# BASE-RAD

## Rapid Application Development for Gambas

### 0.0.123

### Components

gb.image gb.gui gb.form gb.db gb.db.form gb.desktop gb.eval gb.eval.highlight gb.util gb.form.editor gb.settings gb.form.mdi gb.logging gb.mime gb.pdf gb.poppler gb.xml

### Module Main

For compatibility with non-graphical applications that must have a Main method, there is this method that redirects to Starter, which is the one that establishes everything that is necessary for the program to work properly.

### Module Starter

Sets everything that is necessary for the program to work properly. In other words, it makes sure that the databases, directories, files, etc. exist.

- **Run** For compatibility with applications that are not graphical, this method exists, since it allows to operate both with a form and without it.

### Module bat

Files batch processing.

- **Html2Md** Convert an html file to Markdown using the pandoc program.
- **Html2Pdf** Convert an html file To pdf using the wkhtmltopdf Or pandoc program.
- **ImageConvert** Convert a JPEG image to another PNG format.
- **ImageR180Flop** Rotate an image 180ยบ and then reflect it on the vertical axis.
- **Wav2Ogg** Convert a WAV file to another OGG. As input parameter it requires a WAV file path. The OGG file will be created in the same location as the WAV and its path will be returned in case of success.
- **TexNodes** Create a nice and colorful diagram for the node-children model where the nesting level can vary from 2 to 3. As an input parameter it requires a text array where each item of it has the form node-level-1 \ tnode-level-2 \ tnode-level-n.
- **Latex2Pdf** Convert a tex file to pdf using the pdflatex program

- **Text2Ogg** Convert a plain text file to an OGG file. As input parameters it requires a valid path to a TXT file containing the text and the language.
- **AudioExtractor** This function extracts only the audio track from a video file. Returns the destination path of the extracted file
- **VideoExtractor** This function extracts only the video track from a video file. Returns the destination path of the extracted file
- **VideoMixer** Merge two files, one audio and one video, then returns the path of the created file.
- **MediaArrange** Organizes media files according To their metadata.
- **Jpeg2Gif** Convert images into small files to send by email. If the file is processed successfully, its path is returned.
- **Tiff2Jpeg** Convert TIFF images to JPG.
- **ImageMontage22** Creates an image montage in 2x2 matrix form, as input parameter requires a text matrix of the file paths. A list of the files created is returned.
- **ImageMontage24** Creates an image montage in 2x4 matrix form, as input parameter requires a text matrix of the file paths. A list of the files created is returned.
- **ImageBook** Convert scanned images from a book in TIFF format to others in JPEG format.
- **Jpeg2Png** Convert a JPEG image to a PNG format.
- **Jpeg2Pdf** Convert all JPEG images in the list into a PDF file. If the PDF file is created successfully, its path is returned.
- **JpegCopyRC** Creates a copy of a JPG file reducing its quality by a percentage that is passed as a parameter, then returns the path to the created JPG file.
- **JpegCopyRG** Creates a gray scale copy of a color JPG file reducing its quality by a percentage that is passed as a parameter, then returns the path to the created JPG file.
- **PngReduced** Creates a copy of a PNG file reducing its quality by a percentage that is passed as a parameter, then returns the path to the created PNG file.
- **PdfOptimize** Create an shrink copy of a PDF file using Ghostcript.
- **PdfDecompress** Create an uncompressed copy of a PDF file using the qpdf program.
- **PdfReplace** Replace one text string with another In a PDF file using the SED program.The PDF file must be uncompressed.
- **PdfDecrypt** Create an unencrypted Copy Of a PDF file using the gostscript

program.Then, If successful, it returns the path Of the created file.

