

Getting Started with jGRASP for Java

Maria Litvin

Phillips Academy, Andover, Massachusetts

Gary Litvin

Skylight Publishing

1. [Introduction](#)
2. [Downloading and Installing jGRASP](#)
3. [Creating a Project and Running “Hello World”](#)
4. [Bringing Existing Java Files into jGRASP](#)
5. [Command-Line Arguments and User Input](#)
6. [Running GUI Applications](#)
7. [Workspaces](#)
8. [Using Jar Files](#)
9. [Creating Jar Files](#)

1. Introduction

This document serves as a brief introduction to jGRASP. See more detailed information [on their web site](#).

jGRASP was developed by the Department of Computer Science and Software Engineering in the Samuel Ginn College of Engineering at Auburn University. The latest version of jGRASP, as of June 2018, is Release 2.0.4_04. jGRASP is a free download.

According to www.jgrasp.org,

jGRASP is a lightweight development environment, created specifically to provide automatic generation of software visualizations to improve the comprehensibility of software. jGRASP is implemented in Java, and runs on all platforms with a Java Virtual Machine (Java version 1.5 or higher).

jGRASP not only enables the user to edit and run Java programs, it can also produce what its developers call Control Structure Diagrams (CSDs) for Java (and other programming languages). Our primary concern here is how to use jGRASP as an IDE (Integrated Development Environment) for editing and running Java programs.

2. Downloading and Installing jGRASP

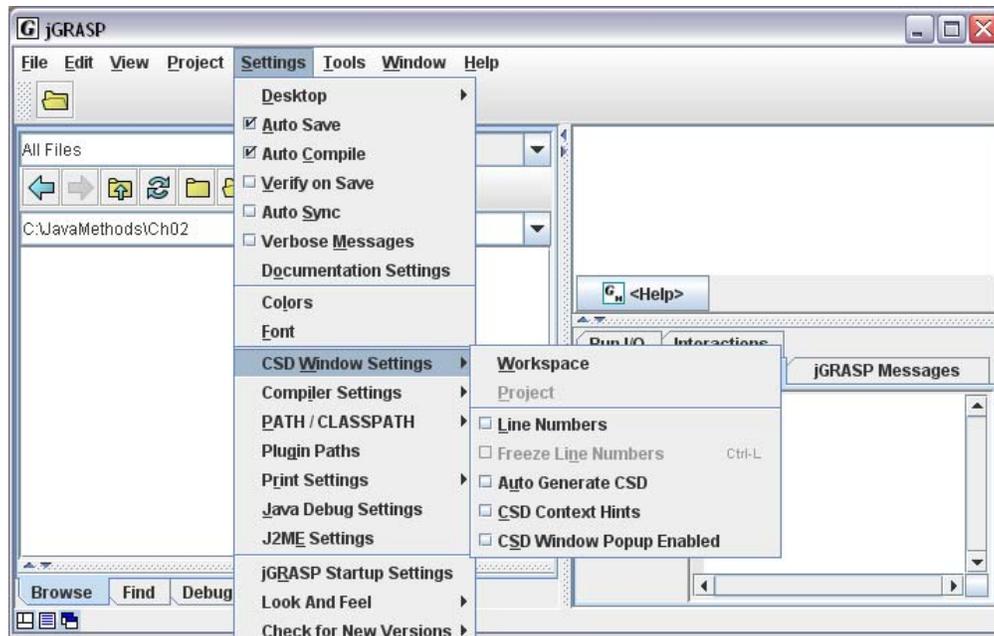
First make sure the Java Development Kit (JDK) is already installed on your computer. See www.skylit.com/javamethods/faqs/GettingStartedJava.pdf for directions.

Go to www.jgrasp.org and click **Download** in the left column, then click on the green button that corresponds to your operating system. Run the downloaded installer file and follow the instructions. Under *Windows*, jGRASP will be installed by default in the `C:\Program Files (x86)\jGRASP` folder. The executable jGRASP file `jgrasp.exe` will be located in `C:\Program Files (x86)\jGRASP\bin` and a shortcut to it will be placed on your desktop. (If the shortcut does not show up or is accidentally deleted, you can always create it by dragging `jgrasp.exe` to the desktop while holding down `Ctrl+Shift` or `Alt`.)

To run jGRASP, double-click on `jgrasp.exe` or on the shortcut.

The **Settings** menu provides a short list of options for jGRASP.

