make.tol de Omr.Forms

Es el programa constructor del sitio web OmrForms.es, que es un sitio dedicado a la comercialización de servicios y formularios en papel para la lectura óptica de datos. Los contenidos para este sitio web se estructuran en base a posts con 3 niveles de importancia (status) y que se guardan en un único fichero de texto denominado agenda de posts que es un banco de contenidos. Los 3 niveles de los posts son: a) A de Anulado, significa que está en el banco de contenidos pero que no se puplica, b) B de Bajo, se publica, pero no sale en el menú principal en la parte superior izquierda de todas las páginas web del sitio y c) C de Común, se publica y aparece enlazado desde el menú principal de todas las páginas.

Omr.Forms es un programa que puede considerarse básico en relación a otros constructores de sitios web desarrollados en lenguaje Tol y que se caracteriza por: a) Emplear una única agenda de posts que se alberga en el fichero de ruta agenda/agendadb.age. b) Los posts no tienen clases para su categorización, a diferencia de otros programas constructores de sitios webs. c) Se basa en un macro-expansor simple de Tol embebido en Html. d) Combina los títulos y los subtítulos, alternativos y disponibles para cada post, de forma que los contenido publicados no son siempre idénticos a los publicados anteriormente. e) La decoración lateral izquierda con ejemplos de formularios para lectoras ópticas de marcas es tambien aleatoria, por lo que es diferente en cada publicación. f) Las páginas las crea de una forma circular, siguiendo la misma secuencia de definición y las enlaza de tal manera, que empezando por cualquier página, leyendo hasta el final y pidiendo leer más, se pueden recorrer todas las páginas.

El programa OmrForms.es está basado en un macro-expansor simple de Tol en Html que permite embeber código Tol dentro de una página semilla Html, que opera a modo de template, y que hace que, al ejecutarse dicho código Tol, se generen con este template todas las páginas del sitio web con todos los diferentes contenidos de los posts de la agenda. El funcionamiento de este programa ha sido comprobado para las versiones de Tol 1.1.5, 1.1.6 y 2.0.1 y no funciona con la versión 1.1.1. Una de las causas de este no funcionamiento con la versión Tol 1.1.1 es la falta de visibilidad desde una macros de las variables definidas dentro de otras macros, esto es, que por ejemplo que la variable global CNT.ALL, que se define dentro de una macro, para la siguiente macro no existe, no la ve, eso si funciona en las versiones del lenguaje Tol más modernas.

Este programa OmrForms.es realiza automáticamente, entre otras funciones: a) la creación de índices de artículos, b) la extracción de las descripciones de las páginas, utilizando para ello los textos resaltados en dichos contenidos y c) la generación, para cada página, de la lista de palabras clave, keywords, extrayendo del contenido las de mayor frecuencia de aparición. De esta forma las descripciones y palabras clave, que declara en sus campos Html meta correspondientes, son los más adecuados y particulares para cada página, sin necesidad de que un humano realice esta tarea. Otras de las funciones principales de este programa OmrForms.es son: a) crear las páginas de control de errores, b) el mapa del sitio web en lenguaje Xml y c) comprobar qué páginas, documentos, imágenes, etc. han sido actualizadas con posterioridad al último envío por ftp y poner dichos nuevos contenidos online o actualizar los existentes.

Las opciones principales de Omr.Forms son: a) la generación de páginas mediante la opción denominada web, b) la creación del índice principal, homepage, y de las páginas absolutas, que valen para protección de directorios y control de errores, con la opción rot, c) la elaboración del mapa del sitio en Xml con la opción xsm, siglas de Xml site map, d) con las opciones ftp, fup y snd la actualización por Ftp del sitio, bien de todo el contenido (opciones ftp + snd, de send) o bien de las últimas páginas, imágenes y documentos actualizados (opciones fup + snd, de ftp update) y e) mediante la opción tst se realizan unos tests de la funcionalidad interna que comprueban ciertos funcionamientos del programa.

Árbol de ficheros

- Omr.Forms construye las páginas web Html del sitio web omrforms.es
  - ← make.tol proceso principal de generación del sitio web omrforms.es
  - tol directorios de código Tol
    - cmm funciones comunes
      - ← txt.tol de textos
      - ← set.tol de conjuntos
      - ← fil.tol de ficheros
      - ← dir.tol de directorios
      - ← tme.tol del macro-expansor simple de Tol en Html
      - ← ftp.tol para generar mandatos para hacer Ftp
      - ← xsm.tol para construir sitemaps en Xml
    - app funciones específicas de la aplicación
      - ← pdb.tol de manejo de los posts de una agenda
      - ← key.tol generación de palabras clave para páginas
    - ← inc.tol para la inclusión de ficheros Tol
  - agenda directorio destinado a la única agenda de post
    - ← agendadb.age ejemplo de un conjunto de posts de contenido para publicar
  - web directorio destinado a las páginas web generadas
    - css directorio para ficheros de estilo Cascade Style Sheet
      - ← common.css fichero de estilo para las páginas Html
    - ← seed.htm semilla de página Html con Tol embebido para generar otras
    - → sitemap.xml mapa del sitio web generado automáticamente en Xml
    - → preciospresupuestoslecturaoptica.html ejemplo del código Html de una de las páginas web generadas
  - → ejemplos.html ejemplos visuales de 3 páginas Html generadas automáticamente
  - → omr_forms.pdf documento resumen de funciones del programa constructor de web

Declaraciones

Inclusiones

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

Constantes

- `Text makSep`
  Linea horizontal para separar fases de operacion.
- `Text agePth`
  Ruta a la agenda con el conjunto de post del web.
- `Set frmTab`
  Conjunto de las imagenes de los formularios OMR.