- **PdfExtractImages** Extract all images from a PDF file using the PDFIMAGES program. Then the extracted images will be saved in the /tmp directory.
- **PdfPage2Image** Convert a page from a PDF file to an image file
- **PngOcrText** Extract the text from a PNG image using the TESSERACT program. If there was success, the text is returned.
- **JpegOcrText** Extract the text from a JPEG image using the TESSERACT program. If there was success, the text is returned.
- **PdfOcrText** Extract the text from a PDF page using the TESSERACT program. If there was success, the text is returned.
- **Pdf2Text2** Extract the text from a PDF page using the poppler utility PDFTOTEXT. If there was success, the text is returned.
- **PdfImageBW** Extract images from PDF in JPG format in Grayscale. Returns the folder where the images were extracted
- **PdfR90** Rotate a PDF pages 90 degrees.
- **Svg2Pdf** Convert an SVG file to PDF with the InkScape program.
- **PoList** Create a list with all the phrases in a .po file
- **SpreadConvert** Convert spreadsheet, CSV, pdf, or html files using the SSCOVERT utility of the GNUMERIC program.
- **Mdb2SQL** Generate statistics from .mdb files using the mdbtools program. As input parameter it requires the full path of the .mdb file.

## Module cad

Working with cad files.

- **GauchoSave** This load all the info form a gaucho xml file, then put it in a collection a nd return it.
- **GauchoLoad** This load all the info form a gaucho xml file, then put it in a collection a nd return it.
- **GauchoExportSVG** Export the information in cModel collection to an SVG file.
- **DXFImport** Import a DXF file into cModel collection.
- **DWGImport** Import a DWG file into cModel collection.
- **DWGLayersList** Get the DWG file layers list.
- **DWGEntityList** Create the stanard entities list.
- **DWGRead** Read a DWG file using the dwgread utility from libreDWG library.

- **DWGSections** Create a list of DWG sections using the previously extracted text with the dwgread function.
- **DWGObjects** Create a list of DWG objects using the previously extracted text with the dwgread function
- **DWGHeader** Create a list of DWG headers using the previously extracted text with the dwgread function
- **DWGDecodeCommon** Decode common fields to DWG entities.
- **DWGDecodeEntity** Decode the specific fields of an entity
- **DXFEntityDefaults** Creates a list with the codes of an entity indicated as a parameter.
- **DXFEntityCodes** Creates a list with all the codes of an entity indicated as a parameter.
- **DWGColors** Return a colors list by the form r,g,b where each index is the DXF color integer number.
- **GauchoDecodeColor** Input a DXF color return a RGB color.
- **GauchoConfig** Create a collection whith te program variables needed
- **DXFSections** Create a list of sections from a DXF file, for this it requires the raw text of the DXF file.
- **DXFBlocks** Creates a list of blocks from a DXF file, for this it requires the raw text of the DXF file.
- **DXFEntities** Creates a list of entities from a DXF file, for this it requires the raw text of the DXF file.
- **DXFHeader** Creates a list of headers from a DXF file, for this it requires the raw text of the DXF file.
- **DXFTables** Creates a list of tables from a DXF file, for this it requires the raw text of the DXF file.

## Module cdg

Coding utilities.

- **ProjInfo** Create a collection with the project metadata contained in the ".project" file. Then, the tags to access to information are:
  Title, Startup, Icon, Version, **Component**, Description, **Authors**, Language, Vendor, Iconart, Iconurl, Iconset, Iconlic, TabSize, Translate, SourcePath, Maintainer, Address, Url, License, Prefix, PackageName, CreateEachDirectory, RuntimeVersion, Packager, Systems, SameFiles, Menus, Groups.
  Note: All the tags return a **String** except for Component and Authors that return a **Variant**[]

- **ProjData** List all modules
- **CodeTag** ' strPath It is the root directory that is passed to the function, and from there it will look for the .project files' Here is the complete method It parses a text string that is passed as a parameter and, in the context of a code snippet, returns what that phrase is.
- **CodeMthod**
- **SpecialSubs** Create an array with the gambas special case subrutines and the event names.
- **CodeStructure** Create an array with the gambas language structures that, for example, can be used to interpret the code.
- **RelationProj** Read the method and code matrices of the project and then analyze the relationships between them generating a matrix with these relationships.
- **DokuProj** Read all the classes and modules of a gambas project, collect information from it in an open way and translate it into a text matrix.
- **DokuHtml** It returns an html with the functions of a module and all the data of these, as an input parameter it requires the root directory from which to search the modules.
- **Dokuwiki** Returns a wiki with the functions of a module and all their data, as an input parameter it requires the root directory from which to look for the modules.
- **GetFarmInfo** Returns an array with the data of a prawn project hosted on the farm, as input parameter requires the project identifier.
- **GetProjectInfo** Devuelve una matriz con los datos de un proyecto de gambas, como parametro de entrada requieres el direcotrio rasɣz del proyecto.
- **GetProjectInfoTags** Returns an array with the data of a shrimp project, as an input parameter you require the root directory of the project.
- **GetProjectsDir** List the shrimp project directories recursively from the directory passed to it as a parameter.

