

Main differences between V1.13.2 and V1.13.2

This version only takes into account the following two points:

- The [GComponentList](#) contextual menu (Items+/-) was not always correctly updated versus the "Remove" possibility.
- It is now possible to use the [setTab\(\)](#) method to modify the content of a tab of a [GTabbedPane](#) object.

Main differences between V1.13.1 and V1.13.2

This version includes these evolutions and corrections:

- On the [GEntryReal](#) widget, the units field is now, by default, stuck just on the right of the entry field. It is no more necessary to overwrite constraints.
- A new type of units is now available managing Bytes.
- When using log file generation, it is now possible to save it in the middle of an execution.
- The forcedStatus management was not correctly done dor the [GEntryString](#) widget.

Main differences between V1.13 and V1.13.1

This version includes these evolutions and corrections:

- On the new [GEntryFileName](#) widget:
 - add a [setLabel\(\)](#) method.
 - add the [getSubComponent\(\)](#) method to rule the constraints.
 - the selected file was not correctly updated when we read a context file.
 - The [GReadWrite](#) interface was not fully respected as the [GTextAreaWithLabel](#) widget did not include it : it is now the case.
- A [resetContextFileName\(\)](#) method has been added in the dans la [GMainFrameAbstract](#) class.
- We can force the error/warning status in the [GMultipleChoice](#).

Main differences between V1.12.1 and V1.13

Summary

- [1 Native Java options when launching a process](#)
- [2 New GEntryFileName widget](#)
- [3 New version of JFreeChart](#)
- [4 Customization](#)
- [5 Anomaly corrections](#)

Native Java options when launching a process

Previously, it was only possible to launch a process as via the [GJavaCommandLauncher](#) class only by specifying the runnable class name and its arguments. Now, there is a new argument allowing to call for native Java options (as heap space size)

New GEntryFileName widget

A new widget ([GEntryFileName](#)) allowing to enter a file name is now available. It consists in:

- a text field where it is possible to enter directly the name but it will verify if the file exists and, if it is not the case, will send an error status
- a button to launch a browser where to search the file
- a specific mechanism to display the relative or the absolute path name.

New version of JFreeChart

Now, **GENIUS** is using [JFreeChart](#) V1.5.0.

Customization

We added some colors customization for the following widgets:

- Text color for [GHyperlinkLabel](#)
- Colors for error and/or warning status
- Background color for [GMainFrameAbstract](#) console.

Anomaly corrections

- Bad management of [GTime](#) widget forced status
- Adding the setDefaultValue for [GTime](#) widget
- Uncompleted test on [GEntryReal](#) widget : now, we test Double.POSITIVE_INFINITY and Double.NEGATIVE_INFINITY and not only Double.NaN
- For a standard application, when we tried to load a context then, cancel the operation, the previous context name was cleared. Now it is kept.

Main differences between V1.11.4 and V1.12.1

Several interesting evolutions and corrections in this new version:

Summary

- [1 GCalculator with SQLite file](#)
- [2 Possibility to read a file where a tab is missing](#)
- [3 Buffered console in the standard application GUI](#)
- [4 Possibility to save a GPlot configuration](#)
- [5 New display formats for GTime](#)
- [6 Anomaly corrections](#)

GCalculator with SQLite file

Now the [GCalculator](#) widget used inside the [GPlotPanel](#) widget is able to manage [SQLite](#) files.

Possibility to read a file where a tab is missing

In the previous versions, if we added a new tab in our application, it was not possible to read an old context with only (n-1) tabs. Now, as for any other basic input widgets, this fonctionnality is available using "by default" values.