Proceso

- `Text ctrExe`
  Argumento validado para la ejecucion del make.
- `Real makHlp`
  Es cierto si se ha visualizado la ayuda.
- `Real makArt`
  Crear todas las paginas de todos los articulos del web, retorna el numero de paginas creadas.
- `Real makRot`
  Crea la pagina raiz index, la de enlaces absolutos, la de errores y copia la comun de todos los web sites para enlaces cruzados.
- `Real makXsm`
  Crear el mapa del sitio, sitemap, en formato XML.
- `Real makFtp`
  Crear los ficheros de mandatos completos ftp o de actualizacion fup.
- `Real makSnd`
  Enviado ficheros utilizando los mandatos creados con ftp o fup (update).

Pruebas

- `Real makTst`
  Realizar el test de algunas funcionalidades necesarias para el proceso.

Set allInc

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

Estructuras de datos

```
Struct PdbSt // Estructura para cada uno de los post que contiene la agenda
{
  Text pstSta, // Status A(nulado), B(ajo, sin menu), C(comun, al menu)
  Text filNam, // Fichero para el post
  Text titLar, // Titulo largo del post
  Text titSor, // Titulo corto del post
  Text titSub, // Subtitulo del post
  Text pstHtm, // Texto del post es html
  Text pstDes  // Descripcion del post
};
```

Constantes

Text makSep

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

Text agePth

```
////////////////////////////////////////////////////////////////////////////
Text agePth = "agenda/agendadb.age";
////////////////////////////////////////////////////////////////////////////
PutDescription("Ruta a la agenda con el conjunto de post del web.",agePth);
////////////////////////////////////////////////////////////////////////////
```

Set frmTab

```
////////////////////////////////////////////////////////////////////////////
Set  frmTab =
{
  Text ageTxt = ReadFile(agePth);
  Text ageImg = TxtBetween2Tag(ageTxt,          // Text
                               "<!-- FRM -->", // Initial tag
                               "<!-- END -->", // End tag
                               TRUE);
  Set  ageSet = Tokenizer(ageImg,">");
  Set  ageCic = EvalSet(ageSet, Set(Text imgTxt)
  {
    Text iniTag = "<img class="+Q("FrmCnt")+" src="+Char(34);
    Text endTag = Char(34);
    Text altTag = "alt="+Char(34);
    Text titTag = "title="+Char(34);

    Text imgPth = TxtBetween2Tag(imgTxt, iniTag, endTag, TRUE);
    Text imgAlt = TxtBetween2Tag(imgTxt, altTag, endTag, TRUE);
    Text imgTit = TxtBetween2Tag(imgTxt, titTag, endTag, TRUE);
    If(Or(imgPth=="", imgAlt=="", imgTit==""), Empty,
       SetOfText(imgPth, imgAlt, imgTit))
  });
  Select(ageCic, Real(Set imgSet) { EQ(Card(imgSet), 3) })
};
////////////////////////////////////////////////////////////////////////////
PutDescription("Conjunto de las imagenes de los formularios OMR.",frmTab);
////////////////////////////////////////////////////////////////////////////
```

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 omrforms.es site
  OPTION
    art: build all articles pages
    rot: build root absolute page
    xsm: build Xml site map
    ftp: build ftp files (go to ftp dir and run manually or use make snd)
    fup: build a file update protocol (newer than ftp.log file)
    snd: send file via ftp/ftu
    ___
    web: build local web: art, rot, xms
    all: do all works: art, rot, xms, fup y snd.
    hlp: show this help
    tst: test some functions");
  TRUE
});
////////////////////////////////////////////////////////////////////////////
PutDescription("Es cierto si se ha visualizado la ayuda.",makHlp);
////////////////////////////////////////////////////////////////////////////
```

Real makArt

```
////////////////////////////////////////////////////////////////////////////
Real makArt = If(!(ctrExe <: [["all", "web", "art"]]), FALSE,
{
  Text WriteLn(makSep+"building all web pages...");
  Real CtrMax =   5; // Numero de posts por pagina
  Real CtrCxD = 360; // Caracteres por descripcion, LinkedIn 250, Facebook 300
  Real CtrKxP =  36; // Palabras clave por pagina
  Set  CtrPdb = PdbRead(agePth); // Leer todos los posts

  Set  CtrBas = Select(CtrPdb, Real(Set pstObj) { pstObj->pstSta == "B" });
  Set  CtrCmn = Select(CtrPdb, Real(Set pstObj) { pstObj->pstSta == "C" });

  Text WriteLn("Status __C "+F(Card(CtrCmn))+" registers");

  Text CtrInC = PdbIndex(CtrCmn); // Articulos comunes para indices lateral
  Text CtrInB = PdbIndex(CtrBas); // Articulos basicos
  Text CtrInd = PdbIndex(CtrPdb); // Todos los articulos

  Text inpPth = "web/seed.htm";
  Text inpHtm = ReadFile(inpPth);

  Set  wriCic = For(1, Card(CtrPdb), Real(Real pdbPos)
  {
    Set  CtrPst = CtrPdb[pdbPos];
    Text CtrPth = "web/"+CtrPst->filNam+".html";
    Text WriteLn("-> "+CtrPth);