## Module dbs

Database utilities.

- **DataExport** Data export of a result in CSV format. The input parameters are as follows.
    - con - Open connection to the database
    - ctn - Collection with several items

  The collection must have the following items

    - File The full path of the file to be exported
    - Query The query that will be used to list the database data

    - Separator The list separator character that will be used in the export file

- **DBTemplate** Create a database template that is passed to it as parameters.
- **FileSqlLoad**
- **DBConf** Write the XML file of the database model
- **DBOpen** Start a database and if it does not exist it creates it.
- **DBSqlite** Starts a database or creates and starts it. Returns a connection and as an input parameter requires an array with the base parameters. If the database does not exist, then create one and start it. If the database does exist the method could take two actions, start it or create a backup and create a new database. stxDB contains the base parameters.
    - 0 - DBHost.
    - 1 - DBName.
    - 2 - DBPath
- **RecordPrimaryKey** Returns the name of the key field in the table. GEFStarter.conProgram As Connection is the connection to the database. tab As String is the name of the table on which you try to know what type of field it is. strFieldCheck As String It is the field to verify. strValueCheck As String Is the value of the record for the field to verify. GEFStarter.stxTableFields As String [] It is the list of all the fields in the database.
- **RecordValue** Returns the value for a given field and a key.
- **RecordKey** Returns the key value for a given field and a value. Note: The field must be of unique type.
- **GetForeignKey** Returns the foreign key for a value in a field. Note: The field must be of unique type.
- **GetTables** Extraction of the list of tables or views of the connection. It is passed a parameter, the type to list
    - Table
    - View
- **FieldInfo** Returns a collection with several items. Depending on the type of database, more or less data can be offered. Sqlite
    - Table
    - Name
    - Type
    - PrimaryKey
    - Unique
    - ForeignTable
    - ForeignKey

    - ForeignShow
    - Nulable

  Data that comes from a separate xml file from the database and that is editable by the user

    - Title
    - Tooltip
    - Group
    - Format
    - Filter

- **GetIndex**
- **RecordExist** Devuelve la clave si el registor existe y -1 si no existe. Como parametros de entrada requiere una conexisɛun, el nombre de la tabla y una coleccion con el campo y el valor de busqueda.
- **RecordDelete** Borra los registros de la tabla que coinciden con las claves de inx.
- **SqlMake** Create an sql query.
- **MakeTableView** Create an SQL statement that is used to create a view in a database.
- **MakeTableViewExtra** Create an SQL statement that is used to create a view in a database.
- **RecordNewRefTest** Insert a new record in the database.
- **RecordEdit** Edit an existing record in the database. If this is well inserted, the function returns the record key value, otherwise it returns -1
- **CheckTable** Function that checks if a table meets the minimum premises to work with it.
- **MDBtoSQL** Statistics of .mdb databases using mdbtools. As input parameter it requires the full path of the .mdb file Dependencies: mdbtools DB.V.T.Bytes TB.R.C.Bytes
- **DataType** Returns the name of the data type, the constant of the gb.db
    - -2 = Blob
    - -1 = Serial
    - 1 = Boolean
    - 2 = Serial
    - 4 = Integer
    - 5 = Long
    - 7 = Float
    - 8 = Date
    - 9 = String

- **MakeDBProfile** This function creates the text to put the in module DBFieldsTitles. As an input parameter it requires a text array with all the lines of the SQL file of the database.
- **MakeViews** Check if the views match whit the tables This function creates a view for each table to put later in the database creation SQL file.
- **RWords** Create a key words list.
- **UsualFieldName** Check if usual field names apply for a field.
- **UsualFieldsNames** Create a usual fields names list.

## Module dsk

Desktop utilities.

