make.tol de con-Q.tv

Constructor de la Newsletter AVPPM con-Q.tv que genera tanto las páginas de la newsletter online, con el histórico de noticias, como el contenido de cada newsletter periódica, con las últimas noticias, que se envía a los socios por correo electrónico. Las noticias son posts con 4 niveles de importancia (status), que se organizan en un directorio con ficheros y estos con posts, que pueden pertenecer a múltiples clases o categorías. El programa esta basado en un macro-expansor a doble nivel de Tol en Html, que permite, por ejemplo, crear índices automáticos por artículos, por clases, por años que se tratan como clases, por niveles e índices de ilustraciones y por categorías de ilustraciones.

Los contenidos del sitio web con-Q.tv no son estáticos sino que cambian ligeramente en cada ejecución del programa lo que se consigue mediante textos alternativos cuando se redactan los posts. Este programa rellena automáticamente las descripciones y la lista de palabras clave de forma diferenciada para cada página como puede verse en los campos meta description y meta keywords de las páginas Html generadas. Las funciones principales de este programa son crear artículos, páginas de categorías, páginas de control de errores, el mapa del sitio web, crear los textos de la newsletter para ser enviados por email y poner todos los contenidos online.

Los posts pueden ser de las 4 siguientes niveles: a) anulados que no salen (que se determinan con la letra A), b) bajos con un tipo de letra pequeña en su clase y con artículo propio (con la letra B), c) comunes con un tipo de letra grande, fondo blanco, en su clase y con su articulo (con la letra C) y d) destacados con letra grande, fondo rosa, en su clase y en su artículo (con la letra D). 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 sus años de publicación.

Los posts pueden pertenecer a múltiples clases y se crean páginas Html de posts y de conjuntos de post por cada clase, a estas clases también se las denomina categorías. Con este programa las páginas de artículos se generan con la opción art, las páginas con los artículos agrupados por categorías con la opción cat que tambien crea los índices de clasificación de los artículos, las páginas Html por defecto del sitio (index.html) y de cada directorio, la absoluta y la de control de errores con la opción rot, el mapa del sitio en Xml con la opción xsm, de Xml site map y las opciones ftp, fup y snd permiten poner el contenido en online.

Este programa y la creación de este sitio web esta basado en un macro-expansor a doble nivel de Tol embebido en Html, donde la semilla de Html contiene Tol embebido y los post también pueden contener Tol embebido, por lo que dentro de la primera expansión, la de la semilla, se pueden realizar otras expansiones, que son las contenida en los post. La macro-espansión de Tol a doble nivel que se emplea en este programa permite que los post contengan código Html y código Tol, por ejemplo, para crear índices, realizar cálculos, poner diferentes contenidos para una misma página en diferentes ejecuciones, etc.

Los comentarios del código de este programa están realizados utilizando unas veces el español, sin acentos dentro del código, 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 por ftp. Este control lo realiza la opción fup, siglas de ftp update, frente a la opción ftp que genera ficheros de mandatos de envío con todo el contenido del web. Nótese que las opciones fup y ftp sólo generan ficheros de mandatos ftp de envío y luego se pueden ejecutar con la opción snd.

Árbol de ficheros

- con-Q.tv construye las páginas del sitio y newsletter Con-Q.tv
  - ← make.tol proceso principal de generacion del sitio web y la newsletter
  - tol directorios de código Tol
    - cmm funciones comunes
      - ← txt.tol funciones de textos
      - ← set.tol funciones de conjuntos
      - ← fil.tol funciones de ficheros
      - ← dir.tol funciones directorios
      - ← tme.tol funciones del macro-expansor de Tol en Html
      - ← img.tol funciones de imagenes Html
      - ← ftp.tol para generar mandatos para hacer Ftp
      - ← xsm.tol para construir sitemaps en Xml
    - app funciones especificas de aplicacion
      - ← ads.tol funciones de anuncios de publicidad
      - ← pdb.tol funciones de manejo de los posts de una agenda
      - ← pht.tol funciones auxiliares para el Html de los post
    - ← inc.tol para la inclusión de ficheros Tol
  - agenda directorio destinado a la agenda de post
    - ← 2006.revista.ed.age conjunto de posts de contenido para publicar
  - web directorio destinado a las páginas web generadas
    - css directorio para ficheros de estilo
      - ← common.css fichero de estilo Css para las páginas Html
    - seed directorio para ficheros semilla
      - ← seed.htm semilla de página Html con Tol embebido
      - ← lopd.htm texto Lopd para insertar en la newsletter de envío por email
    - src directorio para ficheros con código Javascript
      - ← common.js funciones Javascript para diferentes redes sociales

noticia directorio para páginas de noticias

→ 20070328_promociones.html ejemplo de página Html de artículo generada

quiosco directorio para páginas Html de categorias

→ asamblea.html ejemplo de página de categoría generada automáticamente

→ sitemap.xml mapa completo del sitio web generado en Xml automáticamente

→ presentacion.html vídeo de presentación de la Newsletter de la Avppm

→ con-q_tv.pdf documento resumen de funciones del programa de la newsletter

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.
- `Real CtrPxW`
  Number of posts per whole posts page.
- `Real CtrKxP`
  Number of keywords per page.
- `Real CtrPxC`
  Number of posts per category page.
- `Real CtrPxN`
  Number of posts per newsletter/notice page.
- `Real CtrPxA`
  Number of posts per article page.
- `Real CtrPxD`
  Number of posts per description.
- `Real CtrCxD`
  Characters per description, LinkedIn uses 250 and Facebook uses 300.
- `Real CtrPxT`
  Number of posts per title.
- `Real CtrAxP`
  Advertisements per page ratio.
- `Set CatInx`
  Categorias para el indice.

- `Set CatYea`
  Categorias por años.
- `Set CatNot`
  Categorias no indexables.
- `Set CatPag`
  Categorias de las paginas.
- `Set CatAll`
  Hay paginas para todas las categorias.

Proceso

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

Read agenda

- `Set CtrNoA`
  Read all posts minus A(nulated).
- `Set CtrPdb`
  Conjunto de posts Basicos, Comunes y Destacados.
- `Set CtrCmn`
  Conjunto de posts Comunes y Destacados.
- `Set CtrDst`
  Conjunto de posts Destacados.
- `Real makCat`
  Build the category pages for CatPag set.
- `Real makArt`
  Build all articles pages.
- `Real makRot`
  Build root absolute page.
- `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.

