How to Use JCreator LE

Maria Litvin
Phillips Academy, Andover, Massachusetts

Gary Litvin
Skylight Publishing

1. Downloading and Installing *JCreator*

JCreator ™ is a trademark of Xinox Software. *JCreator* LE is a simple Java IDE for Windows. The LE version used to be free, but now Xinox charges a license fee for Version 5. Still you can download the 4.5 freeware version from several web sites.

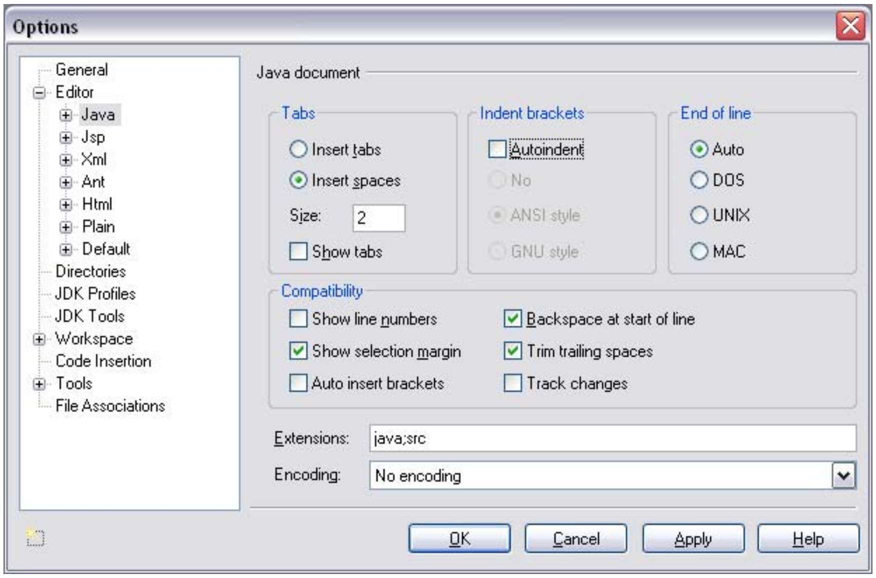
JCreator runs on *Windows* machines and requires Oracle's Java JDK (Java Development Kit) installed — see Getting Started with Java.

Install the JDK and documentation first, then *JCreator*.

2. Configuring Default Properties

Make sure there is no workspace open (choose `File/Close Workspace` if one is open). From the `Configure` menu, choose `Options...` In the dialog box that pops up expand the `Editor` node and click on "Java."

Set the tab size to 2 and select the "Insert spaces" option. You may find it more convenient at the beginning to uncheck the "Show line numbers" and "Autoindent" options:



Java document

Tabs
○ Insert tabs
● Insert spaces
Size: 2
□ Show tabs

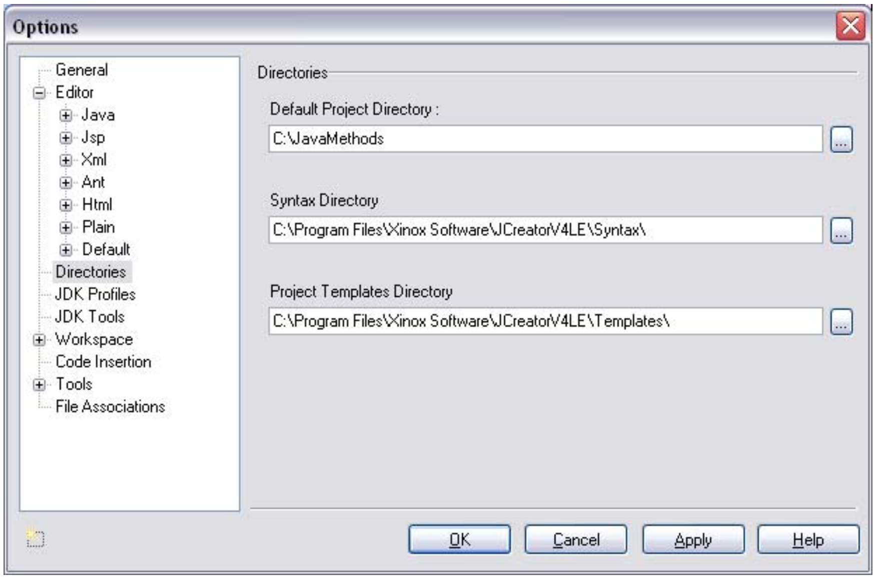
Indent brackets
□ Autoindent
○ No
● ANSI style
○ GNU style