- **Mimex** Returns a list of the programs that are associated with the mime type that is passed as a parameter.
- **TextWidth** Returns the value in pixels of a text string, regarding the typography
- **MimeAppPicture** Returns the application picture.
- **ImageFrame** Return the same image but with a frame
- **FileChooser** Select the full path of a file, with the name and extensions. As optional you can pass a directory that is where the filechooser will go when it is opened. Also as optional you can pass a filter of file types separated by:, for example [txt","csv"]
- **DirChooser** Select the path of a directory using a dialog box.
- **GNumix** Send an email. Create a list containing all the relevant icons path in gnumix theme.

## Module epb

EPUB files utilities.

- **EpubStructureMaker** Creating a basic file and directory structure for packaging an EPUB file. As input parameter requires:

  - drt, Directory
  - std, Standard version, default EPUB 3.1
- **Xhtml** Returns the Xhtml and requires the content html as an input parameter.
- **Html5** Returns Html5 code and as input parameter requires the content html. Let's say you just add the headers to it.
- **EpubPackager** Create an epub file, which is basically a zip file with specific content and structure. The first parameter is the path of the layout directory and the second is the path of the epub file.

- **EpubContentOpf** Create the content.opf file with some parameters that are optionally passed to it.
- **EpubContentOpfBasic** Create the content of the container.xml file.
- **EpubStyleCSS** Create a CSS file content.
- **ContainerRead** Read the container.xml file to create a text array of the data.
- **BaseSections** Create a list of the basic sections of a book.
- **EpubSectionMaker**
- **ContentData** Devuelve una colecciꞔun con los metadatos de un libro.
- **EpubContentOpfReadFromZip** It reads the content.opf file from within the EPUB and returns the XML text.
- **EpubContenOpfEdit** Editing the content.opf file. For more details on Open Packaging Format (OPF) 2.0.1 v1.0.1 http://idpf.org/epub/20/spec/OPF_2.0.1_draft.htm
- **EpubContenOpfManifiest** Editing the content.opf file, add the manifest.
- **EpubContentOpfRead** Returns a collection with the content of a content.opf file. As an input parameter, require the xml text found in said file.

## Module fil

General files and directories utilities.

- **stat_**
- **Stat** Create a file parameters list using the GNU coreutils program stat. Note: the tags for access to the information are:
  Dev, Ino, Path, Link, Mode, SetUID, SetGID, Rdev, Size, BlkSize, Blocks, LastAccess, LastModified, LastChange
  Original https://www.gambas-it.org/wiki/index.php?title=Stat_()
- **RListDir** You must have a list of directives that are included in the route that is passed as a parameter.
- **FileCRC32** Obtaining a file crc32. As parameter require the complete path then as an optional U or L that will convert everything to Ucase or Lcase.
- **FilesNew** Returns a list of files in a directory that is passed as a parameter. Optionally, a list of existing files can be passed as a parameter, which will be omitted from the output list if they are found, and a file extension filter in the mp3 style:ods:txt
- **FilesExist**
- **FileVersion** Returns a text with the version of the file that is passed as a path. Use the command file from terminal. If Mode = True then the Short name is returned.

- **FileTrash** Move a file to the trash. Requires the file path to be passed and if manages to move to the trash, returns True and if it fails, it returns FALSE.
- **ScanFiles** Return a files list nested in a directory
- **ScanDirs** Return a directories list nested in a parent directory
- **FileReplace** Read a file content then replace a specified string and finaly create a new file with the text content. Note: Files are in input mode.
  Original code: https://foro.gambas-es.org/viewtopic.php?f=5&t=6056&start=10
- **FileTemplate** Taking a template file replace the labels with values. Returns a matrix with a list of files, first the product and then the pdf, in case any of these do not exist in the position of the matrix it will make an empty string.
- **ReadZip** Read a text file inside a zip. strZip is the full path of the zip file strFile is the relative path of the txt file inside the zip
- **FileLoad** Returns a text array where each item is a line from the text file. If the line is empty in the text file, that is, it is an empty line, it is not passed to the martix, therefore the result is a return without blank lines.
- **FileLoadRaw** Returns a text array where each item is a line from the text file. Even the blank lines
- **FileLog** A simple log system, where a new line with date and text is added to a text file, the file and the text are neded as parameters.
- **FileNospace** Returns a text, filename concatenating all the fragments passed to it shift to lowercase and removes characters outside the 97-122 ASCCI range.
- **PicData** Returns an array with the metadata extracted by the jhead terminal program.
- **FileExifPages** Returns the number of pages in the file if it does not have the tag then 1. ExifTool is used to extract this information.
- **ArrangePath** Returns a path without line breaks or problematic characters
- **DirBase** Returns the directory without the path to it, that is, it returns the name of the directory. if you pass "/home/user/music" it will return "music"
- **DirParent** Returns the parent directory of another one that is passed as a parameter.