A novice might want to uncheck “Auto Sync” and uncheck all boxes under “CSD Window Settings”:

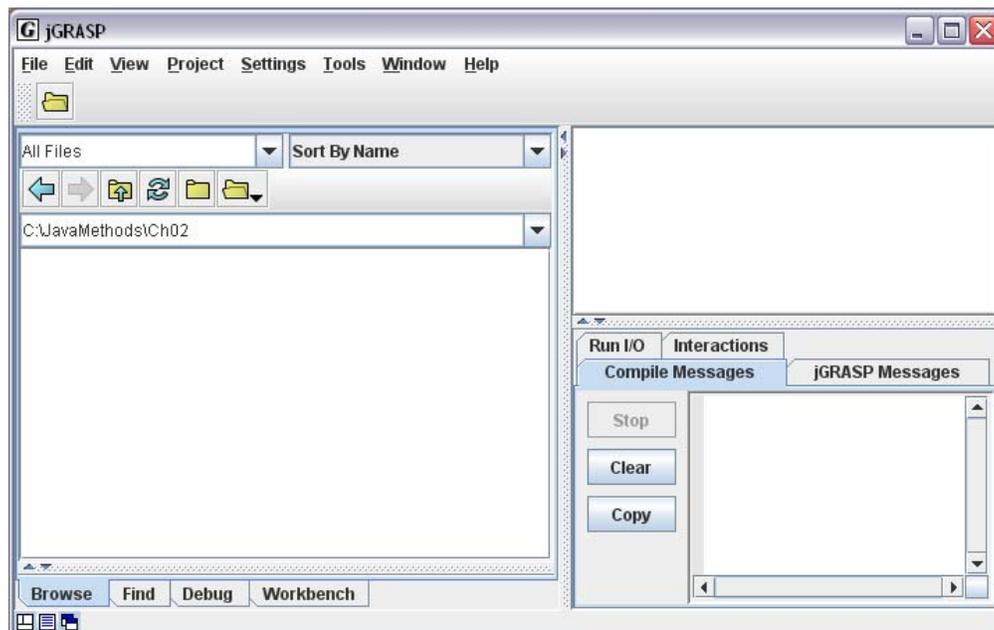


You might want to change the “Look and Feel” setting to match your operating system (for example, “Windows” for *Windows*).

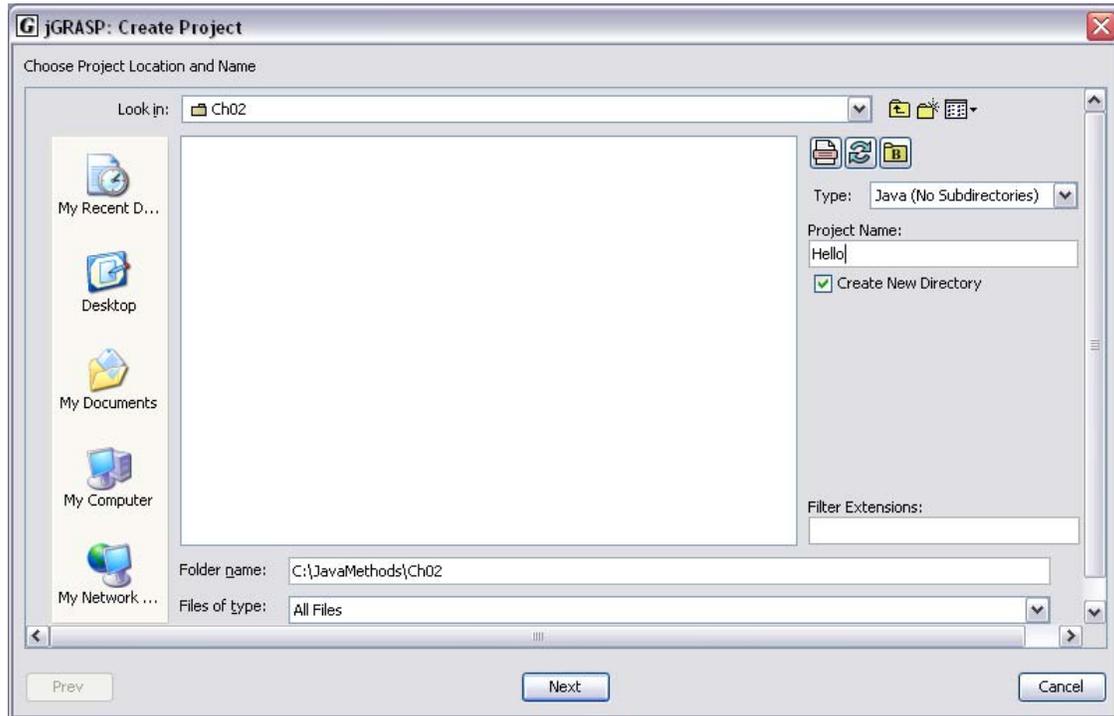
You might also want to change the tab size to 2 and other options under “Settings/CSD Window Settings/File.”