Buffered console in the standard application GUI

There is the possibility to use a [GBufferedTextArea](#) widget for the console of a standard application. Nevertheless, it will be mandatory to add an "exit" line of code at the end of the batch computation:

```
batch = new BatchSW(nomFicData, nomFicEphem);
batch.compute();
System.exit(0); // Useful when GBufferedTextArea is used
```

Possibility to save a GPlot configuration

Now, using [GPlotPanel](#), it is possible to save a plot configuration (then to load it) that will include:

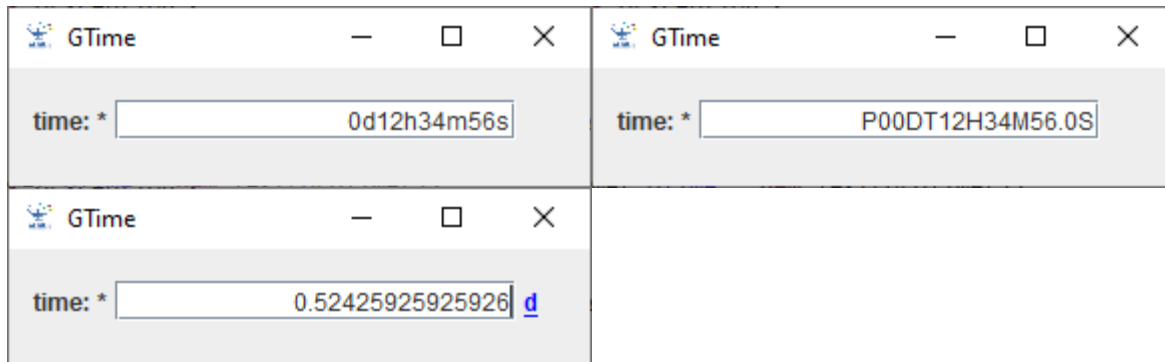
- name of the used file,
- type of plots,
- titles,
- zooms,
- ...

This fonctionnality is still incomplete and will be improved in the next versions.

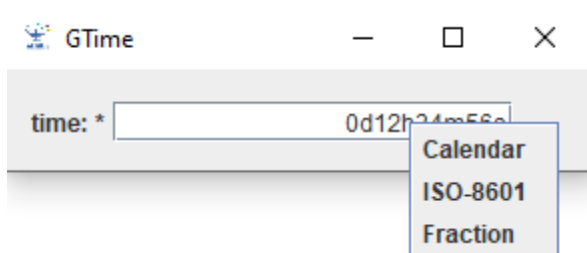
New display formats for GTime

There are new possibility for duration display using [GTime](#) widget :

- as in the previous version (calendar format),
- as a double value with units "s", "m", "h", "d", etc.
- using ISO-8601 format (for example PT6H for 6h00)



Change (and conversion) of the format is done using a right click on the input field.



Anomaly corrections

- [GTime](#):
 - The unit used for a day is "d" and no more "j" (even if this unit is maintained).
 - Method [update\(\)](#) is now public.
 - The name of the method to add an interval of validity is now [addGInterval\(\)](#) rather than [addInterval\(\)](#).
 - The text used for the interval of validity tooltip is corrected and more explicit.
- [GTabbedPane](#): increasing robustness when tabs are created when reading a context file.
- [GComponentList](#): now the "Duplicate" option only appears if [Cloneable](#) interface is available.

Main differences between V1.10.1 and V1.11.4

Some interesting evolutions and corrections have been implemented in this version. No incompatibilities have been identified versus V1.10.x source code (except the fact that Java 1.8 is now necessary).

Summary

- [Java 1.8](#)
- [GPlotPanel equations](#)
- [GPlotPanel pop-up windows for event comments](#)
- [GStandardApplication customisation](#)
- [GStandardApplication message window size](#)
- [Best mouse efficiency for scrolling](#)
- [New GTime widget](#)
- [Anomaly corrections](#)

Java 1.8

Now, **Java 1.8** is needed for generation.

GPlotPanel equations

A very interesting evolution with the possibility to create new variables using initial ones from a **MADONA** column file via a specific equations (see [here](#)). To do it, GENIUS used a new dependency with [exp4j](#) V0.4.8. Nevertheless, this functionality is not yet available for [SQLite](#) files.

GPlotPanel pop-up windows for event comments

Now, when events are displayed, there is the possibility to see the comments associated to these events by passing the mouse over.