## Module sog

Obtaining Operating system information

- **Resume** Create a collection whit the some syetem information.
- **DeFiBro** Returns the default file manager.
- **DistroNoshell** Returns the xmi installed distribution

- **Distro** Return the distro
- **ArqSO** Returns the operating system architecture
- **ArqMicro** Returns the Processor Architecture
- **MicroType** Returns the type of Processor
- **Ram** Returns amount of Ram in MB also with free -m
- **ComputerName** Returns the computer name
- **CurrentUser** Returns the active user
- **GetSystemUsers** ' Project standard information' Project standard information' Loading the .farm file Returns the a list of Linux system users.
- **AllUsers** Returns the users we have created
- **WGroup** Returns the Computer Workgroup
- **SysFile** Returns the filesystem
- **Vgambas** Returns the gambas version installed on the computer
- **AddressIP** ' strPath It is the root directory that is passed to the function, and from there it will look for the .project files' str = Replace(str, " ", "")' str = Replace(str, "", "")' str = Replace(str, " ", "")' str = Replace(str, " ", "") Returns the IP of the computer, it only works under OS with Systemd, for this it uses shell and the command "ip addr show".
- **LANIP** Returns the IP v4 of the local network, as input parameter requires the base IP address, for example "192.168.1" but if the parameter is not passed then use the IP of the computer where the program is running removing the last one number. The output format of each item in the array is host-name [tab] 8.8.8.8
- **LastNIP** Returns the last digit of the IP
- **Hdserial** Returns the serial number of the hard disk to be used as Pk in the BDD
- **UUIDswap** Returns UUID of the swap to be used as PK of the BDD (Requires Administrator permissions)
- **LastUpgrade** Returns the Date of the last time the system was updated
- **GetLcNumeric**

  Returns a text matrix with the configuration of the system number.

  1. Decimales
  2. Miles
  3. ??
  4. ??
  5. ??
  6. Codificacisœun

- **PkgStat** Returns the status regarding the installation of a package. It requires as an input parameter the exact name of the package.
- **XDGFolder** Returns a text matrix with the paths of the music folders, images, documents, etc. the /home/user directory
- **PkgDep** Check if the packages that are passed to you as parameters in an array are installed in the system, it returns an array with the packages that are not installed, if everything was, the returned array would be empty.
- **MimeTypesList** Returns a list of all the mime types that are covered in the system.
- **MimeDefaultApp** Returns a list with the default associated program and the path of the icon if it exists.
- **UUIDHandle** Returns an identifier of length N generated by the random combination of characters and numbers. It is for lists of objects that you want to name with names of 4 or 6 characters for example.
- **UUIDGen** Returns a Universal Unique Identifier generated by the **uuidgen -t** command. Just change the octet of the timestamp. Remember that it is generated based on the time "-t"
- **UUIDProc** Returns a universal unique identifier generated by loading the /proc/sys/kernel/random/uuid file. It always gives all the different octets for each time the function is run.
- **GambasComponents** Create a list with all the gambas components

## Module uty

- **DirParent** Returns the parent directory of another one that is passed as a parameter.
- **DirBase** Returns the directory without the path to it, that is, returns the name of the directory. if you give "/home/user/music" it will return "music"
- **ConType** Function that returns the type of container according to the variable that is passed as a parameter
- **ArrangePath** Returns a path without line breaks or problematic characters
- **Timestamp** Returns a text string over time in "yyyymmddhhnnss" format. From years to seconds.
- **TimeYear** Returns the year from a text string that is passed as a parameter.
- **Timestampu** Returns a text string over time in "yyyymmddhhnnssuu" format. From years to milliseconds with three digits.
- **TypeVar** Function that returns the type of variable as a word. As input parameter requires an integer number.
- **MouseButton** Function that returns the name in English of the mouse button that

has been pressed.