3. Creating a Project and Running “Hello World”

jGRASP window is split into three panes. The left pane under the “Browse” tab allows you to browse folders and files on your computer. Use the  button to create a new folder, for example, [JavaMethods\Ch02](#):



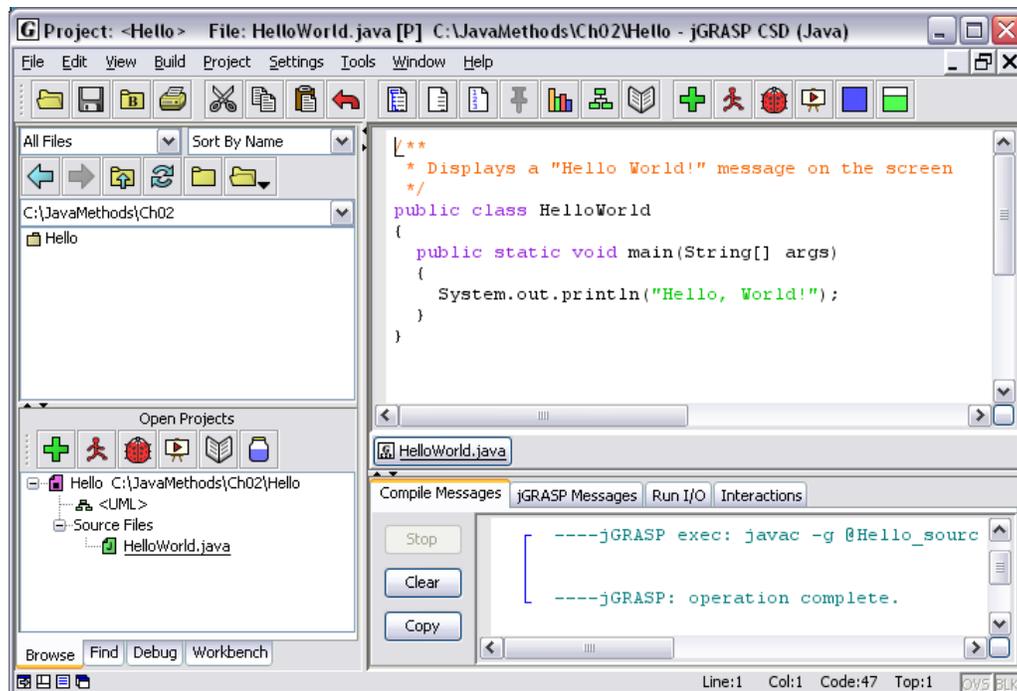
Navigate to the folder in which you want to place your new project, and under the [Project](#) menu select [New](#). In the dialog box that pops up enter the project name, for example, “Hello”:



Check the “Create new Folder” box so that jGRASP places all project files in a separate folder (with the same name as the project).