    Set  CtrCic = SetSubCicle(CtrPdb, pdbPos, CtrMax+1); // +1 link siguiente
```

```
    Set  CtrFrm = PdbFrm(pdbPos, CtrMax, frmTab, TRUE);  // Aleatorio
    Text CtrKey = PdbTxtKeyword(CtrCic, CtrKxP);         // Aleatorio nº+-6

    TmeFile(inpPth, CtrPth)
  });

  Real wriAll = // Una pagina con todos los articulos para revisiones
  {
    Set  CtrPst = CtrPdb[1]; // El primero
    Text CtrPth = "web/completo.html";
    Text WriteLn("-> "+CtrPth);

    Set  CtrCic = SetSubCicle(CtrPdb, 1, Card(CtrPdb)+1); // +1 link al 1º
    Set  CtrFrm = PdbFrm(1, CtrMax, frmTab, FALSE); // Secuencial
    Text CtrKey = PdbTxtKeyword(CtrCic, 40);

    TmeFile(inpPth, CtrPth)
  };

  Card(wriCic)+wriAll
});
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Crear todas las paginas de todos los articulos del web, retorna el numero de
paginas creadas.",
makArt);
////////////////////////////////////////////////////////////////////////////
```

Real makRot

```
////////////////////////////////////////////////////////////////////////////
Real makRot = If(!(ctrExe <: [["all", "web", "rot"]]), FALSE,
{
  Text WriteLn(makSep+"building root pages...");

  Text WriteLn("-> main index");
  Text inxHtm = ReadFile("web/desarrolloimpresionformulariosomr.html");
  Text WriteFile("web/index.html", inxHtm);

  Text WriteLn("-> absolute page and 404 page");
  Text extRef = " href="+Char(34)+"http";
  Text savRef = " _XX_="+Char(34)+"_XX_";
  Text absSav = Replace(inxHtm, extRef, savRef); // Salvar enlaces externos
  Text absRep = TxtReplaceSecuence(absSav,
  [[
     [[ " src=\"", "  src=\"http://www.omrforms.es/"]],
     [[" href=\"", " href=\"http://www.omrforms.es/"]]
  ]]);
  Text absRec = Replace(absRep, savRef, extRef); // Restaurarlos
  Text WriteFile("web/absoluto.html", absRec);
  Text WriteFile("web/error404.html", absRec);

  Text WriteLn("-> common directory page, utf-8");
  FilCopy("../common/dir.html", "web/comxidir.html", TRUE)
});
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Crea la pagina raiz index, la de enlaces absolutos, la de errores y
copia la comun de todos los web sites para enlaces cruzados.",
makRot);
////////////////////////////////////////////////////////////////////////////
```

Real makXsm

```
////////////////////////////////////////////////////////////////////////////
Real makXsm = If(!(ctrExe <: [["all", "web", "xsm"]]), FALSE,
{
  Text WriteLn(makSep+"building Xml site map...");
```

```
  XsmDir("web/sitemap.xml", "web", "http://www.omrforms.es/")
});
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Crear el mapa del sitio, sitemap, en formato XML.",
makXsm);
////////////////////////////////////////////////////////////////////////////
```

Real makFtp

```
////////////////////////////////////////////////////////////////////////////
Real makFtp = If(!(ctrExe <: [["all", "ftp", "fup"]]), FALSE,
{
  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.omrforms.es",          // 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
      ["css",         "css",      "css",    FALSE]],  // ASCII
      ["src",         "js",       "js",     FALSE]],  // ASCII  javascript
      ["img",         "gif",      "gif",    TRUE]],   // Binary
      ["tecnica",     "png",      "tec",    TRUE]],   // Binary
      ["camposomr",   "png",      "cam",    TRUE]],   // Binary
      ["documentos",  "png",      "docpng", TRUE]],   // Binary
      ["documentos",  "pdf",      "docpdf", TRUE]],   // Binary
      ["formularios","gif",       "frm",    TRUE]]    // Binary
    ]])
});
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Crear los ficheros de mandatos completos ftp o de actualizacion fup.",
makFtp);
////////////////////////////////////////////////////////////////////////////
```

Real makSnd

```
////////////////////////////////////////////////////////////////////////////
Real makSnd = If(!(ctrExe <: [["all", "snd"]]), FALSE,
{
  Text WriteLn(makSep+"sending files using ftp/ftu...");
  System(W("ftp/Omr.Forms.bat"))
});
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Enviado ficheros utilizando los mandatos creados con ftp o fup (update).",
makSnd);
////////////////////////////////////////////////////////////////////////////
```

Real makTst

```
////////////////////////////////////////////////////////////////////////////
Real makTst = If(ctrExe != "tst", FALSE,
{
  Text WriteLn(makSep+"test function Txt2Set()");
```

```
  Text tstT2S = " b  b :: aa :: b b :: cc:: ::::d::cc::";
  Text WriteLn(F(Txt2Set(tstT2S, "::", "")));
  Text WriteLn(F(Txt2Set(tstT2S, "::", "C")));
  Text WriteLn(F(Txt2Set(tstT2S, "::", "U")));
  Text WriteLn(F(Txt2Set(tstT2S, "::", "CU")));
  Text WriteLn(F(Txt2Set(tstT2S, "::", "CUN")));
  Text WriteLn(F(Txt2Set(tstT2S, "::", "CNU")));
  Text WriteLn(F(Txt2Set(tstT2S, "::", "CNUS")));
  Text WriteLn(F(Txt2Set(tstT2S, "::", "S")));

  Text WriteLn(makSep+"test function TxtOutside2Tag()");