GStandardApplication customisation

Some access methods are now available to easily add/remove/shift main menu bar items (for example, removing in the "Options" menu, the "debug" choice and replace it by and "expert" choice).

GStandardApplication message window size

The user may increase or decrease the message window size interactively (with the mouse).

Best mouse efficiency for scrolling

The mouse scrolling has been improved.

New GTime widget

A new [GTime](#) widget is now available.

Best precision management when reading MADONA XML files

Some values read in a **MADONA XML** file, once they are converted in **International System (IS)** units may occur a problem of precision. In case of a conversion from degrees to radians, it is quite normal but it is more astonishing for a conversion from km to m as we have just to add a 1000 factor: for example, 0.03827 km will give 38.269999999999996 m. Therefore, **GENIUS** uses now locally **BigDecimal** numbers to improve as much as possible these conversions.

Anomaly corrections

- In the previous versions, there was a possible bug when reading **MADONA** columns files if the first column number was not "0".

Main differences between V1.10 and V1.10.1

Anomaly correction

- Correction of the anomaly in [GPlotPanel](#) when more than one “date formatted” columns are present in a MADONA file.

Main differences between V1.9.1 and V1.10

Summary

- [New Reset fonctionnality](#)
- [Blank characters in GContextFileManagement widget](#)
- [Possibility to plot EVENTS files](#)
- [Mouse efficiency for scrolling](#)
- [GList widget improvements](#)
- [GFreeChartXY widget improvements](#)
- [Writing MADONA column files improvements](#)
- [Anomaly corrections](#)

New Reset fonctionnality

By using the [Copy & Paste](#) fonctionnality, it is also possible to reset the data.

Blank characters in GContextFileManagement widget

Now, when using [GContextFileManagement](#) widget setting directly a file name with blank characters before or after the name, it will correctly take into account it.

Possibility to plot EVENTS files

It is now possible to plot "EVENT" files coming, for example, from the [PSIMU](#) software.

Mouse efficiency for scrolling

The mouse is now more efficient when we want to scroll.

GList widget improvements

The following methods have been added to the [GList](#) class:

- [removeAllElements\(\)](#)
- [getSize\(\)](#)
- [getJList\(\)](#) for getting the original [swing JList](#) object
- [setVisibleRowCount\(\)](#) calling the method of the [swing JList](#) object.

At last, command **Ctrl-a** works better.

GFreeChartXY widget improvements

The [GFreeChartXY](#) widget has now the possibility, after selecting a point on a plot (or anywhere on the (x,y) plane), to get the coordinates of this point. This can be very useful if we want to execute some process using these coordinates. This functionality has also been implemented in the two higher level widgets [GPlotPanel](#) and [GGroundPlotPanel](#).

Moreover, some changes have been done:

- by adding a constructor with a secondary axis but without the *widgetId* argument
- by adding a setter ([setForcedDrawLine\(\)](#) method) to get the possibility to plot both with plain line and symbols.
- by adding the possibility to modify the automatic legend with [removeLegend\(\)](#) et [addLegend\(\)](#) methods.
- by changing how to call the [addSerie\(\)](#) method, removing *serieName* and *shape* redundant arguments

Writing MADONA column files improvements

A new constructor of the [MadonaWriter](#) class is available, allowing to precise how many digits we want

```
public MadonaWriter(final List<String> headerInfoLines, final int colPrec)
```

Moreover it is now possible to specify the amount of blank characters (4 by default) between two columns with the [writeColumns\(\)](#) method of the [FileColumnWriter](#) class:

```
public void writeColumns(final String spaceCol)
```

Anomaly corrections

- The information about discontinuities between two points to be plotted (thanks to a threshold value given in a [Madona file](#)) is now correctly managed.
- Using [GPlotPanel](#) widget:
 - when we use a [SQLite](#) file, we can now display data together from the ephemeris table data but also from the events table.
 - calendar date axis display is done when the unit is *~cal* and no more only if the name of the variable is "DATE".
- Using [Standard application](#) widget, when the pre-processing inhibited the process launch (for example due to an error in the data), the message is now *"start computation aborted ..."* rather than *"start computation ..."*

Main differences between V1.9 and V1.9.1

Anomaly correction

- Correction of the anomaly in [GPlotPanel](#) when plotting data function of calendar dates for specific dates involving some gaps (summer/winter time).