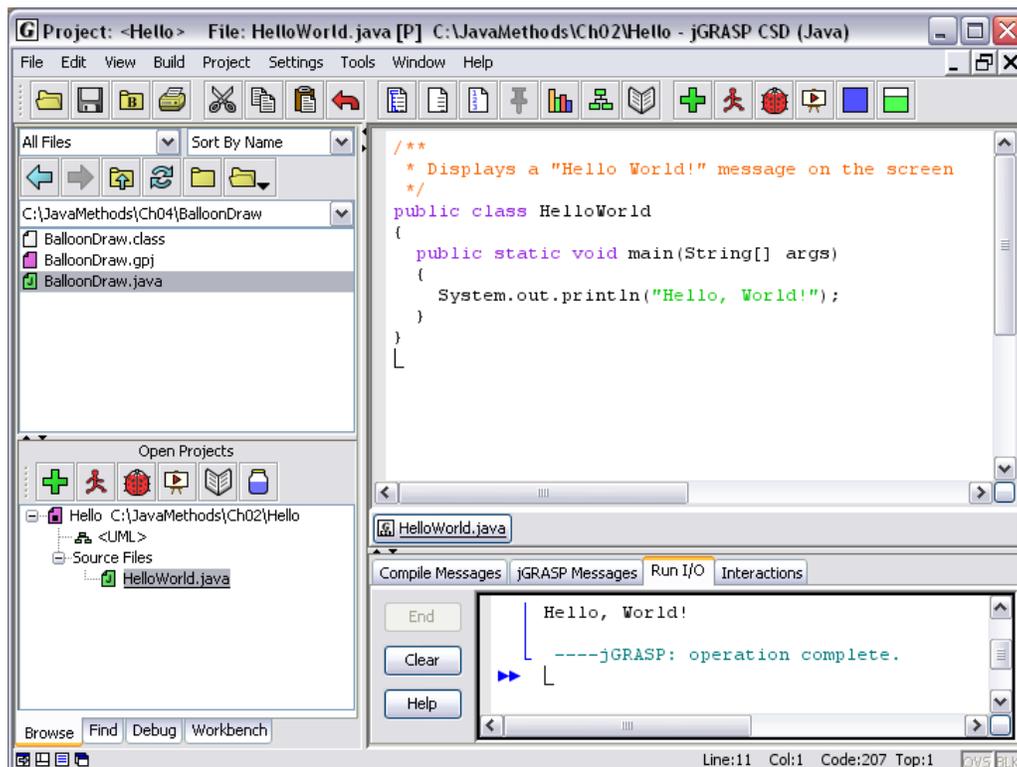
Click [Next](#). jGRASP will come up with another dialog box. It will show that jGRASP has created a file with the [.gpj](#) extension, which describes the project (for example, [Hello.gpj](#)) and placed it in the currently selected folder. If you want to add files to the project right away, check the “Add Files to Project Now” box. Otherwise, the project will remain empty. Click [Create](#).

To create a new Java file in the project, choose [New](#) on the [File](#) menu, then [Java](#). Type in the text for your file in the editor window (or paste it from a different source). For example:



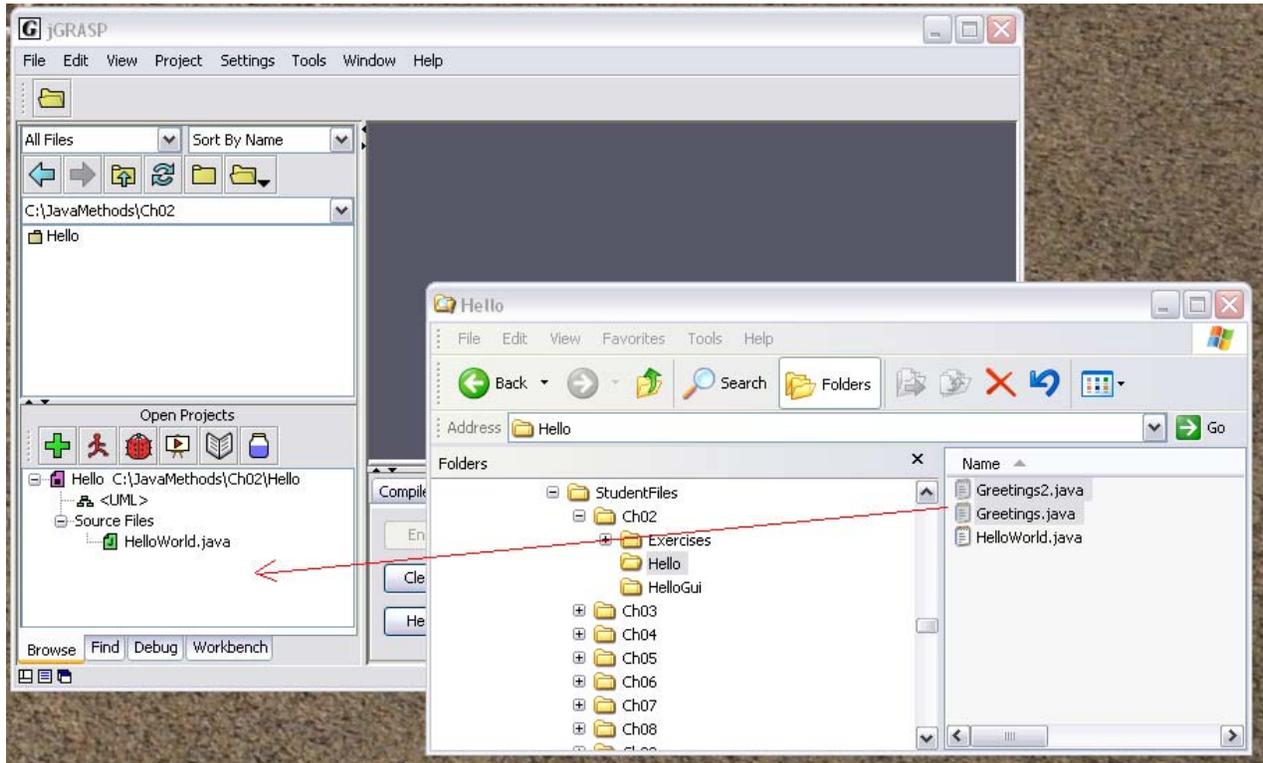
Choose **Save** on the **File** menu (or press **Ctrl+S**). jGRASP is smart enough to propose the name for a Java file that matches the name of the class in the file, with the extension `.java`.