  Text WriteLn(TxtOutside2Tag("
    a <i>b</i> c
    <img src='...' alt='alternativo uno' title='titulo uno'> d
    <img src='...' title="+Q("titulo dos")+" alt="+Q("alternativo dos")+"> e
    <script...>var x=33;</script> f
  ", "<", ">"));

  Text WriteLn(makSep+"test function TxtOutHtmTag()");

  Text WriteLn(TxtOutHtmTag("
    a <i>b</i> c
    <img src='...' alt='alternativo uno' title='titulo uno'> d
    <img src='...' title="+Q("titulo dos")+" alt="+Q("alternativo dos")+"> e
    <script...>var x=33;</script> f
  "));

  TRUE
});
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Realizar el test de algunas funcionalidades necesarias para el proceso.",
makTst);
////////////////////////////////////////////////////////////////////////////
```

txt.tol de Omr.Forms

Funciones de texto.

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. Se basa en una version especial de la funcion Txt2Set().
- `Text F(Anything anyVal)`
  Retorna numeros, fechas, textos, conjuntos como un texto de formato simple.

Funciones

- `Set Txt2Set(Text txtInp, Text sepTok, Text ctrFun)`
  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.
- `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 TxtInside2Tag(Text txtInp, Text tagIni, Text tagEnd)`
  Retorna todo el texto entre 2 tags (tagIni y tagEnd) dentro de txtIpn. Por ejemplo: <aaa(::)bbb(---)ccc>, <(>, <)> -> <::--->.
- `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, de los scripts e incluye el que esta dentro de los alt y los titles de las imagenes. Sirve para extraer texto limpio del que sacar palabras clave.
- `Set TxtLineWrap(Text txtInp, Real linMax, Real cmpCtr)`

Retorna un conjunto de 2 texto el primero con un máximo de linMax caracteres y el segundo con el resto. Es el resultado de cortar txtInp por el primer blanco que permita que el corte cumpla la condición inicial. Si el texto de entrada es mas corte que linMax retorna un conjunto formado por el texto inicial y la tira vacia. Si el corte es imposible busca el mejor corte posible y si no lo encuentra retorna un conjunto formado por el texto inicial y la tira vacia. Si cmpCtr es true los resultados son compactados. Tambien existe en Tol la funcion Wrap() con ciertas semejanzas, aunque mas a TxtParagraphWrap().

- `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) // Set of texts
////////////////////////////////////////////////////////////////////////////
{ Anything SetGetRand(Txt2Set(txtLst, "|", "CN")) };
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un texto elegido al azar de entre los tokens (|) de otro texto.
Se basa en una version especial de la funcion Txt2Set().",
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
            Text ctrFun) // C->Compacta, N->Not null, U->Unicos, S->Ordena
////////////////////////////////////////////////////////////////////////
{
  Text ctrUpp = ToUpper(ctrFun);

  Set  setSep = Tokenizer(Replace(txtInp, sepTok, Char(7)), Char(7));

  Set  setCmp = If(!TextFind(ctrFun, "C"), setSep,
                   EvalSet(setSep, Text(Text eleTxt) { Compact(eleTxt) }));

  Set  setNot = If(!TextFind(ctrFun, "N"), setCmp,
                   Select(setCmp, Real(Text eleTxt) { eleTxt != "" }));

  Set  setUni = If(!TextFind(ctrFun, "U"), setNot, Unique(setNot));

  Set  setSrt = If(!TextFind(ctrFun, "S"), setUni,
                   Sort(setUni, Real(Text a, Text b) { Compare(a,b) }));

  setSrt
};
////////////////////////////////////////////////////////////////////////
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);
////////////////////////////////////////////////////////////////////////////
```

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 TxtInside2Tag()

```
////////////////////////////////////////////////////////////////////////////
Text TxtInside2Tag(Text txtInp, // Texto de entrada
                   Text tagIni, // Tag inicial
                   Text tagEnd) // Tag final
////////////////////////////////////////////////////////////////////////////
{
  Real posIni = TextFind(txtInp, tagIni);
  Text result = If(LE(posIni,0), "",
  {
    Real lenIni = TextLength(tagIni);
    Real posSub = posIni + lenIni;
    Real posEnd = TextFind(txtInp, tagEnd, posSub);
    If(And(EQ(posIni,1),LE(posEnd,0)), txtInp,
    If(And(GT(posIni,1),LE(posEnd,0)), Sub(txtInp,posIni, TextLength(txtInp)),
       Sub(txtInp,posIni,posEnd+TextLength(tagEnd)-1)+ // Recursion
       TxtInside2Tag(Sub(txtInp, posEnd+TextLength(tagEnd), TextLength(txtInp))
```

```
                    tagIni, tagEnd)))
  });
  ReplaceTable(result, [[ [[tagIni, ""]], [[tagEnd, ""]] ]])
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna todo el texto entre 2 tags (tagIni y tagEnd) dentro de txtIpn.
Por ejemplo: <aaa(::)bbb(---)ccc>, <(>, <)> -> <::--->.",
TxtInside2Tag);
////////////////////////////////////////////////////////////////////////////
```

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)
////////////////////////////////////////////////////////////////////////////
{
  Text notScr = TxtOutside2Tag(htmTxt, "<script", "</script>"); // No scripts
  Text notAmp = TxtOutside2Tag(notScr, "&",       ";");         // No &xxx;
  Text notHtm = TxtOutside2Tag(notAmp, "<",       ">");         // No html