Main differences between V1.8 and V1.9

Summary

- New GList and GListSelect widgets
- Finding and reading multiple data
- New GTree widget
- GplotPanel improvements
 - Symbol choice in GPlotPanel widget
 - Calendar date format in GPlotPanel widget
 - Possibility to plot several files in GPlotPanel widget
 - Availability of a tool based on GPlotPanel widget
- New units
- Indication when loading a configuration
- Better file suffix management for the GSaveResults widget
- Minor improvements for GComponentList widget
- Anomaly corrections

New GList and GListSelect widgets

[GList](#) and [GListWithLabel](#) widget has been added allowing to display a list of strings (a bit like [GComboBox](#)) but allowing selecting several elements.

Moreover, a higher level [GListSelect](#) widget is also available allowing selecting elements of a list to copy/move them in another one.

Finding and reading multiple data

GENIUS already owns static methods from class [GFileManipulation](#) allowing reading or writing data. Nevertheless, these methods only work at a single level of the data structures and do not include multiple search.

Now, it is possible to search inside a **XML** file (or a **GENIUS** data structure) one or several data "sub-structures" using mainly the [GMultipleStructureChoice](#) class.

A typical application of such a functionality is to get some data from a "A" application to get it in a "B" application.

New GTree widget

As it is used by the [GMultipleStructureChoice](#) class, a new [GTree](#) widget is now available.

GplotPanel improvements

Symbol choice in GPlotPanel widget

It is now possible to select which symbol will be used for a plot (line for a continuous draw or symbols as circle, square ... for dotted lines).

Calendar date format in GPlotPanel widget

Dates are no more displayed in Julian days but in a calendar format.

Possibility to plot several files in GPlotPanel widget

Since this version, GPlotPanel allows taking into account several files for plots.

- the by default mode is still considering only one file.
- if we need several files, the first one is considered as the "reference" one.
- other files will be taken into account only if they include variables existing in the "reference" file (as read in the file header).
- user could decide to plot directly the values contained in the files or to plot relative values thanks to the "reference" file.

Availability of a tool based on GPlotPanel widget

The [GPlotPanel](#) widget is quite complex and powerful, allowing manipulations after reading text or [SQLite](#) files. Nevertheless, the only way to use it up to now was to develop a [Java](#) code or to call a higher level tool that uses it (as [PSIMU](#)).

That is why, a compiled tool with its own [GUI](#) is available with this version.

New units

In previous versions, the unit symbol for a day was "j" (*jour* in French ...). Now, it is also available as "d". Moreover, two other units are available with "mo" for month (or *mois* in French ...) and "y" for a year (or "a" for *année* in French). Conversions are the following ones:

- 1 year = 365.25 days
- 12 months = 1 year
- 1 day = 86400 s

Indication when loading a configuration

Some configurations may be very long to be loaded because of the amount of data. In such a case, an indicator (a turning wheel) will show that the application is still running.

Better file suffix management for the GSaveResults widget

In previous versions, it was not possible to manage different suffixes for any kind of results files. Now, it is the case, using an additional [API](#) for the [addSingleResultFile\(\)](#) method.

Minor improvements for GComponentList widget

- adding the [add\(final int rank, final GComponent item\)](#) method to add a component to the list.
- adding the [remove\(final int rank\)](#) method to remove an element in the list.
- adding the [setCurrentItemRank\(final int rank\)](#) method to select the current element.

Anomaly corrections

- Correction of the management of additional panels for the [GPlotPanel](#) widget.