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   // Posts database
{
  Set  pstCla, // Conjunto de tipos de post
  Text pstSta, // Status A(nulado), B(ajo), C(comun), D(estacado)
  Text pstFil, // Camino del fichero (path)
  Text pstTit, // Titulo del post sin Html
  Text pstHtm, // Texto del post en Html
  Text pstRes, // Texto resumen del post en Html
  Text pstTxt  // Texto del post en Ascii puro
};
```

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

Real CtrPxW

```
////////////////////////////////////////////////////////////////////////////
Real CtrPxW =  64;
////////////////////////////////////////////////////////////////////////////
PutDescription("Number of posts per whole posts page.", CtrPxW);
////////////////////////////////////////////////////////////////////////////
```

Real CtrKxP

```
////////////////////////////////////////////////////////////////////////////
Real CtrKxP =  32;
////////////////////////////////////////////////////////////////////////////
PutDescription("Number of keywords per page.", CtrKxP);
////////////////////////////////////////////////////////////////////////////
```

Real CtrPxC

```
////////////////////////////////////////////////////////////////////////////
Real CtrPxC =  16;
////////////////////////////////////////////////////////////////////////////
PutDescription("Number of posts per category page.", CtrPxC);
////////////////////////////////////////////////////////////////////////////
```

Real CtrPxN

```
////////////////////////////////////////////////////////////////////////////
Real CtrPxN =   4;
////////////////////////////////////////////////////////////////////////////
PutDescription("Number of posts per newsletter/notice page.", CtrPxN);
////////////////////////////////////////////////////////////////////////////
```

Real CtrPxA

```
////////////////////////////////////////////////////////////////////////////
Real CtrPxA =   4;
////////////////////////////////////////////////////////////////////////////
PutDescription("Number of posts per article page.", CtrPxA);
////////////////////////////////////////////////////////////////////////////
```

Real CtrPxD

```
////////////////////////////////////////////////////////////////////////////
Real CtrPxD =   6;
////////////////////////////////////////////////////////////////////////////
PutDescription("Number of posts per description.", CtrPxD);
////////////////////////////////////////////////////////////////////////////
```

Real CtrCxD

```
////////////////////////////////////////////////////////////////////////////
Real CtrCxD = 384;
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Characters per description, LinkedIn uses 250 and Facebook uses 300.",
CtrCxD);
////////////////////////////////////////////////////////////////////////////
```

Real CtrPxT

```
////////////////////////////////////////////////////////////////////////////
Real CtrPxT =   2;
////////////////////////////////////////////////////////////////////////////
PutDescription("Number of posts per title.", CtrPxT);
////////////////////////////////////////////////////////////////////////////
```

Real CtrAxP

```
////////////////////////////////////////////////////////////////////////////
Real CtrAxP = .00; // Lo anulo antes de lanzar newsletter luego pondre 0.50
```

```
////////////////////////////////////////////////////////////////////////////
PutDescription("Advertisements per page ratio.", CtrAxP);
////////////////////////////////////////////////////////////////////////////
```

Set CatInx

```
////////////////////////////////////////////////////////////////////////////
Set CatInx =
[[
  "ANEI",
  "Asamblea",
  "Ayuntamiento de Madrid",
  "Cámara de Comercio",
  "CEIM CEOE",
  "Cliente",
  "CNC",
  "Consultoría",
  "Comunidad de Madrid",
  "Consumidor",
  "Desarrollo de negocio",
  "Distribuidor",
  "Economía",
  "Editor",
  "Elección",
  "Historia",
  "Internet",
  "Junta directiva",
  "Lector",
  "Legal",
  "Libro",
  "Madrid Emprende",
  "Ministerio de Industria",
  "Nota de prensa",
  "Prensa",
  "Publicidad",
  "Punto de conveniencia";
  "Qred",
  "Quiosco",
  "Quiosquero",
  "Red.es";
  "Revista",
  "Servicio de pago centralizado",
  "Tecnología",
  "Verano",
  "WiFi"
]];
////////////////////////////////////////////////////////////////////////////
PutDescription("Categorias para el indice.", CatInx);
////////////////////////////////////////////////////////////////////////////
```

Set CatYea

```
////////////////////////////////////////////////////////////////////////////
Set CatYea = PdbYearSet(CtrAge);
////////////////////////////////////////////////////////////////////////////
PutDescription("Categorias por años.", CatYea);
////////////////////////////////////////////////////////////////////////////
```

Set CatNot

```
////////////////////////////////////////////////////////////////////////////
Set CatNot =
[[
  "Advertencia legal", // Todas son bajas
  "Ilustración",       // Listas de ilustraciones
```

```
  "Mapa",               // Indice de articulos
  "Noticia",            // Son casi todos los articulos y repite el indice
  "Post"                // Todos son post, pues se mete automaticamente
]];
////////////////////////////////////////////////////////////////////////////
PutDescription("Categorias no indexables.", CatNot);
////////////////////////////////////////////////////////////////////////////
```

Set CatPag

```
////////////////////////////////////////////////////////////////////////////
Set CatPag = CatNot << CatYea << CatInx;
////////////////////////////////////////////////////////////////////////////
PutDescription("Categorias de las paginas.", CatPag);
////////////////////////////////////////////////////////////////////////////
```

Set CatAll

```
////////////////////////////////////////////////////////////////////////////
Set CatAll = CatPag;
////////////////////////////////////////////////////////////////////////////
PutDescription("Hay paginas para 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 con-Q.tv site
  OPTION
    cat: build all category pages
    art: build all articles pages
    rot: build root absolute page
    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
    all: do all works: cat, art, rot, xsm, fup, snd
    hlp: show this help
    tst: test some functions");
  TRUE
});
////////////////////////////////////////////////////////////////////////////
PutDescription("Es cierto si se ha visualizado la ayuda.", makHlp);
////////////////////////////////////////////////////////////////////////////
```

Set CtrNoA

```
////////////////////////////////////////////////////////////////////////////
Set  CtrNoA = If(! (ctrExe <: [["all", "cat", "art"]]), Empty,
                 PdbRead(CtrAge));
////////////////////////////////////////////////////////////////////////////
PutDescription("Read all posts minus A(nulated).", CtrNoA);
////////////////////////////////////////////////////////////////////////////
```

Set CtrPdb

```
////////////////////////////////////////////////////////////////////////////
Set  CtrPdb = Select(CtrNoA, Real(Set a) { a->pstSta <: [["B","C","D"]] });
////////////////////////////////////////////////////////////////////////////
PutDescription("Conjunto de posts Basicos, Comunes y Destacados.", CtrPdb);
////////////////////////////////////////////////////////////////////////////
Text WriteLn("Status _BCD "+F(Card(CtrPdb))+" registers");
```

Set CtrCmn

