GLayout

De Wiki
Aller à : navigation, rechercher
GLayout

GENIUS gives a specific Layout (based on MigLayout) well adapted to conditional display.

Remark: since the V1.2 version, a new API is availaible to manage it. The old one (using strings) is yet available via the setStringContraint() method. It is preferable to use the new API. Anyway, the old one will allow to access to all Miglayout funnctionalities.

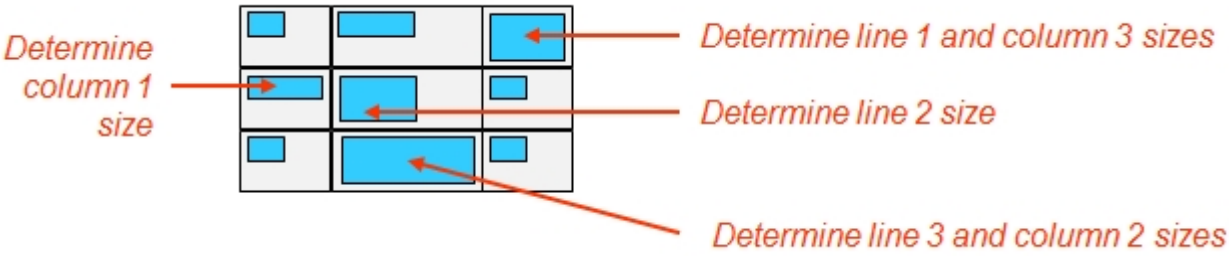
By default, every new graphic widget will be set to the next line,

```
public class MyPanel extends GPanel {

  private GButton but1;
  private GButton but2;
  private GButton but3;

  public myPanel() {
    but1 = new GButton("Bouton 1");
    but2 = new GButton("Bouton 2");
    but3 = new GButton("Bouton 3");
  }

  public void display() throws GException {
    put(but1);
    put(but2);
    put(but3);
  }

  public void generic() { }

};
```

Thus, the result of such code will give us:

Anyway, be careful of the fact that this layout is based on on a grid cell => the size of a cell may

depend on another component situated below ...

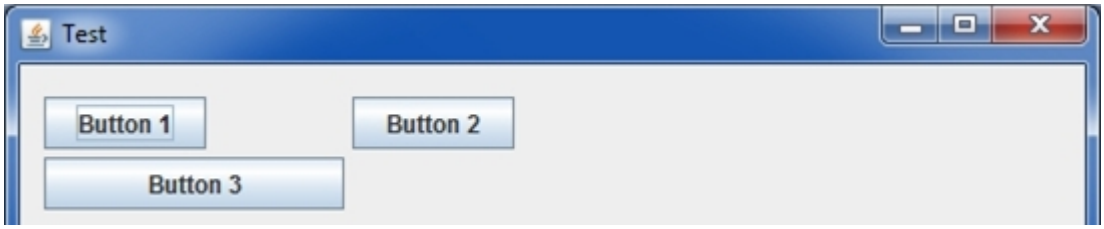


The setConstraint method

We may use, for each object, the setConstraint() method:

```
but2.setConstraint(null); // No more “by default” way => on the same line
but3.setConstraint(new GConstraint(GConstraint.newline(),
GConstraint.width(150)));
```

So, we see that to use this method, we must give it a GConstraint object as a single argument. This object will wait itself for one or several arguments which implement the GBasicConstraintInterface Interface. Most of the MigLayout constraints are available via static methods already proposed by the GConstraint class as in the previous example GConstraint.newline() or GConstraint.width().



For information, the "old fashion" will use the following syntax:

```
but2.setConstraint(""); // No more “by default” way => on the same line
but3.setConstraint("newline, width 150");
```

We can also take into account all the objects of a given type included in a GPanel by calling another setConstraint() method:

```
// Height of all the buttons of the panel "pan" fixed to 50 pixels
this.setConstraint(GButton.class, new GConstraint(GConstraint.height(50)));
```