Main differences between V1.7 and V1.8

Summary

- [Clear constraints](#)
- [Complex status management](#)
- [Absolute path for condensed status](#)
- [GPanel interval margins](#)
- [GDetachedPlotPanel list](#)
- [Specific icon](#)
- [Ground track widget](#)
- [Anomaly corrections](#)

Clear constraints

It is now possible to reset constraints of a widget using the [clearConstraint\(\)](#), [clearInnerDescendantConstraint\(\)](#) and [clearAllInnerDescendantConstraint\(\)](#) methods.

Complex status management

Thanks to the [setForcedStatus\(\)](#) method, it is now possible to manage more complex status as above several entries.

Absolute path for condensed status

When managing error/warning status, it is now possible to get directly the absolute path of a data using the [getPathInConfigFile\(\)](#) method.

GPanel interval margins

For [GPanel](#) only, we may manage internal margins using the [setMargins\(\)](#) method.

GDetachedPlotPanel list

It is now possible to get the list of detached panels (useful for needing updates or refresh of such panels).

Specific icon

Using the static [GEnvironment.setIcon\(\)](#) method, it is now possible to use a specific icon for your application rather than the by default Java one. Nevertheless, Under **Windows OS**, it is not yet possible to change the icon of an executable jar file (except creating a shortcut).

Ground track widget

A new widget is available allowing to plot ground tracks above a planisphere image.

Anomaly corrections

- An anomaly occurred in [GPlotPanel](#) when trying to plot data function of absolute dates for specific dates involving some gaps.
- When writing MADONA files, an additional line is created just after the header; hence these files are no more directly readable by the [GPlotPanel](#) widget.

Main differences between V1.6.2 and V1.7

Summary

- SQLite files
- GPlotPanel evolutions
- GFreeChartXY evolutions
- Status notion for GComboBoxWithLabel widget
- Copy/Paste/Import/Export functionality for a component of a widget list
- GArithmeticException
- GUnitFactory
- Anomaly corrections

SQLite files

GENIUS proposes now possibilities to store data in [SQLite](#) formatted files. It is particularly interesting for computation results as ephemeris or any kind of numerical results.

GPlotPanel evolutions

- Now, GPlotPanel may interpret SQLite data files.
- It is possible to initialize the starting directory when the widget is created (or via a setter method).
- It is also possible to deactivate the possibility to choose the file to plot (considering the name of the file has been initialized previously inside the code) : it allows to use GPlotPanel for fixed configurations with the setSelectedFile() method.
- Plots properties are available via the setDatasetPlotProperties() method (continuous plots, point by point, etc ...).
- At last, it is now possible to manage plots discontinuities.

=> incompatibility in the code due to a different argument in the constructor.

GFreeChartXY evolutions

It is now possible to define the kind of shape for plots drawn point by point and to add tooltips for such points.

And as for GPlotPanel, it is now possible to manage plots discontinuities.

=> incompatibility in the code due additional arguments for the addSeries() method.

Status notion for GComboBoxWithLabel widget

It is now possible to associate an **error/warning** status to some values proposed in a [GComboBoxWithLabel](#) widget.

Copy/Paste/Import/Export functionality for a component of a widget list

GENIUS gives now the possibility to activate **Copy/Paste/Import/Export** functionality for elements of a widget list for example in [GComponentList](#) (useful for some [GENOPUS](#) widgets).

GArithmeticException

A new exception has been introduced: ([GArithmeticException](#)). It is raised every time we try to put a **NaN** value in a real number widget ([GEntryReal](#), [GEntryRealVector](#), [GSliderReal](#) and [GSliderRealWithLabel](#)). Indeed, we can't have **NaN** inside a textfield but getting it inside a context file and read it, or via the [setValue\(\)](#), [setDefaultValue\(\)](#) and [setSavedValue\(\)](#) methods. This exception is important to protect **GENIUS** low levels widget against **NaN** as well as [ArithmeticException](#) returned by some **PATRIUS** objects in [GENOPUS](#) widgets.

GUnitFactory

The static [GUnitFactory.getGUnitArray\(\)](#) method is now available to simplify the creation of table of units.