To run a program, make sure its “main” class (the class with the `main` method) is open in the editor window and choose **Run** on the **Build** menu, or press **Ctrl+R**, or click the “Run” button  on the toolbar. The console output will appear in the “Messages” pane under the “Run I/O” tab:



4. Bringing Existing Java Files into jGRASP

To add files to a project after it has been created, drag and drop the files from the file manager on your computer to the jGRASP's project window:



Or, choose [Add Files](#) on the [Project](#) menu, navigate to the file(s) you wish to add, and click [Add](#).

Make sure you choose “Copy” on the dialog box that pops up; choosing “Move” will relocate the original file(s), and choosing “Link” may modify the original file(s).

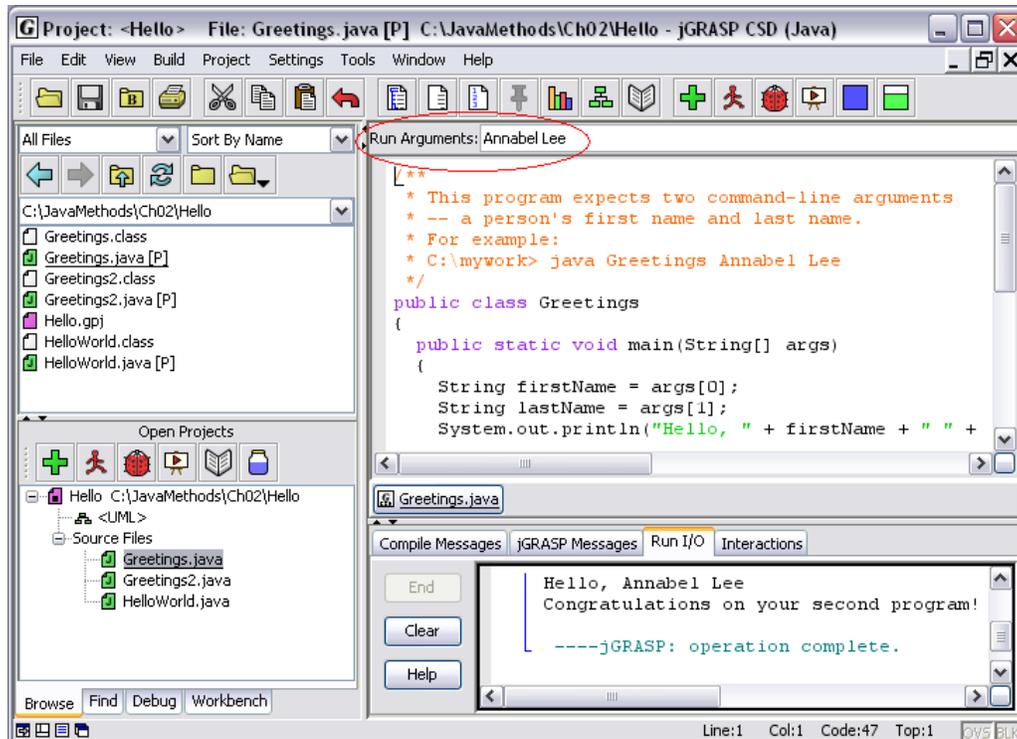
To open the file in the editor, click on a Java file in the “Browse” or “Project” window. To close a file, click on the “cross” button in the upper-right corner (or right-click on the corresponding tab and chose “Close”).