Old fashion ...

```
pan.setConstraint(GButton.class, "height 50");
```

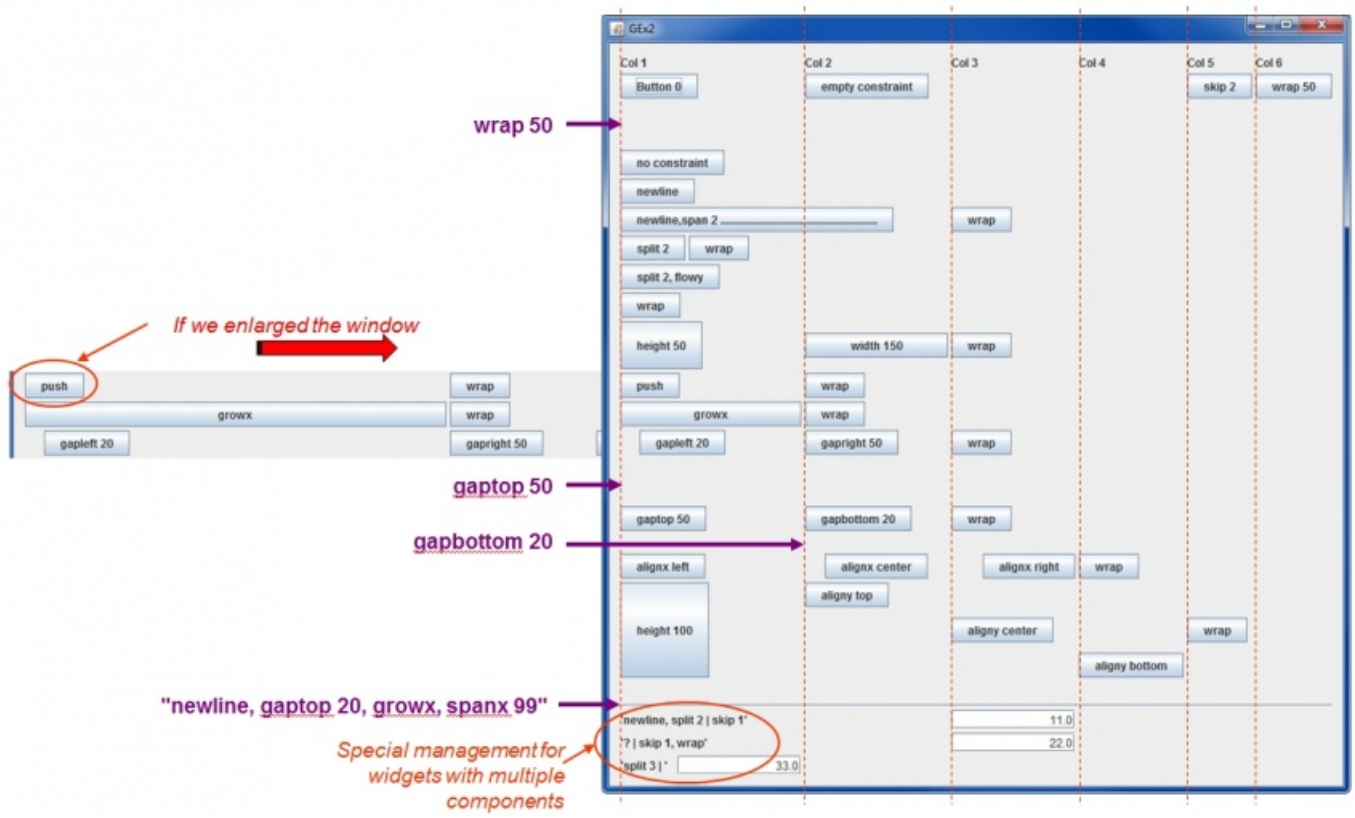
Some available « constraints »

The following table gives an overview of some basic MigLayout constraints. To get the exact API of the GBasicConstraintInterface methods proposed by GConstraint class, see the Java doc.

wrap [gapsize]	Go to the next line after the component (gapsize => amount of pixel after it)
newline [gapsize]	Go to the next line before the component (gapsize => amount of pixel before it)
skip [count]	Skip one or several columns (depending of the value of count, 1 by default)
span [countx [county]], **spanx** [countx], **spany** [county]	Allows to the component to spread on several cells (countx for horizontal axis and county for vertical axis)
split [count]	Allows to put several components on a single cell
flowx, **flowy**	Direction when a component is added (flowx by default)
height, **width** size	Specify the height (resp. width) of the component in pixel (preferred size)
push (**pushx**, **pushy**)	« Push » the next components (visible when the main window is enlarged)
grow (**growx'**, **growy**)	Fill the cell with the component
gap left [right] [top] [bottom], **gaptop**, **gapbottom**, **gapleft**, **gapright** [gap]	Specify the gap (in pixels by default)
align [alignx, aligny], **alignx**, **aligny** [align]	Specifiy alignment: (left, center, right) for alignx and (top, center, bottom) for aligny

For more constraints, one may read the MigLayout user manual.