```
////////////////////////////////////////////////////////////////////////////
Set  CtrCmn = Select(CtrPdb, Real(Set a) { a->pstSta <: [["C","D"]] });
////////////////////////////////////////////////////////////////////////////
PutDescription("Conjunto de posts Comunes y Destacados.", CtrCmn);
////////////////////////////////////////////////////////////////////////////
Text WriteLn("Status __CD "+F(Card(CtrCmn))+" registers");
```

Set CtrDst

```
////////////////////////////////////////////////////////////////////////////
Set  CtrDst = Select(CtrCmn, Real(Set a) { a->pstSta == "D" }); // Destacado
////////////////////////////////////////////////////////////////////////////
PutDescription("Conjunto de posts Destacados.", CtrDst);
////////////////////////////////////////////////////////////////////////////
Text WriteLn("Status ___D "+F(Card(CtrDst))+" registers");
```

Real makCat

```
////////////////////////////////////////////////////////////////////////////
Real makCat = If(And(ctrExe!="all",ctrExe!="cat"), FALSE,
{
  Text WriteLn(makSep+"building all category pages...");

  Real CtrArt = FALSE; // Categories, not articles

  // Make category pages
  Set  cicCat = EvalSet(CatPag, Real(Text CtrPag)
  {
    Text CtrDif = CtrPag; // Diferencia clases de artículos similares
    Text CtrFil = PhtCleanFilePath("web/quiosco", CtrPag, "html"); // Output
    Text Write(CtrFil+";"+CtrPag);

    Real maxPst = Case(CtrPag=="Noticia", CtrPxN,  // Pocos
                       CtrPag=="Post",    CtrPxW,  // Muchos, revisar en Word
                       TRUE,              CtrPxC); // Ni pocos ni muchos

    Set  SelPdb = // Selected post from database for this page
    {
      Set selSet = SetFirstN(Select(CtrPdb, Real(Set objPst) // B, C y D
```

```
                                        { CtrPag <: objPst->pstCla }), maxPst);
      Real selCrd = Card(selSet);
      Text WriteLn(";"+F(selCrd)+";posts selected");

      selSet
    };
    TmeFile("web/seed/seed.htm", CtrFil)
  });

  Card(cicCat)
});
////////////////////////////////////////////////////////////////////////////
PutDescription("Build the category pages for CatPag set.", makCat);
////////////////////////////////////////////////////////////////////////////
```

Real makArt

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

  Real CtrArt = TRUE; // Articles

  // Make article pages
  Set  cicArt = For(1, Card(CtrCmn), Real(Real pstNum) // Only C y D
  {
    Set  SelPdb = SetSubCicle(CtrCmn, pstNum, CtrPxA);
    Set  pstFst = CtrCmn[pstNum]; // The very first
    Text CtrPag = pstFst->pstTit;
    Text CtrDif = PhtSpanishDate(pstFst);
    Text CtrFil = "web/"+pstFst->pstFil; // Output
    Text WriteLn(CtrDif+"|"+CtrFil+"\n"+Repeat(" ",10)+"|"+CtrPag);
    TmeFile("web/seed/seed.htm", CtrFil)
  });

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

Real makRot

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

  Text inxHtm = ReadFile("web/quiosco/noticia.html");

  // Texto Lopd que se inserta en la pagina newsletter de envio por email
  Text lpdHtm = ReadFile("web/seed/lopd.htm");

  Text cssSrc = ReadFile("web/css/common.css");
  Text cssOld =
    "<link href='../css/common.css' rel='stylesheet' type='text/css' />";
  Text cmmNew =
    "<style type='text/css'>\n"    +
    Replace(cssSrc, " url('../", " url('http://www.con-q.tv/") +
    "</style>";

  Text absSav = ReplaceTable(inxHtm, // Save external references
  [[
     [[ " src=\"http:",     " src=\"Http:"]], // H grande evita cambios
     [[" href=\"http:",    " href=\"Http:"]]  // H grande evita cambios
```

```
  ]],1);

  Text absUrl = "\"http://www.con-q.tv/";
  Text absRep = ReplaceTable(absSav,
  [[
     [[ " src=\"../",    " src="+absUrl]],
     [[" href=\"../",    " href="+absUrl]],
     [["<!--LPD-->", lpdHtm]],
     [[      cssOld, cmmNew]]
  ]],1);
  Text absErr = Replace(absRep, "<h1>", "<h1 class='Red'>");

  Real FilWriteIfDiff("web/index.html",          absRep);
  Real FilWriteIfDiff("web/css/index.html",      absRep);
  Real FilWriteIfDiff("web/foto/index.html",     absRep);
  Real FilWriteIfDiff("web/noticia/index.html", absRep);
  Real FilWriteIfDiff("web/quiosco/index.html", absRep);
  Real FilWriteIfDiff("web/seed/index.htm",      absRep); // No en sitemap.xml

  Real FilWriteIfDiff("web/error404.html",       absErr);

  Text newCls = PhtCleanHtml(absRep);
  Real FilWriteIfDiff("web/newsletter.html", newCls); // Absolute + cleaned
  Real FilWriteIfDiff("web/newslocal.html",           // Relativa + cleaned
       Replace(PhtCleanHtml(absRep), " src="+absUrl, " src=\""));

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

Real makXsm

```
////////////////////////////////////////////////////////////////////////////
Real makXsm = If(And(ctrExe!="all",ctrExe!="xsm"), FALSE,
{
  Text WriteLn(makSep+"building xml site map...");
  XsmDir("web/sitemap.xml", "web", "http://www.con-q.tv/")
});
////////////////////////////////////////////////////////////////////////////
PutDescription("Build Xml site map.", makXsm);
////////////////////////////////////////////////////////////////////////////
```

Real makFtp

```
////////////////////////////////////////////////////////////////////////////
Real makFtp = If(!(ctrExe <: [["ftp","fup","all"]]), 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.con-q.tv",             // Host remoto
    locPth,                     // Directorio local
    dtePth,                     // Fichero señal de fecha de actualizacion
    [[//dir        extension    type    binary or ascii
      ["",         "html",     "html",   FALSE]],  // Ascii Html
      ["",         "ico",      "ico",    TRUE]],   // Binary favicon.ico
      ["",         "xml",      "xml",    FALSE]],  // Ascii site map, rss
      ["",         "pdf",      "pdf",    TRUE]],   // Documentos Pdf
      ["src",      "js",       "js",     FALSE]],  // Javascript
      ["css",      "css",      "css",    FALSE]],  // Css
```

```
      [["css",     "png",      "png",   TRUE]],   // Png solo de css
      [["",        "jpg",      "jpg",   TRUE]]    // Jpeg de css y foto
    ]])
});
////////////////////////////////////////////////////////////////////////
PutDescription(
"Crear los ficheros de mandatos completos ftp o de actualizacion fup.",
makFtp);
////////////////////////////////////////////////////////////////////////
```

Real makSnd

