "+yeaDte))
    };

    // Store and returns a post object
    PdbSt(pstCla, pstNam, pstTit, pstDte, pstUpd, pstTxt, pstLnk)
  });
  Set inpSor = Sort(inpTab, Real(Set a, Set b) // Lasts first
               { Compare(b->pstDte,a->pstDte) });

  // Check duplicated names
  Real chkDup =
  {
    Set inpCla = Classify(inpTab, Real(Set a, Set b) // By name
                 { Compare(a->pstNam,b->pstNam) });
    Set inpDup = EvalSet(inpCla, Real(Set cla)
    {
      Real crd = Card(cla);
      Text nam = cla[1]->pstNam;
      If(EQ(crd,1), TRUE, err("Name: "+nam+" "+
                              FormatReal(crd,"%.0lf")+" times"))
    });
    Card(inpDup)
  };

  Text WriteLn("Readed "+FormatReal(Card(inpSor),"%.0lf")+" registers");

  inpSor
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Reads and returns a post database.",
PdbRead);
////////////////////////////////////////////////////////////////////////////
```

Set PdbFirstN()

```
////////////////////////////////////////////////////////////////////////////
Set PdbFirstN(Set  inpSet, // Post database
              Real maxNum, // Maximum numbers of posts to return
              Code funSel) // Post selection conditions
////////////////////////////////////////////////////////////////////////////
{
  Set  selFst = Select(inpSet, funSel); // Select all that funSel()
  SetFirstN(selFst, maxNum)             // Fst maxNum or all if there are few
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns the maxNum recents posts selected using the function funSel.",
```

```
PdbFirstN);
////////////////////////////////////////////////////////////////////////////
```

Real PdbAppend()

```
////////////////////////////////////////////////////////////////////////////
Real PdbAppend(Text  pstTxt) // Post text
////////////////////////////////////////////////////////////////////////////
{
  Text pstOut = pstTxt + PdbSep + "\n";

  Text Case(
   !TextFind(pstTxt, "<Pst.Nam> [Salmerón A."),
     AppendFile(CtrAge+"/06.externos.txt", pstOut),       // Otros autores

   TextFind(pstTxt, "<iframe"),
     AppendFile(CtrAge+"/05.iframe.txt", pstOut),         // Videos

   TextFind(pstTxt, "; Educación básica;"),
     AppendFile(CtrAge+"/04.niñas.txt", pstOut),          // Niñas educacion

   TextFind(pstTxt, "<Pst.Cla> Pintura y poesía;"),
     AppendFile(CtrAge+"/03.arte.txt", pstOut),           // Arte

   TextFind(pstTxt, "<Pst.Cla> Negocios;"),
     AppendFile(CtrAge+"/02.negocios.txt", pstOut),       // Negocios

   TextFind(pstTxt, "<Pst.Cla> Tecnología;"),
     AppendFile(CtrAge+"/01.tecnologia.txt", pstOut),     // Tecnologia

   TRUE,
     AppendFile(CtrAge+"/00.inclasificable.txt", pstOut)  // Error
  );

  TRUE
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Guarda el contenido de un post en un fichero.
Esta funcion solo se emplea cuando es necesario convertir una agenda grande
en varias pequeñas.
A los ficheros se les pone extension txt y no age, pero poder revisarlos
antes de la siguiente carga.",
PdbAppend);
////////////////////////////////////////////////////////////////////////////
```

pht.tol de Antonio.Salmeron

Posts Html functions.

Declaraciones

inc.tol de Antonio.Salmeron

Inclusion of common functions and application functions

Declaraciones

Inclusiones comunes

- `Set txtInc`
  Text functions.
- `Set dteInc`
  Date functions.
- `Set setInc`
  Set functions.
- `Set filInc`
  File functions.
- `Set dirInc`
  Directory functions.
- `Set tmeInc`
  Macro expansor for Tol inside Html.
- `Set htmInc`
  Html functions.
- `Set ftpInc`
  Ftp functions.
- `Set xsmInc`
  Xml site maps functions.
- `Set pdfInc`
  Pdf functions.

Inclusiones de aplicación

- `Set pdbInc`
  Posts database funtions.
- `Set phtInc`
  Posts html functions.

Set txtInc

```
////////////////////////////////////////////////////////////////////////////
Set  txtInc = Include("cmm/txt.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Text functions.", txtInc);
////////////////////////////////////////////////////////////////////////////
```

Set dteInc

```
////////////////////////////////////////////////////////////////////////////
Set  dteInc = Include("cmm/dte.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Date functions.", dteInc);
////////////////////////////////////////////////////////////////////////////
```

Set setInc

```
////////////////////////////////////////////////////////////////////////////
Set  setInc = Include("cmm/set.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Set functions.", setInc);
////////////////////////////////////////////////////////////////////////////
```

Set filInc

```
////////////////////////////////////////////////////////////////////////////
Set  filInc = Include("cmm/fil.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("File functions.", filInc);
////////////////////////////////////////////////////////////////////////////
```

Set dirInc

```
////////////////////////////////////////////////////////////////////////////
Set  dirInc = Include("cmm/dir.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Directory functions.", dirInc);
////////////////////////////////////////////////////////////////////////////
```

Set tmeInc

```
////////////////////////////////////////////////////////////////////////////
Set  tmeInc = Include("cmm/tme.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Macro expansor for Tol inside Html.", tmeInc);
////////////////////////////////////////////////////////////////////////////
```

Set htmInc

```
////////////////////////////////////////////////////////////////////////////
Set  htmInc = Include("cmm/htm.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Html functions.", htmInc);
////////////////////////////////////////////////////////////////////////////
```

Set ftpInc

```
////////////////////////////////////////////////////////////////////////////
Set  ftpInc = Include("cmm/ftp.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Ftp functions.", ftpInc);
////////////////////////////////////////////////////////////////////////////
```

Set xsmInc

```
////////////////////////////////////////////////////////////////////////////
Set  xsmInc = Include("cmm/xsm.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Xml site maps functions.", xsmInc);
////////////////////////////////////////////////////////////////////////////
```

Set pdfInc

```
////////////////////////////////////////////////////////////////////////////
Set  pdfInc = Include("cmm/pdf.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Pdf functions.", pdfInc);
////////////////////////////////////////////////////////////////////////////
```

Set pdbInc

```
////////////////////////////////////////////////////////////////////////////
Set  pdbInc = Include("app/pdb.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Posts database funtions.", pdbInc);
////////////////////////////////////////////////////////////////////////////
```

Set phtInc

```
////////////////////////////////////////////////////////////////////////////
Set  phtInc = Include("app/pht.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Posts html functions.", phtInc);
////////////////////////////////////////////////////////////////////////////
```