- **ArrayInclude** Remove from the list the texts that do not have the text string passed as a parameter
- **ArrayMax** Returns the maximum value of the list of integers.
- **Between** Returns the text string between the two that are passed as a parameter Original code by Juan Luis Lopez
- **Retab** Function that returns an array of text separating a string by the separated character passed as a parameter.
- **ArrayOrder** Sort a text list that has characters with accents and spaces, as the main parameter the list is passed to it and optionally the order asc or desc.
- **ArrayExclude** Remove from the list the texts that have the text string passed as a parameter
- **ArrayNoVaccum** Remove empty items from the list.
- **NamingCon** Returns a list of control names and their three-character abbreviation.
- **HMStoSeconds** Returns the time in seconds of a string that is passed to it in the format HH:MM:SS HOURS:MINUTES:SECONDS.
- **SecondsToHMS** Returns the time in a matrix where: 0-Y 1-M 2-D 3-H 4-N 5-S and as an input parameter requires the time in seconds
- **ListDeldup** Removes duplicate items from a list, requires a String [], and returns a String [].
- **WhereRun** Indicates if the program is running from the IDE or from an executable only using prawn code.
- **PathOrganizer** This function receives a list of paths, whether they are files or directories, it delivers a text matrix with the files that match the given extension and, in case the parameter recursive this True, also all the files from each directory.
- **DirGambas** It analyzes if the directory that is passed as an input parameter is a directory of a gambas project.
- **ScanDir** Scans a directory that is passed as a parameter in search of files can be filtered with a list of extensions separated by a colon ":".
- **Clocky**
- **StringSerial** Function that returns a text array of a string where each item is a character of it.

## Module vag

Utilities to validate and manage values.

- **StrCount** Function that count the times that a substring appear in other string

- **Formula** Evaluate a string to decode a formula.
- **Footprint** Returns a code that consists of counting the characters of all the words of a phrase that is passed as a parameter. for example for the phrase "santa claus" it returns "_1a2e1l1n1o1p2". This is interesting to find out if a name exists in a database regardless of the order, that is, first name, last name or vice versa.
- **ArrangeParagraph** Verify that a paragraph is in the correct form.
- **SQLtr** Converts the word into an expression that allows filtering with accents in a like statement.
- **SQLike** Converts the word into an expression that allows filtering with accents in a like statement.
- **OnlyLeters** Word validation.
- **CaptionCheck** Validation of the text of a control in KDE the text of the buttons for example has an ampersand before the text.
- **Diacritics** Function that tells if a text that is passed as a parameter has characters diacritical, that is, accented for example.
- **NoSymbols** Only letters and numbers, and the letters without accents.
- **OnlyNumbers** Returns a text with only numbers.
- **OnlyCyrilic** Validation of words in Cyrillic.
- **OnlyText** Validation of latin text only, Numbers NO, Double space NO, Space At the beginning and / or at the end NO.
- **OnlyTextCyrilic** Validation of Cyrilic text only, Numbers NO, Double space NO, Space At the beginning and / or at the end NO.
- **ConvertPath** Decode the hexadecimal characters in the URIs by traversing the given string Params: strInput the URintPos to decode Return: the decoded URintPos
  Original code http://foro.gambas-es.org/viewtopic.php?f=1&t=2054&postdays=0&postorder=asc&start=10
- **ChkExt** Checks if the extension has the short name and if so changes it to the long one.
- **ChkSeparator** Given a text file separated by commas, tabs etc. This function analyzes which is the separator character. To do this, it checks all the ascii characters and for each line and if they are in all the lines then it lists them. Once we have the set of characters that are in all the lines, the occurrences of each one in each line are counted and as soon as some character appears different amount of times in two different lines, it is discarded.
- **StrColor** Convert the color passed in hexagesimal text format, such as #000000 into a number to be used in the prawn code.