```
////////////////////////////////////////////////////////////////////////
Real makSnd = If(And(ctrExe!="all",ctrExe!="snd"), FALSE,
{
  Text WriteLn(makSep+"sending files using ftp...");
  System(W("ftp/con-Q.tv.bat"))
});
////////////////////////////////////////////////////////////////////////
PutDescription("Send files via ftp.", makSnd);
////////////////////////////////////////////////////////////////////////
```

Real makTst

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

  Real numCic = 2000;
  Set  letCic = For(1, numCic, Text(Real pos)
  { SetGetRand([["a","b","c","d"]]) });
  Set letCla = Classify(letCic, Real(Text a, Text b) { Compare(a, b) });
  Set letTst = EvalSet(letCla, Real(Set letSet)
  {
    Text WriteLn(letSet[1]+" "+FormatReal(Card(letSet),"%.0lf")
                           +" "+FormatReal(Card(letSet)/numCic,"%.2lf"));
    Card(letSet)
  });

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

txt.tol de con-Q.tv

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.
- `Text TxtReplaceFirst(Text txtInp, Text txtOld, Text txtNew)`
  Returns the result of change, only the very first time, in the input text txtInp the old text txtOld by the new text txtNew.
- `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 TxtInside2TagPlus(Text txtInp, Text tagIni, Text tagEnd)`
  Returns all the texts between 2 tags plus theis tags (tagIni and tagEnd) in txtInp. Is diferente that the classic TxtInside2Tag(). For example: <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.
- `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.

- `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().

- `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 TxtTwitterWrap(Text txtCut, Text txtNot)`
  Retorna un texto que no supere el maximo de caracteres de Twitter a partir de un texto inicial que se puede cortar y uno final que no se puede.

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

Text TxtReplaceFirst()

```
////////////////////////////////////////////////////////////////////////////
Text TxtReplaceFirst(Text txtInp, // Input text
                     Text txtOld, // Old text
                     Text txtNew) // New text
////////////////////////////////////////////////////////////////////////////
{
  Real posIni = TextFind(txtInp, txtOld);
  Text result = If(LE(posIni,0), txtInp, // No aparece, no se cambia nada
  {
    Real lenIni = TextLength(txtOld);
    Real posSub = posIni + lenIni;
    Real posEnd = TextLength(txtInp);
    Sub(txtInp, 1, posIni-1) + txtNew + Sub(txtInp, posSub, posEnd)
  })
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns the result of change, only the very first time, in the input text
txtInp the old text txtOld by the new text txtNew.",
TxtReplaceFirst);
////////////////////////////////////////////////////////////////////////////
```

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 TxtInside2TagPlus()

```
////////////////////////////////////////////////////////////////////////////
Text TxtInside2TagPlus(Text txtInp, // Input text
                       Text tagIni, // Initial tag
                       Text tagEnd) // End tag)
////////////////////////////////////////////////////////////////////////////
{
  Real posIni = TextFind(txtInp, tagIni);
  If(LE(posIni,0), "",
  {
    Real lenIni = TextLength(tagIni);
    Real posSub = posIni + lenIni;
    Real posEnd = TextFind(txtInp, tagEnd, posSub);
    Real txtLen = TextLength(txtInp);

    If(And(EQ(posIni,1),LE(posEnd,0)), txtInp,
    If(And(GT(posIni,1),LE(posEnd,0)), Sub(txtInp,posIni, txtLen),
       Sub(txtInp,posIni,posEnd+TextLength(tagEnd)-1)+ // Recursion
       TxtInside2TagPlus(Sub(txtInp, posEnd+TextLength(tagEnd), txtLen),
                         tagIni, tagEnd)))
  })
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns all the texts between 2 tags plus theis tags (tagIni and tagEnd)
in txtInp.
Is diferente that the classic TxtInside2Tag().
For example: <aaa(::)bbb(---)ccc>, <(>, <)> -> <(::)(---)>.",
TxtInside2TagPlus);
////////////////////////////////////////////////////////////////////////////
```

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

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

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 TxtTwitterWrap()

```
////////////////////////////////////////////////////////////////////////////
Text TxtTwitterWrap(Text txtCut, // Texto inicial cortable
                    Text txtNot) // Texto final no cortable
////////////////////////////////////////////////////////////////////////////
{
  Real chrTwi = 117;  // Number of characters for Twitter, teoricamente 140

  Set  setTwi = TxtLineWrap(txtCut, chrTwi-TextLength(txtNot), TRUE);

  Compact(setTwi[1]+txtNot)
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un texto que no supere el maximo de caracteres de Twitter a partir
de un texto inicial que se puede cortar y uno final que no se puede.",
TxtTwitterWrap);
```

set.tol de con-Q.tv

Set functions. This code assumes the following behaviour of: The function BinGroup():

Text WriteLn("<"+BinGroup("+",Empty)+">" -> <>

Text WriteLn("<"+BinGroup("+",[["a"]])+">"); -> <a>

Text WriteLn("<"+BinGroup("+",[["a","b"]])+">"); -> <ab>

Text WriteLn("<"+BinGroup("+",[["a","a"]])+">"); -> <aa>

Text WriteLn("<"+BinGroup("+",[["a","b","a"]])+">"); -> <aba>

The function SetSum():

Text WriteLn("<"+SetSum(Empty)+">"); -> <>

Text WriteLn("<"+SetSum([["a"]])+">"); -> <a>

Text WriteLn("<"+SetSum([["a","b"]])+">"); -> <ab>

Text WriteLn("<"+SetSum([["a","a"]])+">"); -> <aa>

Text WriteLn("<"+SetSum([["a","b","a"]])+">"); -> <aba>

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.
- `Text Set2TxtKeyword(Set valSet, Real minChr, Real a2zOrd, Real maxKey)`
  Returns a text list with all the elements of valSet converted in a text format with commas like a keywords list, ordered and without repetitions. Remove all word with LE(TextLength(), minChr). Returns the maxKey elements more occurrencies.
- `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.
- `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]
- `Anything SetGetRand(Set setInp)`
  Retorna un elemento al azar del conjunto de entrada setInp. Si setInp es Empty retorna FALSE.

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

Text Set2TxtKeyword()

```
////////////////////////////////////////////////////////////////////////////
Text Set2TxtKeyword(Set  valSet, // Set of texts
                    Real minChr, // Minimum number of chars
                    Real a2zOrd, // If true ordered
                    Real maxKey) // Maximun number of keywords
////////////////////////////////////////////////////////////////////////////
{
  Set cmmWrd =
  [[
     "además",
     "ambos", "ambas",
     "aunque",
     "cualquier", "cualquiera",
     "cuando",
     "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"
  ]];