Note that the [Generate CSD](#) command on the [View](#) menu (or pressing [F2](#)) temporarily changes the indentation step to three spaces when the CSD is removed. Closing and reopening the window restores the indentation step to two spaces.

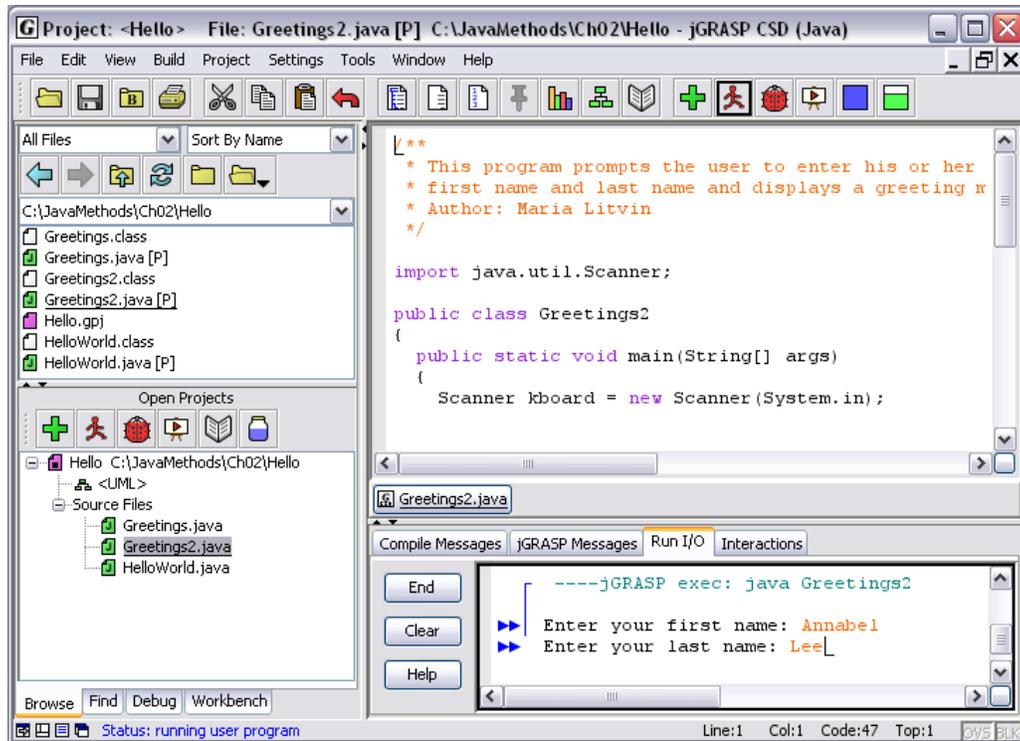
5. Command-Line Arguments and User Input

The `Greetings` program (*Java Methods* Section 2.4) expects command-line arguments; `Greetings2` accepts input from the user.

To run a program that takes command-line arguments, check `Run Arguments` on the `Build` menu. A text entry field will appear where you can enter the arguments:

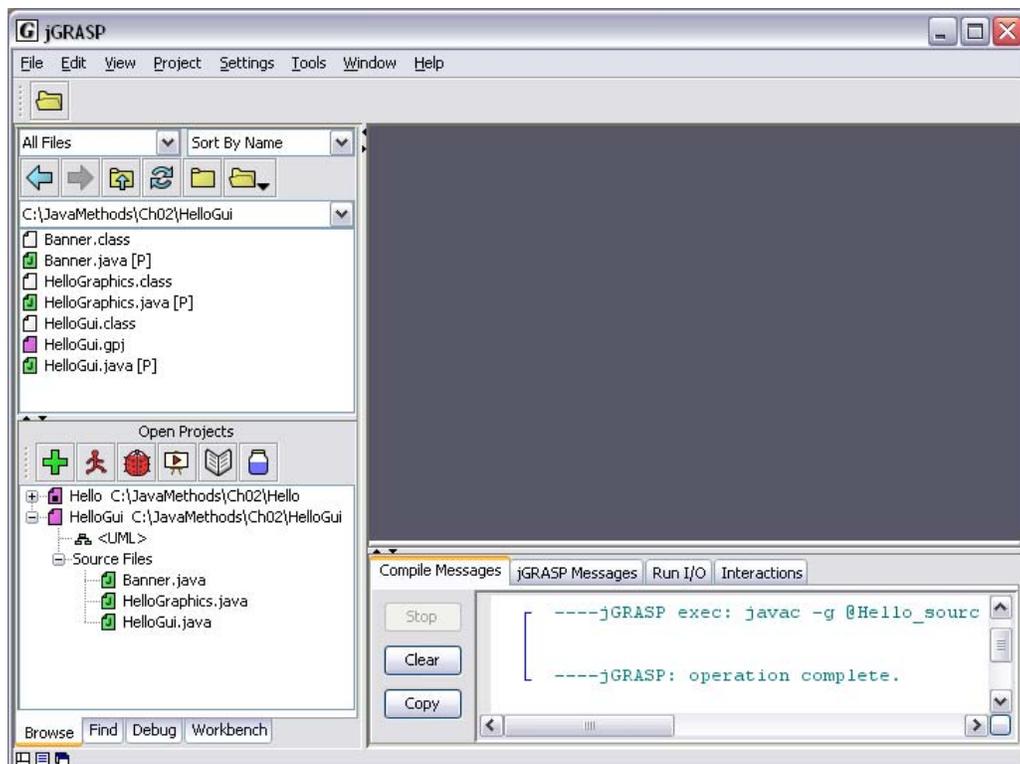


The `Greetings2` program prompts the user for input. Predictably the console dialog appears on the “Messages” window under the `Run I/O` tab. For example:



6. Running GUI Applications

Create another project (from [Project](#) menu choose [New](#)). For example, create a new project named “HelloGui” and copy all the files from the *Java Methods* Chapter 2 [HelloGui](#) folder to the project’s folder:



Click on any Java file to open it in the editor, then press **CTRL+R** to run it.

You can also click on the “Run” button  on the “Open Projects” toolbar and select a program you wish to run.

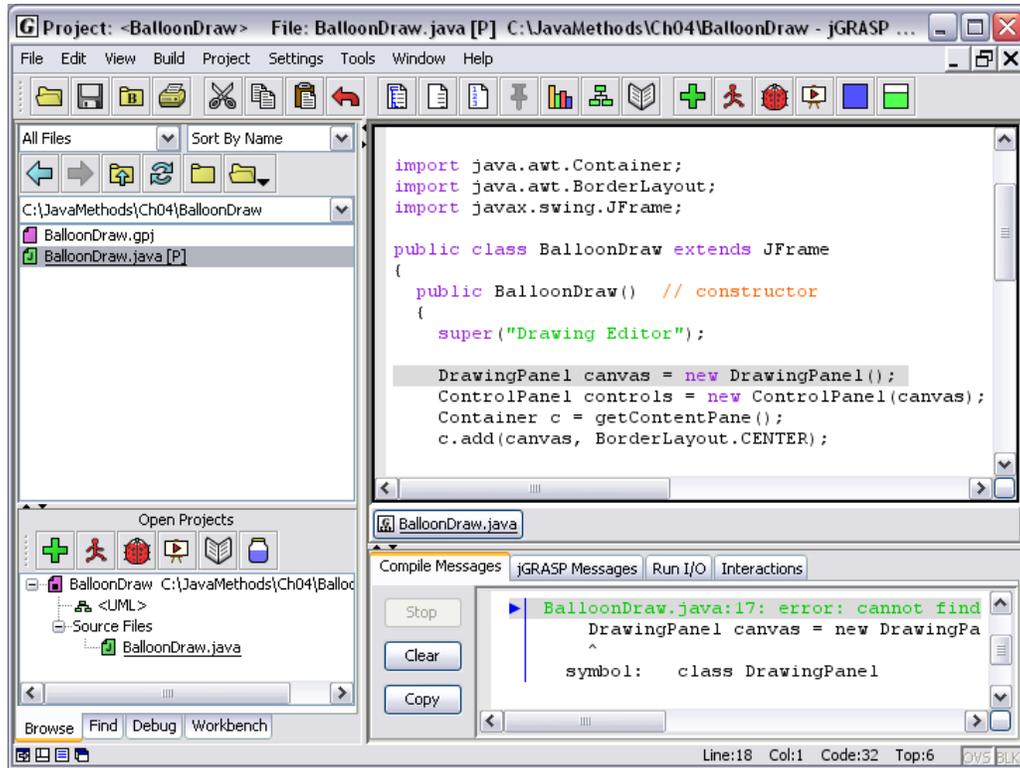
7. Workspaces

A “workspace” in jGRASP is a logical concept. Initially projects are associated with the default workspace. You can create your own workspace and add projects to it. The **File** menu has an entry **Workspace**, which has commands to choose, create, or edit a workspace. A workspace description is stored in a `.gws` file in the `.grasp_settings/workspaces` folder somewhere in the user’s system data area.