  Text htmQuo = Replace(htmTxt, Char(34), "'"); // Unifica comillas
  Text difSep = Char(7);    // Separador no usual
  Text difEnd = difSep+" "; // Terminador no usual

  Text titTag = "title='";  // Titulos de la imagenes separados por un blanco
  Text titTxt = TxtInside2Tag(Replace(htmQuo, titTag, difEnd), difSep, "'");

  Text altTag = "alt='";    // Alts de la imagenes separados por un blanco
  Text altTxt = TxtInside2Tag(Replace(htmQuo, altTag, difEnd), difSep, "'");

  Compact(notHtm + titTxt + altTxt)
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna todos el texto fuera de los tags de html, de los scripts e incluye
el que esta dentro de los alt y los titles de las imagenes.
Sirve para extraer texto limpio del que sacar palabras clave.",
TxtOutHtmTag);
////////////////////////////////////////////////////////////////////////////
```

Set TxtLineWrap()

```
////////////////////////////////////////////////////////////////////////////
Set  TxtLineWrap(Text txtInp, // Texto de entrada
```

```
                Real linMax, // Maximo numero de caracteres por linea
                Real cmpCtr) // Si true entonces compacta
////////////////////////////////////////////////////////////////////////////
{
  Text txtCmp = If(cmpCtr, Compact(txtInp), txtInp);
  Text txtRev = Reverse(txtCmp);
  Real txtLen = TextLength(txtCmp);
  Set  cutSet = If(LE(txtLen, linMax), [[txtCmp, ""]], // Ya esta hecho
  {
    Real blkPos = TextFind(txtRev, " ", txtLen-linMax); // Busca para atras

    If(GE(blkPos, 1),
    {
      SetOfText(Sub(txtCmp, 0,               txtLen-blkPos),
                Sub(txtCmp, txtLen-blkPos+1, txtLen))
    },
    {
      // No se puede cortar
      Real blkBad = TextFind(txtCmp, " ", linMax+1); // Busca hacia adelante

      If(LT(blkBad, 0), [[txtCmp, ""]], // No hay corte posible
      {
        SetOfText(Sub(txtCmp, 0,         blkBad-1), // Hay un mal corte
                  Sub(txtCmp, blkBad+1, txtLen))
      })
    })
  });
  If(cmpCtr, SetOfText(Compact(cutSet[1]),Compact(cutSet[2])), cutSet)
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un conjunto de 2 texto el primero con un máximo de linMax caracteres
y el segundo con el resto.
Es el resultado de cortar txtInp por el primer blanco que permita que el corte
cumpla la condición inicial.
Si el texto de entrada es mas corte que linMax retorna un conjunto formado
por el texto inicial y la tira vacia.
Si el corte es imposible busca el mejor corte posible y si no lo encuentra
retorna un conjunto formado por el texto inicial y la tira vacia.
Si cmpCtr es true los resultados son compactados.
Tambien existe en Tol la funcion Wrap() con ciertas semejanzas,
aunque mas a TxtParagraphWrap().",
TxtLineWrap);
////////////////////////////////////////////////////////////////////////////
```

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

set.tol de Omr.Forms

Funciones de conjuntos.

Declaraciones

Funciones

- `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 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 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]

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 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 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 Omr.Forms

Funciones de ficheros.

Declaraciones

Funciones

- `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 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 Omr.Forms

Funciones de directorios.

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.

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

tme.tol de Omr.Forms

Macro-expansor basico 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 basica que permite inyectar Tol dentro de semillas Html del tipo seed.htm pero no dentro del Html de las agendas de post, esto es, que al expandir Tol dentro del Html aparezca no solo codigo Html sino, de nuevo, codigo Tol, existe una version que si lo admite.

Declaraciones

Constantes

- `Text TmeIni`
  Tag inicial por defecto, puede usarse otro.
- `Text TmeEnd`
  Tag final por defecto, puede usarse otro.
- `Text TmeSep`
  Caracter de uso interno que usa Tokenizer().
- `Text TmeEmpty`
  Todo el codigo inyectado ha de retornar algun texto. Si el codigo solo tiene definiciones, sin retornar nada util, siempre se puede retornar TmeEmpty.

Funciones

- `Set TmeGetMacros(Text tagIni, Text tagEnd, Text codTxt)`
  Retorna un conjunto con todo el codigo entre los tags inicial y final.
- `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 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 This version does not use FilWriteIfDiff(), it always update all web site.
- `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("Tag inicial por defecto, puede usarse otro.", TmeIni);
////////////////////////////////////////////////////////////////////////////
```

Text TmeEnd

```
////////////////////////////////////////////////////////////////////////////
Text TmeEnd = "}"+">";
////////////////////////////////////////////////////////////////////////////
PutDescription("Tag final por defecto, puede usarse otro.", TmeIni);
////////////////////////////////////////////////////////////////////////////
```

Text TmeSep

```
////////////////////////////////////////////////////////////////////////////
Text TmeSep = Char(7);
////////////////////////////////////////////////////////////////////////////
PutDescription("Caracter de uso interno que usa Tokenizer().", TmeSep);
////////////////////////////////////////////////////////////////////////////
```

Text TmeEmpty

```
////////////////////////////////////////////////////////////////////////////
Text TmeEmpty = "";
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Todo el codigo inyectado ha de retornar algun texto. Si el codigo solo tiene
definiciones, sin retornar nada util, siempre se puede retornar 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 aren't 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(
"Retorna un conjunto con todo el codigo entre los tags inicial y final.",
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 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));

  Text wriFil = If(outFil != "", WriteFile(outFil, outRep), "");
  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
This version does not use FilWriteIfDiff(), it always update all web site.",
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);
////////////////////////////////////////////////////////////////////////////
```

ftp.tol de Omr.Forms

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

xsm.tol de Omr.Forms

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 Omr.Forms

Funciones para una agenda de posts.