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

  Text htmNot = TxtOutHtmScr(lstTxt);

  Text lstCls = Compact(ReplaceTable(htmNot,
    [[ [[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
  Set  setSel = Select(setTxt, Real(Text a) { GT(TextLength(a), minChr) });

// 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
  Set  keySor = If(! a2zOrd, keyFst,
                   Sort(keyFst, Real(Text a, Text b)
                     { Compare(ToLower(a),ToLower(b)) })); // Order

// Convert to text
  Set2Txt(keySor, "", "", ", ", ", ", "", "", "", "")
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns a text list with all the elements of valSet converted in a
text format with commas like a keywords list, ordered and without repetitions.
Remove all word with LE(TextLength(), minChr).
Returns the maxKey elements more occurrencies.",
Set2TxtKeyword);
////////////////////////////////////////////////////////////////////////////
```

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

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

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

fil.tol de con-Q.tv

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

dir.tol de con-Q.tv

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 con-Q.tv

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

img.tol de con-Q.tv

Image functions.

Declaraciones

Constantes

- `Text ImgTag`
  Tag inicial para imagenes.

Funciones

- `Set ImgGetSet(Text htmCod)`
  Retorna el conjunto de imagenes de una pieza de codigo Html.
- `Text ImgNormalize(Text imgCod)`
  Retorna el codigo Html de una imagen normalizado.
- `Text ImgGetSrc(Text imgCod)`
  Retorna de un codigo Html de una imagen la url de la imagen.
- `Text ImgGetAlt(Text imgCod)`
  Retorna de un codigo Html de una imagen el texto alt de la imagen.
- `Text ImgGetTit(Text imgCod)`
  Retorna de un codigo Html de una imagen el texto title de la imagen.
- `Text Img2Htm(Text imgSrc, Text imgAlt, Text imgTit, Text imgCla, Text imgDef)`
  Retorna el codigo Html de una imagen con sus atributos completos rellenando aquellos que faltan.
- `Text Img2Div(Text imgSrc, Text imgAlt, Text imgTit, Text imgLnk, Text imgLst, Text imgCla, Text imgDef)`
  Retorna el codigo Html de una imagen con sus atributos completos rellenando aquellos que faltan y todo dentro de una estructura Html div.
- `Text ImgMap(Set ctrPdb, Text txtPat, Text namCla, Text txtDef)`
  Retorna el codigo Html de un mapa o indice de imagenes extraidas de un conjunto de post en Html.

Constantes

Text ImgTag

```
////////////////////////////////////////////////////////////////////////
Text ImgTag = "<img ";
////////////////////////////////////////////////////////////////////////
PutDescription("Tag inicial para imagenes.", ImgTag);
////////////////////////////////////////////////////////////////////////
```

Set ImgGetSet()

```
////////////////////////////////////////////////////////////////////////
Set ImgGetSet(Text htmCod)
////////////////////////////////////////////////////////////////////////
{
  Text imgTxt = Compact(TxtInside2TagPlus(htmCod, ImgTag, ">"));
  If(imgTxt == "", Empty,
  {
    Text imgRep = Replace(imgTxt, "><", ">"+Char(7)+"<");
    Tokenizer(imgRep, Char(7))
  })
};
////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna el conjunto de imagenes de una pieza de codigo Html.",
ImgGetSet);
////////////////////////////////////////////////////////////////////////
```

Text ImgNormalize()

```
////////////////////////////////////////////////////////////////////////
Text ImgNormalize(Text imgCod)
////////////////////////////////////////////////////////////////////////
{
  Text imgCmp = Compact(imgCod);
  ReplaceTable(imgCmp,
  [[
    [["= ",      "="]],
    [[" =",      "="]],
    [[". '",     "'"]], // Elimina los puntos finales
    [[Char(34), "'"]]
  ]])
};
////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna el codigo Html de una imagen normalizado.",
ImgNormalize);
////////////////////////////////////////////////////////////////////////
```

Text ImgGetSrc()

```
////////////////////////////////////////////////////////////////////////
Text ImgGetSrc(Text imgCod)
////////////////////////////////////////////////////////////////////////
{ TxtBetween2Tag(ImgNormalize(imgCod), "src='", "'", TRUE) };
////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna de un codigo Html de una imagen la url de la imagen.",
ImgGetSrc);
////////////////////////////////////////////////////////////////////////
```

Text ImgGetAlt()

```
////////////////////////////////////////////////////////////////////////
Text ImgGetAlt(Text imgCod)
////////////////////////////////////////////////////////////////////////
{ TxtBetween2Tag(ImgNormalize(imgCod), "alt='", "'", TRUE) };
////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna de un codigo Html de una imagen el texto alt de la imagen.",
ImgGetAlt);
////////////////////////////////////////////////////////////////////////
```

Text ImgGetTit()

```
////////////////////////////////////////////////////////////////////////
Text ImgGetTit(Text imgCod)
////////////////////////////////////////////////////////////////////////
{ TxtBetween2Tag(ImgNormalize(imgCod), "title='", "'", TRUE) };
////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna de un codigo Html de una imagen el texto title de la imagen.",
ImgGetTit);
////////////////////////////////////////////////////////////////////////
```

Text Img2Htm()

```
////////////////////////////////////////////////////////////////////////
Text Img2Htm(Text imgSrc, // Image path
             Text imgAlt, // Alternate text, mandatory
             Text imgTit, // Image title
             Text imgCla, // Image class
             Text imgDef) // Para cuando no hay textos
////////////////////////////////////////////////////////////////////////
{
  Text htmSrc =   " src="+Q(imgSrc);

  Text htmCla = " class="+Q(imgCla);

  Text htmAlt = If(imgAlt!="", " alt="+Q(imgAlt),     // Hay alt
                If(imgTit!="", " alt="+Q(imgTit),     // Sin alt pero si title
                               " alt="+Q(imgDef)));   // Sin nada

  Text htmTit = If(imgTit!="", " title="+Q(imgTit),   // Hay title
                If(imgAlt!="", " title="+Q(imgAlt),   // Sin title pero si alt
                               " title="+Q(imgDef)); // Sin nada

  Text htmImg = "\n  <img" + htmSrc +
                "\n     "   + htmCla +
                "\n       " + htmAlt +
                "\n    "    + htmTit + " />";

  htmImg
};
////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna el codigo Html de una imagen con sus atributos completos rellenando
aquellos que faltan.",
Img2Htm);
////////////////////////////////////////////////////////////////////////
```

Text Img2Div()