Anomaly corrections

- Copy Paste on [GTable1D](#) widget: after executing a "copy/paste" action on a [GTable1D](#) widget and if the result is only one modified vector (one value inside the vector but no size changing), the " * " character indicating a modification occurred now appears.
- Update of the title for [GPlotPanel](#) widget: in previous versions, after loading the first plot from a file, the title was "calculated" thanks to abscissa and ordinate which was required. On the contrary, after a "Clear" and even if a new file is loaded, the title was no more updated. It is no more the case: after activating the "Clear" button or if a new file is loaded, the title is set to a void character string.

Main differences between V1.6.1 and V1.6.2

The identified bugs on the V1.6.1 have been corrected:

- Now, the JVM will no more continue to run after exiting a GENIUS application when using the GStandardApplication widget.
- It is now possible to read a MADONA XML formatted files as the absolute path of the file contains some blank characters.
- In GStandardApplication, texts are always added at the end of console areas.

Main differences between V1.6 and V1.6.1

This version only corrects an anomaly when using "headless" mode (i.e. using code without display definition as for clusters).

Otherwise, it is completely equivalent to the V1.6 version.

Known bugs

- Sometimes the JVM will continue to run after exiting a GENIUS application when using the GStandardApplication widget => will be corrected in V1.6.2
- After reading a MADONA XML formatted files, if an error (exception) is raised as the absolute path of the file contains some blank characters, nothing occurs (i.e. no messages appear while data do not have been loaded) => will be corrected in V1.6.2

Main differences between V1.5 and V1.6

Some minor evolutions and corrections have been implemented in this version. No incompatibility has been identified on V1.5 code source.

Summary

- [Creation of the GEntryConstant widget](#)
- [Creation of the GStandardApplication widget](#)
- [Creation of class and methods to build VTS files](#)
- [Unique GUnit class](#)
- [Possibility to set the minimum and maximum amount of elements in GComponentList](#)
- [Not useful scrollbars in GComponentList](#)
- [Possibility to get one or several columns data with the getColumn\(\) method of GPlotDataMadonaReader](#)
- [Evolution of GCommandLauncher to allow two different GTextArea interfaces \(Output & Error\)](#)
- [Non recoverable exceptions management](#)
- [1Anomaly corrections](#)

Creation of the GEntryConstant widget

This widget is a specific [GEntryConstant](#) widget offering the display of a dedicated button to get predefined values. It is useful for specific physical "constants" that depend on a model as gravitational term, equatorial radius, etc ...

Creation of the GStandardApplication widget

Specific widgets have been created to get a "standard" main application frame (see specific [How to build a Standard Application GUI](#) topic)

Creation of class and methods to build VTS files

It is now possible to use the objects to create VTS files (ephemeris, attitude and event ones):

- [VTSHeaderInfo](#)
- [VTSEPHWriter](#)
- [VTSATTWriter](#)
- [VTSEVEWriter](#)

Unique GUnit class

The internal units management has been slightly modified in order not to have specific units for temperatures and percentages. This management also allows to manage consistency between different units (i.e. « km » is consistent with « m »). This functionality is mainly used by the [GPlotPanel](#) widget.

Possibility to set the minimum and maximum amount of elements in GComponentList

It is now possible to set a minimum and/or maximum values for the size of the list displayed (useful if the widgets are memory consuming and may cause some JVM memory error).

=> possible compilation error due to the add of an exception when using the `setList()` method.

Not useful scrollbars in GComponentList

A side effect of a V1.5 evolution was the fact that, by default, a [GPanel](#) has its horizontal and vertical scrollbars activated (if needed). As [GComponentList](#) creates intermediate [GPanel](#), it ensued a not very ergonomic scrollbar stacking. Now, these intermediate scrollbars do not appear anymore.

Possibility to get one or several columns data with the `getColumns()` method of GPlotDataMadonaReader

In the previous version, this method allowed getting to data columns : one for the abscissae, the other for the ordinates. Now it is possible to get as many columns as possible.