- **SplitText** Partition a text given as a parameter, if the second argument, which is the word or cut letter, is null, each item of the array will be a character of the text string, if, on the contrary, a parameter of cut and it exists in the chain, it will be divided by this parameter. But in the case that a cut phrase is passed and it does not exist, the same original phrase will be returned without altering as zero item of the matrix.
- **ID** Function that returns the identification number.
- **CapitalWords** Returns a text where each word has the first letter in uppercase and all subsequent letters in lowercase.
- **Capital** Returns a text with the first letter in uppercase and all subsequent letters in lowercase.
- **Chek4SQLscript** Returns a text suitable for SQL queries, removes line breaks and non-compatible characters.
- **Chek4SQL** Returns a text suitable for SQL queries, removes line breaks and characters not compatible with SQL statements.
- **Paragraph** Paragraph validation, double space NO, Space at the beginning and / or at the end NO, line breaks at the beginning and / or at the end NO etc.
- **RSpaces** It removes the spaces at the beginning and the end of a string and if there are repeated spaces between the beginning and the end, it converts them into one.
- **UnTager** Rename the standard html tags to internal nomenclature tags.
- **ReTager** Rename the internal nomenclature tags to standard html tags.
- **XmlValidate** Convert a text string to conform to the XML standard
- **MinimalMatrix** Given an amount of elements, calculate the minimum square matrix that contains that amount.
- **DropFiles** Returns a files list from the drop text

## Module xmg

Managing XML files.

- **XmlSVG** Crete a simple SVG file.
- **XmlDwg** Crete a simple xml file with minimal graphical info. For use in GauchoCAD.
- **XmlConfRead** Returns a collection. As input parameter, require the path to the xml file.
- **XmlKeyValue** Returns a text array with all the values found with the tag that is stopped as a parameter. As input parameter it requires the xml text and the key from which you want to obtain the value.
- **XmlBook** Saves the data that is in the collection of variables in the configuration

XML file. Creating a data file associated with a PDF file to save data about the cover structure, index, content, etc.

- **XmlConfMaker** Creation of initial xml configuration file.
- **QueryXML** Query an attribute of in an xml file. It works with the parent / child [attributes] structure and returns a collection of the form attribute: value.
- **GetVar** Returns the value of a variable (a node of the xml) to "consult" this information should be done in the general form:

    - myvar = xmg.GetVar ("NodeName/AttName/AttReturn")
    - myvar = xmg.GetVar ("AttributeName") '(If nodes called "Variable)
    - myvar = xmg.GetVar ("Variable/Name/Value") '(General case)
    - myvar = xmg.GetVar ("Table/products/title") '(from a table)
    - myvar = xmg.GetVar ("Field/Productname/Type") '(table called "products)

- **XMLTagAtt** Write a var value in some file. Query an attribute of in an xml file and extract an attribute from it
- **EditXml** Saves attributes of tables and fields in an xml file works with the name of the element (the type) and then with the attributes of this and, depending on the type of element, its parent. Saves attributes of tables and fields in an xml file works with the name of the element (the type) and then with the attributes of this and, depending on the type of element, its parent. Editing an xml file, such as SVG to edit the value, color etc. of a text by the element tag method. This way of working allows the file to be edited multiple times, since, as the tag remains unalterable, it can be accessed repeatedly.
- **EditXml2** Editing an xml file, such as SVG to edit the value, color etc. of a text by the element tag method. This way of working allows the file to be edited multiple times, since, as the tag remains unchanged, it can be accessed repeatedly.
- **XmlTagValue** Editing an xml file, such as SVG to edit the value, color etc. of a text by the element tag method. This way of working allows the file to be edited multiple times, since, as the tag remains unalterable, it can be accessed repeatedly. They can also be altered image paths. Returns a text matrix where each item is a Tag [tab] Value pair. As input parameter require the xml text.
- **XTitle** Search the title of a field or a table in the XML text that are in the xml item of the Collection which is passed as a parameter. The items that must be in the collection are

    - Table
    - Field (Only if it asks for the title of a field)
    - Xml (The xml text extracted from the configuration file)
    - Attrib (The attribute queried)

- **XmlIni** Create a collection with the variables and their initial parameters