```
////////////////////////////////////////////////////////////////////////
Text Img2Div(Text imgSrc, // Image path
             Text imgAlt, // Alternate text, mandatory
             Text imgTit, // Image title
             Text imgLnk, // Main link
             Text imgLst, // List of links
             Text imgCla, // Image class
             Text imgDef) // Para cuando no hay textos
////////////////////////////////////////////////////////////////////////
{
  Text htmImg = Img2Htm(imgSrc, imgAlt, imgTit, imgCla, imgDef);
```

```
  Text htmTxt = Case
  (
    And(imgAlt != "", imgTit == ""),              imgAlt,
    And(imgAlt == "", imgTit != ""),              imgTit,
    And(imgAlt == "", imgTit == ""),              imgDef,
        imgAlt == imgTit,                         imgAlt,
        TRUE,                          imgTit+": "+imgAlt
  )+". ";

  Text htmDes = "<p>"+htmTxt+imgLst+"</p>";

  "\n<div class="+Q(imgCla)+">" +
  "\n  "    + imgLnk + htmImg + "</a>" +
  "\n  "    + htmDes +
  "\n</div><div style="+Q("clear:both")+"></div>\n"
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna el codigo Html de una imagen con sus atributos completos rellenando
aquellos que faltan y todo dentro de una estructura Html div.",
Img2Div);
////////////////////////////////////////////////////////////////////////////
```

Text ImgMap()

```
////////////////////////////////////////////////////////////////////////////
Text ImgMap(Set  ctrPdb, // Set of post from a post database
            Text txtPat, // Text pattern for seleccion in alts and title
            Text namCla, // Name for image class
            Text txtDef) // Para cuando no hay textos
////////////////////////////////////////////////////////////////////////////
{
  Set  imgPdb = EvalSet(ctrPdb, Set(Set objPst)
  {// Si se crean imágenes dinámicas no las ve, sería recursivo en este post
    Set imgSel = ImgGetSet(objPst->pstHtm);

    EvalSet(imgSel, Set(Text imgCod)
    {
      Text imgSrc = ImgGetSrc(imgCod);
      Text imgAlt = ImgGetAlt(imgCod);
      Text imgTit = ImgGetTit(imgCod);
      Text imgLnk = PhtLinkPost(objPst);

      // Text WriteLn(imgSrc+"\n"+imgAlt+"\n"+imgTit+"\n"+imgLnk+"\n");
      SetOfText(imgSrc, imgAlt, imgTit, imgLnk)
    })
  });

  Set  imgGrp = BinGroup("<<", imgPdb);
  Text lowPat = Compact(ToLower(txtPat));
  Set  imgSel = If(lowPat == "", imgGrp, // Todas las imagenes
  {
    Select(imgGrp, Real(Set imgSet)
    { Or(TextFind(Compact(ToLower(imgSet[2])), lowPat),    // Busca en alt
         TextFind(Compact(ToLower(imgSet[3])), lowPat)) }) // Busca en title
  });

  // Clasifica por el path de la imagen reverse
  Set  imgCla = Classify(imgSel, Real(Set a, Set b) { Compare(b[1], a[1]) });

  Set  imgCic = EvalSet (imgCla, Text(Set imgSet)
  {
    Text imgSrc = imgSet[1][1];
    Text imgAlt = imgSet[1][2];

    // Solo si title es diferente al alt
    Text imgTit = If(imgAlt == imgSet[1][3], "", imgSet[1][3]);

    // El primer link
    Text imgLnk = "<"+TxtBetween2Tag(imgSet[1][4], "<", ">", TRUE)+">";
```

```
    Text imgLst = Set2Txt(Traspose(imgSet)[4],
      "En ", ".", ", ", " y ", "", "", "", ""); // Todos los links

    Img2Div(imgSrc, imgAlt, imgTit, imgLnk, imgLst, namCla, txtDef)
  });
  SetSum(imgCic)
};
////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna el codigo Html de un mapa o indice de imagenes extraidas de un
conjunto de post en Html.",
ImgMap);
////////////////////////////////////////////////////////////////////////
```

ftp.tol de con-Q.tv

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 con-Q.tv

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

ads.tol de con-Q.tv

Adverstiment functions.

Declaraciones

Constantes

- `Text AdsUrl`
  Url for javascript ads code.
- `Text AdsCla`
  Div class for ads code.
- `Text AdsHtm`
  Html call code for ads.

Funciones

- `Text AdsRand(Real adsRat)`
  Con una probabilidad de adsRat retorna el texto de un anuncio y con una probabilidad de 1-adsRat retorna el texto vacio. Por ejemplo, AdsRand(0.25) retorna un anuncio en 1 de cada 4 llamadas.

Constantes

Text AdsUrl

```
////////////////////////////////////////////////////////////////////////////
Text AdsUrl = "http://www.edicionesacontracorriente.com/src/ads.js";
////////////////////////////////////////////////////////////////////////////
PutDescription("Url for javascript ads code.", AdsUrl);
////////////////////////////////////////////////////////////////////////////
```

Text AdsCla

```
////////////////////////////////////////////////////////////////////////////
Text AdsCla = "<div class='Ads'>";
////////////////////////////////////////////////////////////////////////////
PutDescription("Div class for ads code.", AdsCla);
////////////////////////////////////////////////////////////////////////////
```

Text AdsHtm

```
////////////////////////////////////////////////////////////////////////////
Text AdsHtm = "\n" + AdsCla + "
  <script
    type='text/javascript'
     src="+Q(AdsUrl)+">
```

```
  </script>
  <script type='text/javascript'>
    AdsBan('',700,'');
  </script>
</div>
";
////////////////////////////////////////////////////////////////////////////
PutDescription("Html call code for ads.", AdsHtm);
////////////////////////////////////////////////////////////////////////////
```

Text AdsRand()