Declaraciones

Constantes

- `Text PdbSep`
  Separador de los post en la agenda de contenidos.

Funciones

- `Set PdbRead(Text inpFil)`
  Lee y retorna una estructura de posts en el mismo orden de lectura la agenda. Los posts anulados, con su estado a A no son retornados. Si un post no tiene estado se asume el estado C, que es el común.
- `Text PdbIndex(Set setPdb)`
  Retorna un indice para un conjunto de posts.
- `Text PdbTxtKeyword(Set setPdb, Real maxWrd)`
  Retorna un texto con las maxWrd keywords de un conjunto de posts.
- `Set PdbFrm(Real pdbPos, Real ctrMax, Set frmTab, Real ctrRnd)`
  Retorna un conjunto de ctrMax imagenes de formularios y los puede elegir de forma secuencial a partir de pdbPos o de forma aleatoria.
- `Text PdbGetRand(Text inpTxt, Text iniTag)`
  Retorna un texto del campo que comienza con fldIni del texto pstTxt de un post, si existen varias altertanivas de texto retorna una al azar. Se retorna compactado.
- `Text PdbGetDescription(Text inpTxt)`
  Retorna la descripcion del post que es el parrafo principal.

Constantes

Text PdbSep

```
////////////////////////////////////////////////////////////////////////////
Text PdbSep = Repeat("_",78);
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Separador de los post en la agenda de contenidos.",
PdbSep);
////////////////////////////////////////////////////////////////////////////
```

Set PdbRead()

```
////////////////////////////////////////////////////////////////////////////
Set PdbRead(Text inpFil)
////////////////////////////////////////////////////////////////////////////
{
  Text WriteLn("Reading "+inpFil+"...");
```

```
  Text inpTxt = Replace(ReadFile(inpFil), PdbSep+"\n", Char(7));
  Set  inpSet = Tokenizer(inpTxt, Char(7));
  Text fldEnd = "\n<Pst.";
  Set  inpTab = EvalSet(inpSet, Set(Text inpPst)
  {
    Text iniSta = TxtBetween2Tag(inpPst,"<Pst.Sta>", fldEnd, TRUE);
    Text pstSta = Sub(iniSta+"C",1,1); // C por defecto

    Text filNam = TxtBetween2Tag(inpPst,"<Pst.Fil>", fldEnd, TRUE);

//  Antes se sorteaba en lectura un titulo que se usaba en todas las páginas.
//  Ahora se sortea en escritura por lo que la aleatoriedad es aun mayor.
//  Text titLar = PdbGetRand    (inpPst,"<Pst.Tit>");
    Text titLar = TxtBetween2Tag(inpPst,"<Pst.Tit>", fldEnd, TRUE);

//  Text iniSor = PdbGetRand(    inpPst,"<Pst.Lbl>");
    Text iniSor = TxtBetween2Tag(inpPst,"<Pst.Lbl>", fldEnd, TRUE);
    Text titSor = If(iniSor!="", iniSor, titLar); // Si no hay -> el largo

//  Text titSub = PdbGetRand    (inpPst,"<Pst.Sub>");
    Text titSub = TxtBetween2Tag(inpPst,"<Pst.Sub>", fldEnd, TRUE);

    Text pstHtm = TxtBetween2Tag(inpPst,"<Pst.Htm>", fldEnd, FALSE);

    Text pstDes = PdbGetDescription(pstHtm);

    PdbSt(pstSta, filNam, titLar, titSor, titSub, pstHtm, pstDes)
  });

  Set  inpSel = Select(inpTab, Real(Set pstObj)
                       { pstObj->pstSta != "A" }); // No Anulados
  Set  selSet = Select(inpSel, Real(Set pstObj)
                       { pstObj->filNam != "" });  // Con nombre de fichero

  Set  selCla = Classify(selSet, Real(Set a, Set b) // Check ficheros dobles
                { Compare(a->filNam, b->filNam) });
  Set  selDup = Select(selCla, Real(Set claSet) { GT(Card(claSet),1) });
  Set  wriDup = EvalSet(selDup, Real(Set claSet)
                { Text WriteLn("Repeated: "+claSet[1]->filNam); TRUE });

  Text WriteLn("Readed Ok  "+F(Card(selSet))+" registers");
  Text WriteLn("Status ABC "+F(Card(inpTab))+" registers");
  Text WriteLn("Status _BC "+F(Card(inpSel))+" registers");

  selSet
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Lee y retorna una estructura de posts en el mismo orden de lectura la agenda.
Los posts anulados, con su estado a A no son retornados.
Si un post no tiene estado se asume el estado C, que es el común.",
PdbRead);
////////////////////////////////////////////////////////////////////////////
```

Text PdbIndex()

```
////////////////////////////////////////////////////////////////////////////
Text PdbIndex(Set setPdb) // Set of post
////////////////////////////////////////////////////////////////////////////
{
  Set indCic = EvalSet(setPdb, Text(Set pdbObj) // Index
  {
    "    <li><a href='" +   pdbObj->filNam  + ".html'>" +
                        R(pdbObj->titLar) + "</a>" +
            ": "+ R(pdbObj->titSub) + ".</li>\n"
  });
  "\n  <ul>\n" + SetSum(indCic) + "  </ul>\n"
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un indice para un conjunto de posts.",
PdbIndex);
```

