

# Process management

De Wiki

Aller à : [navigation](#), [rechercher](#)  
[Process management](#)

## Sommaire

- [1 The Command Launcher](#)
- [2 Pre-process \(before\)](#)
- [3 Post-process \(after\)](#)
- [4 GProgressBar](#)

## The Command Launcher

It is good to have a **GUI** ... but it has to be useful ! And most of the time, it is used to launch a computation program. Several solutions are available:

- Launch a Java thread ... but it could not be stopped **asynchronously** (`stop()` method is deprecated) except by stopping the **GUI** !!!
- Launch an executable independent of the **GUI**

**GENIUS** makes available classes [GJavaCommandLauncher](#), [GCommandLauncher](#), [GExecButton](#) and [GExecMenuItem](#). They will launch:

- Either a Java class, if it owns a « main » static method
- Either an executable (for example issued from a Fortran compilation)

A consequence is that entry data will only be passed by files.

First, we will have to instantiate a [GJavaCommandLauncher](#) or a [GCommandLauncher](#). Here is an example launching a Java executable class:

```
GJavaCommandLauncher cmd;  
String path = System.getProperty("java.class.path");  
cmd = new GJavaCommandLauncher ( new String[] {"myClass", "args ..."}, path,  
                                "Start computation", "Stop computation",  
                                null);  
// To assign the process output towards the standard output  
cmd.setCopyOutputToStdout(true);
```

Issued from this **cmd** object, we could get the associated launch button and item menu:

```
GExecButton butExec = cmd.getGExecButton();  
GExecMenuItem itemExec = cmd.getGExecMenuItem();
```

## Pre-process (before)

Considering we implement the GListener interface, we may execute some action inside the [before\(\)](#) method (for example, testing that all input data are OK):

```
public void before(GEvent e) throws GFileManipulatorException {

    if ( e.contains(butExec,itemExec) ) { // Launch button or menu have been
activated

        if ( ! cmd.isRunning() ) { // Program is not yet running, so we may
initialize it
            if ( valeursOK ) { // Flag indicating values are OK (for example)
                GFileManipulation.writeConfig("data.xml", "MAN", objetIhm); // We
prepare the input data file
            }
            else {
                cmd.setInhibited(true); // Finally, we do not launch the process !
            }
        }

    }

}
```

## Post-process (after)

Once, the executable has stoppped, we may also execute some other actions. But be careful of the fact that:

- we will pass through the `//after()` method after clicking on the [GExecButton](#) (or the [GExecMenuItem](#)) ... that does not mean that the process is finished. So, we will use the [getFinalnalSource\(\)](#) method (as explained in [GListener interface section](#)).
- the process may have stoppped nominally or because we stopped it before its end.

The example below shows how to manage this situation:

```
public void after(GEvent e) {

    if ( e.getFinalSource() == cmd ) {

        // We actually launched the application
        if ( cmd.getProcessStatus() == ProcessStatus.FINISHED_NORMALLY ) {
            System.out.println("Computation nominally finished ...");
        }
        else if ( cmd.getProcessStatus() == ProcessStatus.FINISHED_BY_USER ) {
            System.out.println("Computation stopped by user ...");
        }
    }

}
```

```
}  
  
}
```

## GProgressBar

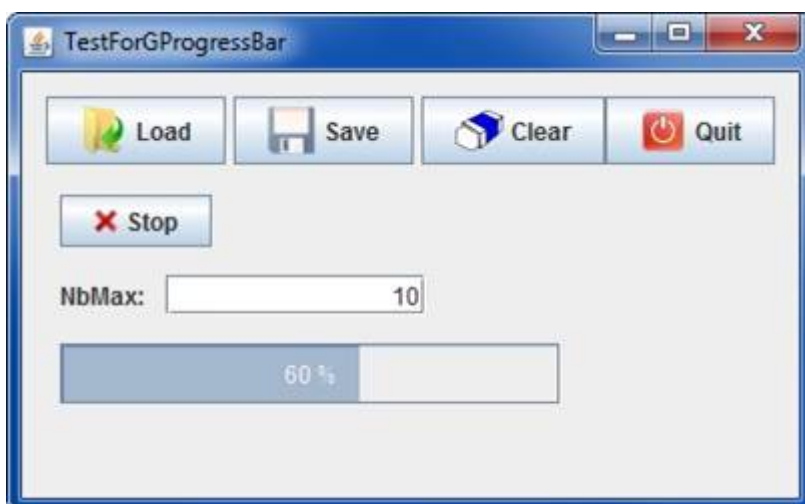
Another interesting widget is [GProgressBar](#) to know the evolution of the computation process.

```
// Getting the progress bar from the cmd object  
GProgressBar pBar = cmd.getGProgressBar();  
  
// Calling corresponding swing object to resize it  
pBar.getJProgressBar().setPreferredSize(new Dimension(250,30));  
  
...  
  
public void generic() throws GException {  
    put(cmd);  
    ...  
    put(pBar); // Display as any other widget  
}  
  
...  
  
public void display() throws GException { generic(); }  
  
...
```

The computation program will send the information as this:

- Send on the standard output "[@value@n](#)" with value between 0 and 100
- Or use [GEnvironment.sendProgress\(value\)](#) with value between 0 and 100

It is possible to force the value with the [setValue\(val\)](#) method.



[Return to the introduction](#) ↑ [Go to the next page](#) →

Récupérée de « [http://genius.cnes.fr/index.php?title=Process\\_management&oldid=410](http://genius.cnes.fr/index.php?title=Process_management&oldid=410) »

## Menu de navigation

### Outils personnels

- [3.14.15.94](#)
- [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

<input type="text"/>	<input type="button" value="Rechercher"/>	<input type="button" value="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 07:56.
- [Politique de confidentialité](#)
- [À propos de Wiki](#)
- [Avertissements](#)