```
////////////////////////////////////////////////////////////////////////////
Text AdsRand(Real adsRat)
////////////////////////////////////////////////////////////////////////////
{ If(GE(Rand(0,1), adsRat), "", AdsHtm) };
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Con una probabilidad de adsRat retorna el texto de un anuncio y
con una probabilidad de 1-adsRat retorna el texto vacio.
Por ejemplo, AdsRand(0.25) retorna un anuncio en 1 de cada 4 llamadas.",
AdsRand);
////////////////////////////////////////////////////////////////////////////
```

pdb.tol de con-Q.tv

Posts database functions.

Declaraciones

Constantes

- `Text PdbSep`
  Post separator inside the agendas.

Funciones

- `Set PdbRead(Text inpDir)`
  Reads and returns a post database as a set of structures.
- `Set PdbFirstN(Set inpSet, Real maxNum, Code funSel)`
  Returns the maxNum recents posts selected using the function funSel.
- `Set PdbYearSet(Text inpDir)`
  Returns a set with all the years numbers, as texts, that have published posts in the agenda directory inpDir.

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 }; // Functio

  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
    Text pstSta = Sub(TxtBetween2Tag(inf, "<Pst.Sta>", "\n<Pst.", TRUE) +
                      "C",1,1); // C por defecto
```

```
    Text pstFil = TxtBetween2Tag(inf, "<Pst.Fil>", "\n<Pst.", TRUE);
    Text pstTit = TxtBetween2Tag(inf, "<Pst.Tit>", "\n<Pst.", TRUE);
    Text pstHtm = TxtBetween2Tag(inf, "<Pst.Txt>", "\n<Pst.", FALSE);

    Text pstRes = TxtOutside2Tag(pstHtm,"<!--INI-->","<!--END-->"); // Resumen

    Set  pstCla =
      Txt2Set(TxtBetween2Tag(inf, "<Pst.Cla>", "\n<Pst.",  TRUE), ";") <<
      SetOfText("Post", Sub(pstFil, 1, 4)); // All are post + publication year

    Text pstTxt = TxtOutHtmScr(pstHtm);

    // Camino completo y limpio
    Text pstPth = PhtCleanFilePath("noticia", pstFil, "html");

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

    // Store
    Set  pstObj = PdbSt(pstCla,
                        pstSta,
                        pstPth,
                        pstTit,
                        pstHtm,
                        pstRes,
                        pstTxt);
    pstObj
  });

  Set inpSel = Select(inpTab, Real(Set a) { a->pstSta!="A" }); // Not deleted

  // Check
  Real chkDup =
  {
    Set codCla = Classify(inpTab, Real(Set a, Set b)
                 { Compare(a->pstFil, b->pstFil) });
    Set filDup = EvalSet(codCla, Real(Set cla)
    {
      Real crd = Card(cla);
      Text nam = cla[1]->pstFil;
      If(EQ(crd,1), TRUE, err("Name: "+nam+" "+F(crd)+" times"))
    });
    Card(filDup)
  };

  Text WriteLn("Status ABCD "+F(Card(inpTab))+" registers");
  Text WriteLn("Status _BCD "+F(Card(inpSel))+" registers");

  Set inpSor = Sort(inpSel, Real(Set a, Set b) // Reverse file order
               { Compare(b->pstFil, a->pstFil) });

  inpSor // Los ficheros van con fecha, los mas modernos primero
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Reads and returns a post database as a set of structures.",
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);
////////////////////////////////////////////////////////////////////////////
```

Set PdbYearSet()

```
////////////////////////////////////////////////////////////////////////////
Set PdbYearSet(Text inpDir)
////////////////////////////////////////////////////////////////////////////
{
  Text WriteLn("Reading "+inpDir+" years...");
  Set  pthSet = DirExtAll(inpDir, "age", FALSE, TRUE);

  Set  yeaSet = EvalSet(pthSet, Text(Text pthFil)
  { TxtBetween2Tag(pthFil, inpDir+"/", ".", TRUE) }); // agenda/2013.encue...

  // Agrupa años iguales y ordena de más reciente a más antiguo
  Set  yeaCla = Classify(yeaSet, Real(Text a, Text b) { Compare(b,a) });

  EvalSet(yeaCla, Text(Set claSet) { claSet[1] }) // Uno por clase
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Returns a set with all the years numbers, as texts, that have published
posts in the agenda directory inpDir.",
PdbYearSet);
////////////////////////////////////////////////////////////////////////////
```

pht.tol de con-Q.tv

Posts Html functions.

Declaraciones

Funciones

- `Text PhtExpText(Text txtHtm, Real ctrHtm)`
  Expande pequeños textos muy habituales dentro de los posts.
- `Text PhtCleanFilePath(Text dirNam, Text filNam, Text filExt)`
  Retorna limpia la ruta de un fichero eliminado los acentos y cambiando por _ todos los caracteres que no son basicos.
- `Text PhtLinkClass(Text claNam)`
  Retorna el link de enlace a una pagina de clases.
- `Text PhtLinkPost(Set pstObj)`
  Retorna el link de enlace a un post.
- `Text PhtLinkClassSet(Set setCla)`
  Retorna una lista de link de enlaces a un conjunto de clases, a modo de indice.
- `Text PhtLinkPostSet(Set setPst)`
  Retorna una lista de link de enlaces a un conjunto de post, a modo de indice.
- `Text PhtQuote(Text resTxt, Real opeQuo, Real common)`
  Retorna el codigo html con las imagenes de las comillas de apertura y cierre y el resumen del post.
- `Text PhtCleanHtml(Text htmCod)`
  Retorna un codigo limpio de referencias externas a internet, con el objetivo que al enviarse el codigo html por correo electronico el gestor de email no diga que para proteger la privacidad se han bloqueado elementos externos. Tambien elimina los iframes de videos que no se visualizan por email, al hacerlo deja las imagenes insertadas entre el <iframe y el </iframe> que muestran la portada del video. Tambien elimina los anuncios publicitarios.
- `Text PhtSpanishDate(Set objPst)`
  Retorna una fecha en formato español, aaaa/mm/dd, al que estan acostumbrados los lectores de la newsletter.

Text PhtExpText()

```
////////////////////////////////////////////////////////////////////////////
Text PhtExpText(Text txtHtm, // Expande este texto
                Real ctrHtm) // Si true con html si false sin html
////////////////////////////////////////////////////////////////////////////
{
  Text outHtm = ReplaceTable(txtHtm,
  [[
    [["_avp_", "<a href=" + Q("http://www.avppm.es")  + ">AVPPM</a>"]],
    [["_cnQ_", "<a href=" + Q("http://www.con-q.es")  + ">con-Q</a>"]],
    [["_Qre_", "<a href=" + Q("http://www.qred.es")   + ">Qred</a>"]],
    [["_new_", "<a href=" + Q("../newsletter.html")   + ">Newsletter</a>"]]
  ]], 1); // Solo 1 ciclo
```

```
  If(ctrHtm, outHtm, TxtOutHtmScr(outHtm))
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Expande pequeños textos muy habituales dentro de los posts.",
PhtExpText);
////////////////////////////////////////////////////////////////////////////
```

Text PhtCleanFilePath()

```
////////////////////////////////////////////////////////////////////////////
Text PhtCleanFilePath(Text dirNam, // Directorio
                      Text filNam, // Nombre del fichero sin extension
                      Text filExt) // Extension sin punto
////////////////////////////////////////////////////////////////////////////
{
  Text filLow = ToLower(filNam);
  Text filRep = ReplaceTable(filLow,
    [[
       [["á", "a"]], [["é", "e"]], [["í", "i"]], [["ó", "o"]], [["ú", "u"]],
       [["ñ", "n"]]
    ]]);

  Set  letCic = TxtForChr(filRep, Text(Text let)
  { If(Or(And(let>="0", let<="9"), And(let>="a", let<="z")), let, "_") });

  dirNam + "/" + SetSum(letCic) + "." + filExt
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna limpia la ruta de un fichero eliminado los acentos y cambiando por
_ todos los caracteres que no son basicos.",
PhtCleanFilePath);
////////////////////////////////////////////////////////////////////////////
```