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

Text PdbTxtKeyword()

```
////////////////////////////////////////////////////////////////////////////
Text PdbTxtKeyword(Set  setPdb, // Set of post
                   Real maxWrd) // Average of word for a random number
////////////////////////////////////////////////////////////////////////////
{
  Text setKey = KeyWord(
                  EvalSet(setPdb, Text(Set pst)
                  {
                    pst->titLar+" "+
                    pst->titSub+" "+
                    pst->titSor+" "+
                    TxtOutHtmTag(pst->pstHtm)
                  }),
                  5, FALSE, Round(Rand(maxWrd-5, maxWrd+5)));

  ReplaceTable(setKey,
   [[ [["antonio",  ""]],
      [["salmerón", ""]],
      [["cabañas",  ""]],
      [[", ",       ","]] ]])
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un texto con las maxWrd keywords de un conjunto de posts.",
PdbTxtKeyword);
////////////////////////////////////////////////////////////////////////////
```

Set PdbFrm()

```
////////////////////////////////////////////////////////////////////////////
Set PdbFrm(Real pdbPos, // Post position
           Real ctrMax, // Maximum of images
           Set  frmTab, // Forms tab
           Real ctrRnd) // If true ramdom, else secuential
////////////////////////////////////////////////////////////////////////////
{
  Real frmCrd = Card(frmTab);

  Real frmIni = If(ctrRnd,
    Min(frmCrd, Max(1, Round(Rand(0,frmCrd)+0.5))), // Aleatorio
    (1+(pdbPos-1)*CtrMax) % Card(frmTab));          // Secuencial

  SetSubCicle(frmTab, frmIni, CtrMax)
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un conjunto de ctrMax imagenes de formularios y los puede elegir de
forma secuencial a partir de pdbPos o de forma aleatoria.",
PdbFrm);
////////////////////////////////////////////////////////////////////////////
```

Text PdbGetRand()

```
////////////////////////////////////////////////////////////////////////////
Text PdbGetRand(Text inpTxt, // Texto de un post
                Text iniTag) // Inicio del nombre del campo
////////////////////////////////////////////////////////////////////////////
{
  Text subTxt = TxtBetween2Tag(inpTxt, iniTag, "\n<Pst.", TRUE);
  If(subTxt=="", "", R(subTxt))
```

```
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un texto del campo que comienza con fldIni del texto pstTxt de un
post, si existen varias altertanivas de texto retorna una al azar.
Se retorna compactado.",
PdbGetRand);
////////////////////////////////////////////////////////////////////////////
```

Text PdbGetDescription()

```
////////////////////////////////////////////////////////////////////////////
Text PdbGetDescription(Text inpTxt) // Texto de un post
////////////////////////////////////////////////////////////////////////////
{
  Text iniTag = "<p class="+Q("PutDes")+">";
  Text subTxt = TxtBetween2Tag(inpTxt, iniTag, "</p>", TRUE);
  Text clsTxt = TxtOutHtmTag(subTxt);
  Compact(clsTxt)
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna la descripcion del post que es el parrafo principal.",
PdbGetDescription);
////////////////////////////////////////////////////////////////////////////
```

key.tol de Omr.Forms

Funciones para palabras clave.

Declaraciones

Funciones

- `Text KeyWord(Set valSet, Real minChr, Real a2zOrd, Real maxKey)`

  Retorna un texto formado por una lista de palabras separadas por comas. Selecciona las maxKey palabras mas frecuentes de un conjunto de textos u otros valores que intenta convertir a texto. Las palabras mas frecuentes las ordena alfabeticamente si a2zOrd. Retira todas las palabras que no superen la longitud minChr y todas las que aparecen en una tabla interna de palabras comunes.