8. Using Jar Files

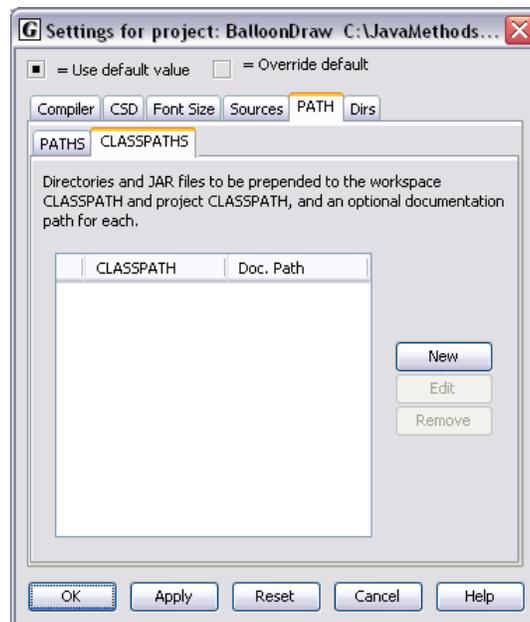
A jar (Java archive) file can hold several compiled Java classes and serve as a library for a Java project. The file name for a jar file has the extension `.jar`. Standard Java library supplied with the JDK is a jar file, and it is automatically included into Java projects. A programmer or a third party can supply their own jar files. For example, the authors of the *Java Methods* textbook supply `EasyClasses.jar`, which holds the class files for `EasyReader`, `EasyWriter`, `EasySound`, and `EasyDate` classes (see [Appendix D](#)).

As an example, let’s set up a project for the *BalloonDraw* program (Java Methods, Section 4.2), which uses the `balloondraw.jar` library. Create a new project and add `BalloonDraw.java` to it. An attempt to run the program generates a list of errors because several required classes from `balloondraw.jar` are missing:

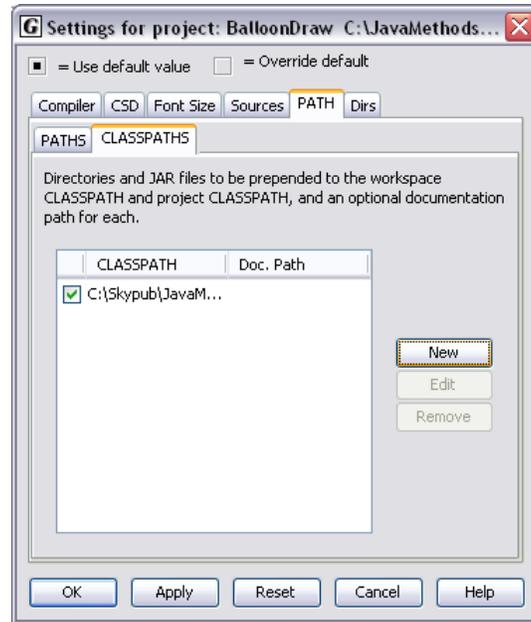


To add a jar library to a project, follow these steps:

1. On the [Settings](#) menu choose [PATH / CLASSPATH](#) ⇒ [Project](#).
2. Select [PATH](#) and “[CLASSPATHS](#)” tabs and click [New](#):



3. Navigate to the jar file you need to add to the project and click [Choose](#).



Click [OK](#).

If a jar file includes a class that is explicitly added to a project, the class in the project takes precedence over the class in the jar.

For example, we can add [Balloon.java](#) to the project and edit it. [Balloon.class](#) in the project will take precedence over [Balloon.class](#) in [balloondraw.jar](#).

9. Creating Jar Files

A jar file can hold a complete Java program (a class that has the [main](#) method and all the supporting classes) and it can be configured to be runnable: a double click on the file will run the program. Try, for example, [balloondraw.jar](#) in the *Java Methods* student files, [Ch04/BalloonDraw](#).

The same jar file can be runnable and can also serve as a library.

jCRASP allows you to quickly create a library or an executable jar file for a project. From the [Project](#) menu choose [Create JAR or Zip File For Project](#). On the window that pops up, select the type of the jar or zip file, the classes that go into it, and its destination. See jGRASP's Help and tutorials for details.