Text PhtLinkClass()

```
////////////////////////////////////////////////////////////////////////////
Text PhtLinkClass(Text claNam)
////////////////////////////////////////////////////////////////////////////
{
  Text claPth = PhtCleanFilePath("../quiosco", claNam, "html");
  "<a href=" + Q(claPth) + ">" + claNam + "</a>"
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna el link de enlace a una pagina de clases.",
PhtLinkClass);
////////////////////////////////////////////////////////////////////////////
```

Text PhtLinkPost()

```
////////////////////////////////////////////////////////////////////////////
Text PhtLinkPost(Set pstObj)
////////////////////////////////////////////////////////////////////////////
{ "<a href=" + Q("../"+pstObj->pstFil) + ">" + pstObj->pstTit + "</a>" };
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna el link de enlace a un post.",
PhtLinkPost);
////////////////////////////////////////////////////////////////////////////
```

Text PhtLinkClassSet()

```
////////////////////////////////////////////////////////////////////////////
Text PhtLinkClassSet(Set  setCla) // Set of classes
////////////////////////////////////////////////////////////////////////////
{
  SetSum(EvalSet(setCla, Text(Text claNam)
  { "<li>" + PhtLinkClass(claNam) + ".</li>" }))
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna una lista de link de enlaces a un conjunto de clases,
a modo de indice.",
PhtLinkClassSet);
////////////////////////////////////////////////////////////////////////////
```

Text PhtLinkPostSet()

```
////////////////////////////////////////////////////////////////////////////
Text PhtLinkPostSet(Set  setPst) // Set of posts
////////////////////////////////////////////////////////////////////////////
{
  SetSum(EvalSet(setPst, Text(Set pstObj)
  { "<li>" + PhtLinkPost(pstObj) + ".</li>" }))
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna una lista de link de enlaces a un conjunto de post,
a modo de indice.",
PhtLinkPostSet);
////////////////////////////////////////////////////////////////////////////
```

Text PhtQuote()

```
////////////////////////////////////////////////////////////////////////////
Text PhtQuote(Text resTxt, // Resume text
              Real opeQuo, // Open quote, ese close
              Real common) // Normal color
////////////////////////////////////////////////////////////////////////////
{
  Text filQuo = Case
  (
    And( opeQuo,  common), "quoteopenred.png",
    And( opeQuo, !common), "quoteopenwhite.png",
    And(!opeQuo,  common), "quoteclosered.png",
    And(!opeQuo, !common), "quoteclosewhite.png",
    TRUE,                  "quoteopenred.png"
  );
  Text htmQuo = "<p><img src=" + Q("../css/"+filQuo) +
                        " alt=" + Q("Resumen de la noticia") + " />";
  TxtReplaceFirst(resTxt, "<p>", htmQuo)
};
////////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna el codigo html con las imagenes de las comillas de apertura y
cierre y el resumen del post.",
PhtQuote);
////////////////////////////////////////////////////////////////////////////
```

Text PhtCleanHtml()

```
////////////////////////////////////////////////////////////////////////////
Text PhtCleanHtml(Text htmCod) // Limpia para email, deja interior iframe
```

```
////////////////////////////////////////////////////////////////////////
{
  Text notDoc = TxtOutside2Tag(htmCod, "<!DOCTYPE",          ">");
  Text notXml = "<html>" +
                TxtOutside2Tag(notDoc, "<html",              ">");
  Text notBck = TxtOutside2Tag(notXml, "background-image:", ";");
  Text notEFr = Replace(       notBck, "</iframe>",          "");
  Text notIFr = TxtOutside2Tag(notEFr, "<iframe",           ">");
  Text notIco = TxtOutside2Tag(notIFr, "<link rel=",        ">");
  Text notScr = TxtOutside2Tag(notIco, "<script",           "</script>");
  Text notFrm = TxtOutside2Tag(notScr, "<form",             "</form>");
  Text notAds = TxtOutside2Tag(notFrm, AdsCla,             "</div>");
  notAds
};
////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna un codigo limpio de referencias externas a internet,
con el objetivo que al enviarse el codigo html por correo electronico
el gestor de email no diga que para proteger la privacidad se han bloqueado
elementos externos.
Tambien elimina los iframes de videos que no se visualizan por email,
al hacerlo deja las imagenes insertadas entre el <iframe y el </iframe> que
muestran la portada del video.
Tambien elimina los anuncios publicitarios.",
PhtCleanHtml);
////////////////////////////////////////////////////////////////////////
```

Text PhtSpanishDate()

```
////////////////////////////////////////////////////////////////////////
Text PhtSpanishDate(Set  objPst) // Posts
////////////////////////////////////////////////////////////////////////
{
  Text dteTxt = TxtBetween2Tag(objPst->pstFil,"noticia/","_", TRUE);
  Sub(dteTxt,7,8)+"/"+Sub(dteTxt,5,6)+"/"+Sub(dteTxt,1,4)
};
////////////////////////////////////////////////////////////////////////
PutDescription(
"Retorna una fecha en formato español, aaaa/mm/dd, al que estan acostumbrados
los lectores de la newsletter.",
PhtSpanishDate);
////////////////////////////////////////////////////////////////////////
```

inc.tol de con-Q.tv

Inclusion of common functions and application functions

Declaraciones

Inclusiones comunes

- `Set txtInc`
  Text functions.
- `Set setInc`
  Set functions.
- `Set filInc`
  File functions.
- `Set dirInc`
  Directory functions.
- `Set tmeInc`
  Macro expansor for Tol inside Html.
- `Set ftpInc`
  Ftp functions.
- `Set xsmInc`
  Xml site maps functions.
- `Set imgInc`
  Image functions for Html code.

Inclusiones de aplicación

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

Set txtInc

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

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 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 imgInc

```
////////////////////////////////////////////////////////////////////////////
Set  imgInc = Include("cmm/img.tol"); //
////////////////////////////////////////////////////////////////////////////
PutDescription("Image functions for Html code.", imgInc);
////////////////////////////////////////////////////////////////////////////
```

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

Set adsInc

```
////////////////////////////////////////////////////////////////////////////
Set  adsInc = Include("app/ads.tol");
////////////////////////////////////////////////////////////////////////////
PutDescription("Adverstiment functions.", phtInc);
////////////////////////////////////////////////////////////////////////////
```