Text KeyWord()

```
////////////////////////////////////////////////////////////////////////////
Text KeyWord(Set  valSet, // Set of texts
             Real minChr, // Minimum number of chars
             Real a2zOrd, // If true ordered
             Real maxKey) // Maximun number of keywords
////////////////////////////////////////////////////////////////////////////
{
  Text allTxt = ToLower(F(valSet));   // Todo a texto
  Text fixWit = Replace(allTxt, " ", " "); // Unico &xxx; que respeta
  Text notHtm = TxtOutHtmTag(fixWit); // Fuera de html y scripts, sin &xxx;

  Text clsTxt = Compact(ReplaceTable(notHtm,
    [[ [[Char(34), " "]], [["'",      " "]],
       [[".",      " "]], [[":",      " "]],
       [[";",      " "]], [[",",      " "]],
       [["[",      " "]], [["]",      " "]],
       [["(",      " "]], [[")",      " "]],
       [["«",      " "]], [["»",      " "]],
       [["¡",      " "]], [["!",      " "]],
       [["¿",      " "]], [["?",      " "]],
       [["/",      " "]], [["\\",     " "]]
    ]]));

  Set  setTxt = Tokenizer(clsTxt, " "); // Romper por palabras

  // Selecciona palabras con mas de minChr caracteres
  Set  setSel = Select(setTxt, Real(Text oneWrd)
                { GT(TextLength(oneWrd), minChr) });

  // Clasificar por palabras
  Set  setCla = Classify(setSel, Real(Text a, Text b) { Compare(a,b) });

  // Tabla de frecuencias [palabra, ocurrencias]
  Set  tabFrq = EvalSet(setCla, Set(Set oneCla)
                        { [[ oneCla[1], Card(oneCla) ]] });

  // Ordenar por ocurrencia de mas a menos
  Set  srtFrq = Sort(tabFrq, Real(Set a, Set b)
                     { Compare(Real(b[2]), Real(a[2])) });

  // Proyectar por la columna de palabras que es la primera
  Set  keySet = EvalSet(srtFrq, Text(Set wrdFrq) { wrdFrq[1] });

  Set  keyMin = keySet - // Palabras comunes a retirar
  [[
     "además",
     "ambos", "ambas",
```

```
      "aunque",
      "cualquier", "cualquiera",
      "cuando", "cuándo",
      "deberá",
      "desde",
      "dentro",
      "entre",
      "esta", "estas", "esto", "estos",
      "junto",
      "mismo", "mismos", "misma", "mismas",
      "mucho", "muchos", "mucha", "muchas",
      "ningún", "ninguna",
      "nosotros", "nosotras",
      "nuestro", "nuestros", "nuestra", "nuestras",
      "otra", "otras", "otro", "otros",
      "parte",
      "porque",
      "puede", "pueden", "pudiendo",
      "quienes",
      "siempre",
      "siguiente", "siguientes",
      "sobre",
      "sumamente",
      "también",
      "tanto",
      "través",
      "zulú"
    ]];

     // Seleccionar las maxKey palabras o todas si maxKey == 0
    Set  keyFst = If(maxKey, SetFirstN(keyMin, maxKey), keyMin);

    // Ordenar alfabeticamente si se necesita
    Set  keySor = If(!a2zOrd, keyFst, Sort(keyFst, Real(Text a, Text b)
                                           { Compare(a,b) }));

    // Convertir a texto separado por comas
    ReplaceTable(F(keySor), [[ [["[", ""]], [["]", ""]], [["|", ", "]] ]])
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un texto formado por una lista de palabras separadas por comas.
Selecciona las maxKey palabras mas frecuentes de un conjunto de textos u
otros valores que intenta convertir a texto.
Las palabras mas frecuentes las ordena alfabeticamente si a2zOrd.
Retira todas las palabras que no superen la longitud minChr y todas las que
aparecen en una tabla interna de palabras comunes.",
KeyWord);
////////////////////////////////////////////////////////////////////////////
```

inc.tol de Omr.Forms

Inclusion de las funciones comunes y de aplicacion.

Declaraciones

Inclusiones comunes

- `Set txtInc`
  Incluir funciones de textos.
- `Set setInc`
  Incluir funciones de conjuntos.
- `Set filInc`
  Incluir funciones de ficheros.
- `Set dirInc`
  Incluir funciones de directorios.
- `Set tmeInc`
  Inclusion del macro-expansor simple de Tol en Html.
- `Set ftpInc`
  Incluir funciones para generar mandatos Ftp.
- `Set xsmInc`
  Incluir funciones para construir sitemaps en Xml.

Inclusiones de aplicación

- `Set pdbInc`
  Funciones de manejo de los post de la agenda.
- `Set keyInc`
  Funciones generar palabras clave de páginas.

Set txtInc

```
////////////////////////////////////////////////////////////////////////////
Set  txtInc = Include("cmm/txt.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Incluir funciones de textos.",txtInc);
////////////////////////////////////////////////////////////////////////////
```

Set setInc

```
////////////////////////////////////////////////////////////////////////////
Set  setInc = Include("cmm/set.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Incluir funciones de conjuntos.",setInc);
////////////////////////////////////////////////////////////////////////////
```

Set filInc

```
////////////////////////////////////////////////////////////////////////////
Set  filInc = Include("cmm/fil.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Incluir funciones de ficheros.",filInc);
////////////////////////////////////////////////////////////////////////////
```

Set dirInc

```
////////////////////////////////////////////////////////////////////////////
Set  dirInc = Include("cmm/dir.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Incluir funciones de directorios.",dirInc);
////////////////////////////////////////////////////////////////////////////
```

Set tmeInc

```
////////////////////////////////////////////////////////////////////////////
Set  tmeInc = Include("cmm/tme.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Inclusion del macro-expansor simple de Tol en Html.",tmeInc);
////////////////////////////////////////////////////////////////////////////
```

Set ftpInc

```
////////////////////////////////////////////////////////////////////////////
Set  ftpInc = Include("cmm/ftp.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Incluir funciones para generar mandatos Ftp.",ftpInc);
////////////////////////////////////////////////////////////////////////////
```

Set xsmInc

```
////////////////////////////////////////////////////////////////////////////
Set  xsmInc = Include("cmm/xsm.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Incluir funciones para construir sitemaps en Xml.",xsmInc);
////////////////////////////////////////////////////////////////////////////
```

Set pdbInc

```
////////////////////////////////////////////////////////////////////////////
Set  pdbInc = Include("app/pdb.tol"); // Posts database funtions
////////////////////////////////////////////////////////////////////////////
PutDescription("Funciones de manejo de los post de la agenda.",pdbInc);
////////////////////////////////////////////////////////////////////////////
```

Set keyInc

```
////////////////////////////////////////////////////////////////////////////
Set  keyInc = Include("app/key.tol"); // Keywords
////////////////////////////////////////////////////////////////////////////
PutDescription("Funciones generar palabras clave de páginas.",keyInc);
////////////////////////////////////////////////////////////////////////////
```