Evolution of GCommandLauncher to allow two different GTextArea interfaces (Output & Error)

Modification of [GCommandLauncher](#) to allow to send `stdout` and `stderr` flows towards two different [GTextAreaInterface](#). If only one [GTextAreaInterface](#) is given, both `stdout` and `stderr` will be sent.

Non recoverable exceptions management

Some exceptions cannot be directly caught. As an example, in the previous versions, we may raise an exception when we enter a wrong date format in a [GEntryDate](#) field. The result was displaying a pop up window including the stack trace of the error which is not very ergonomic. Now this sort of pop up windows will not appear but is possible for your application to get it or to have a specific pop up window. A specific tutorial is available in the ([Training:genius examples.zip](#)).

Anomaly corrections

- Error when reading a date with [GPlotDataMadonaReader](#): in the previous version, reading a date could occur up to one second error.
- After a copy-paste action on a [GTable1D](#) widget, the result was a single modified vector with only one modified value and no change on the size of the vector. Moreover, the "*" sign did not appear.

Main differences between V1.4.1 and V1.5

Summary

- [Better management of the "*" sign when read/write and copy/paste actions](#)
- [Data status for GtabbedPane](#)
- [Exit Event test on GFrame](#)
- [GPlotDataReader interface creation and possibility to read dates with GPlotPanel](#)
- [Global variables management](#)
- [Some improvements for GBufferedTextArea](#)
- [No log file creation by default](#)
- [Modification of the stdout management with G\[Java\]CommandLauncher](#)
- [Anomaly corrections](#)

Better management of the "*" sign when read/write and copy/paste actions

There are both aspects covered by this change :

- first, the apparition (or not) of the "*" character after a copy/paste action is more intuitive
- then it is possible now to customize the way these "*" characters will appear after reading or writing a configuration file (see specific [GReadWrite interface and data files management](#) and [Modified data topic](#)).

=> incompatibility in the code (due to the add of a boolean flag for `GFileManipulation.readConfig()`, `GFileManipulation.writeConfig()`, `GContextFileManagement.selectLoadFile()` and `GContextFileManagement.selectSaveFile()` methods).

Data status for GtabbedPane

Now, the data status management is automatically taken into account by the [GTabbedPane](#) widget. Moreover, each tab can be configured individually to display the status of the contained panel or not.

Exit Event test on GFrame

We can customize the behaviour of a [GFrame](#) after clicling on the "x" case. For example, it is then possible to manage this behaviour by displaying a [GFrame](#) window asking if we actually want to quit or not (using the fact that this "click" event is now managed as a **GENIUS** event).

GPlotDataReader interface creation and possibility to read dates with GPlotPanel

It is now possible to interpret dates with [GPlotPanel](#) (see specific [GPlotPanel](#) topic). Moreover, it is possible to create our own way to read files via the [GPlotDataReaderInterface](#).

Global variables management

This version proposes an easy solution to manage the update of the content of a widget depending on the modification done on another one when both widgets are not included in the same higher level widget. It is explained in the following [Update data](#) topic

Some improvements for GBufferedTextArea

The behaviour of a [GBufferedTextArea](#) is more intuitive (for example, the number of the first page is no more 0 but 1 !).

No log file creation by default

By default, in the previous versions, a **Genius.log** file was automatically created. Now it is no more created and if we want to do it, we have to use the [GEnvironment.setLogFileName\(\)](#) method.

1.1. MODIFICATION OF THE STDOUT MANAGEMENT WITH G[JAVA]COMMANDLAUNCHER

Due to the introduction of the [GTextAreaInterface](#) (see V1.4.1) as argument of a [GJavaCommandLauncher](#) object, a side effect occurred for stdout management. So, if we want to duplicate output in a console (or to get this output in a console when no [GTextAreaInterface](#) has been passed [null argument]), we need to use [setCopyOutputToStdout\(\)](#) method.

Anomaly corrections

- [GComboBox](#) and [GComboBoxWithLabel](#) before management.
- [GEntryInt](#) and [GEntryDate](#) tooltip management.
- Amount of digits in the tooltip not consistent with the one chosen for the input area.