On the following window, we can see several examples of such use of these layout constraints ...



We can see that the management of a GEntryReal (for example) may be more confuse than for a simple GButton. Indeed, this widget is composed of several other subwidgets :

```
=> subwidget 0: GLabelWithIndicator
   => subwidget 0.0: GLabel
   => subwidget 0.1: GIndicator (the "*" when the value is modified)
=> subwidget 1: GTextField
=> subwidget 2: GUnit
```

So, with the old fashion, it was possible (more or less easily) to access to such subwidgets using "|", "?", "+" syntax. We will not give more details about it here.

With the new API, it is easier to explain it with the setInnerDescendantContraint() method:

For example, if we want:

1. to apply "newline" to the global widget
2. to set 200 pixels for the textfield width
3. to skip a cell to the Gunit field

the solution is this one :

```
GEntryReal real = new GEntryReal(...);

// Applying "newline" to the GLabel
real.setInnerDescendantConstraint(new GConstraint(GConstraint.newline(), 0,
0);
```

```
// Applying "width 200" to the texfield
real.setInnerDescendantConstraint(new GConstraint(GConstraint.width(200)),
1);
// Applying "skip"to the texfield
real.setInnerDescendantConstraint(new GConstraint(GConstraint.skip(1)), 2);
```

Return to the introduction ↑ Go to the next page →

Menu de navigation

Outils personnels

- 3.138.118.250
- Discussion avec cette adresse IP
- Créer un compte
- Se connecter

Espaces de noms

- Page
- Discussion

Variantes

Affichages

- Lire
- Voir le texte source
- Historique
- Exporter en PDF

Plus

Rechercher

Rechercher Lire

GENIUS

- Welcome

- Quick Start
- News

Basic principles

- GFrame and GPanel
- Main widgets
- Links with Swing
- GLayout
- Conditional Display
- GListener interface

More deeper in the concept

- Units management
- GContainer
- GReadWrite interface and data files management
- Modified data
- Process management

Still more ...

- Validity controls
- Menu bar
- Icons
- GClear interface

Still more again ...

- Tooltips
- Shortcuts
- Copy & paste
- Plots
- Results File Management
- GPlotPanel
- GGroundPlotPanel
- Internationalization
- Log file
- Update data

Some other widgets

- GTabbedPane
- GTable1D
- GTable2D
- GComponentList

- GDialog and GDetachedPanel
- GContextFileManagement
- How to build a standard application
- GPanTest
- Create your own widget

Evolutions

- Main differences between V1.11.4 and V1.12.1
- Main differences between V1.10.1 and V1.11.4
- Main differences between V1.10 and V1.10.1
- Main differences between V1.9.1 and V1.10
- Main differences between V1.9 and V1.9.1
- Main differences between V1.8 and V1.9
- Main differences between V1.7 and V1.8
- Main differences between V1.6.2 and V1.7
- Main differences between V1.6.1 and V1.6.2
- Main differences between V1.6 and V1.6.1
- Main differences between V1.5 and V1.6
- Main differences between V1.4.1 and V1.5
- Main differences between V1.3 and V1.4.1

Training

- Training slides
- Tutorials package for V1.12.1
- Tutorials package for V1.11.4
- Tutorials package for V1.10.1
- Tutorials package for V1.9.1
- Training & tutorials package for V1.8
- Training & tutorials package for V1.7
- Training & tutorials package for V1.6

Links

- CNES freeware server

Outils

- Pages liées
- Suivi des pages liées
- Pages spéciales
- Adresse de cette version
- Information sur la page
- Citer cette page

- Dernière modification de cette page le 10 juillet 2017 à 07:52.

- Politique de confidentialité
- À propos de Wiki
- Avertissements
-