End of line
● Auto
○ DOS
○ UNIX
○ MAC

Compatibility
□ Show line numbers
☑ Show selection margin
□ Auto insert brackets
☑ Backspace at start of line
☑ Trim trailing spaces
□ Track changes

Extensions: java;src
Encoding: No encoding

Click on the "Directories" node and enter the name of the folder where you will keep your projects, for example `C:\JavaMethods`, in the "Default Project Directory" line:



Directories

Default Project Directory : C:\JavaMethods

Syntax Directory: C:\Program Files\Xinox Software\JCreatorV4LE\Syntax\

Project Templates Directory: C:\Program Files\Xinox Software\JCreatorV4LE\Templates\

Click on the “Code Insertion” node and check the “Start open brace on new line” box.

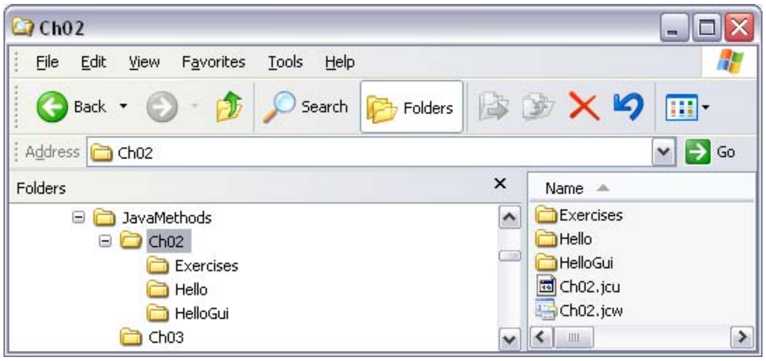
Then click OK.

3. Setting up Workspaces and Projects

JCreator organizes your programs into workspaces and projects. A workspace contains one or several projects. A project contains one or several programs. For example, you can have a workspace for all the labs and exercises in one chapter of a textbook, and a project for a particular lab or set of exercises. You give names to your workspaces and projects.

Workspaces and projects are akin to folders and subfolders. In fact, it is customary to keep all the files for a given project in the same folder. The name of the project folder is usually the same as the name of the project. When you create a project, *JCreator* creates a folder for it, unless you choose an existing folder.

JCreator keeps the description of a workspace in a file with the extension `.jcw`. The projects within the workspace can be scattered on disk, but it is customary to group them as subfolders in the same folder. The `.jcw` file can be placed in that folder, too. For example:



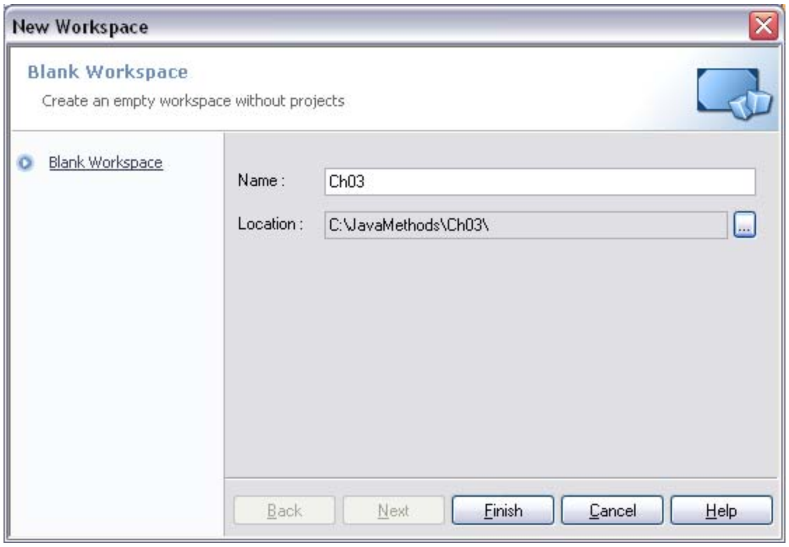
Address: Ch02

JavaMethods
Ch02
Exercises
Hello
HelloGui
Ch03

Exercises
Hello
HelloGui
Ch02.jcu
Ch02.jcw

Here the folder `Ch02` contains the workspace file `Ch02.jcw` and three project folders for this workspace: `Exercises`, `Hello`, and `HelloGui`.

To create an empty workspace, first create a folder for your workspace using *Windows Explorer* (for example, C:/JavaMethods/Ch03). Then open or return to *JCreator*, click on File/New, and choose Blank Workspace.... In the dialog box that pops up, navigate to the desired location for the workspace files, for example, C:\JavaMethods\Ch03, and enter the name for your workspace (for example, Ch03):



Name : Ch03

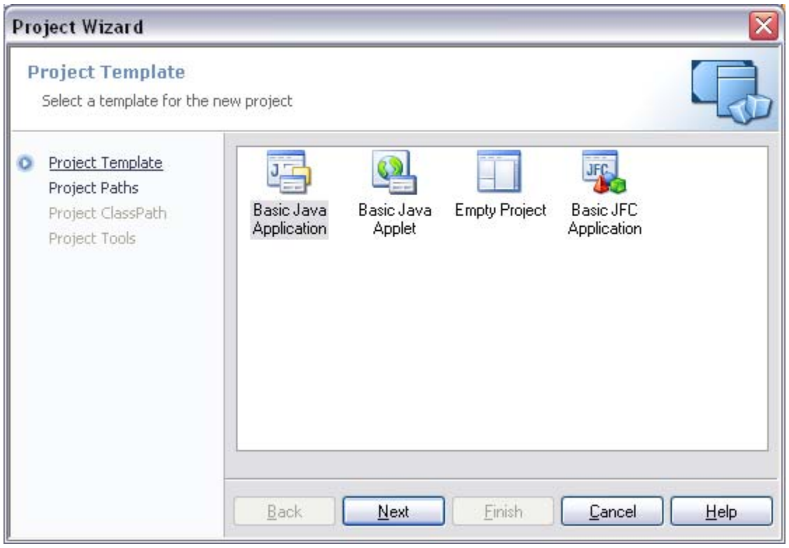
Location : C:\JavaMethods\Ch03\

Click Finish.

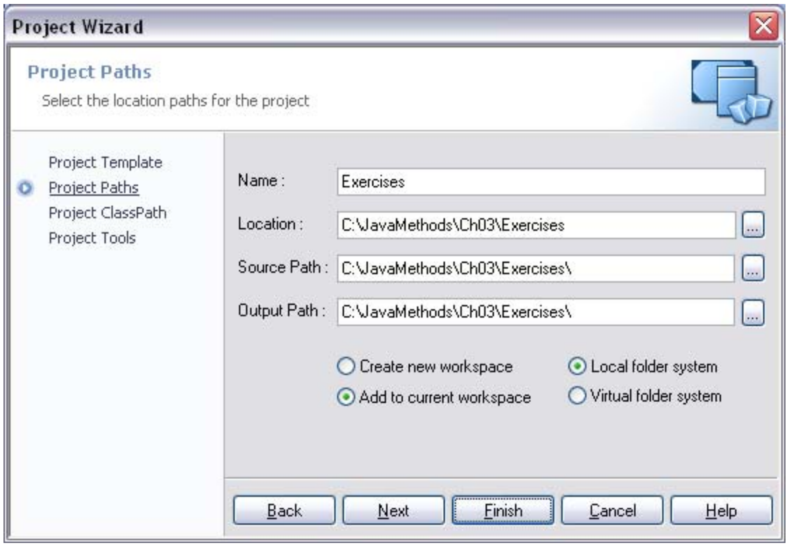
You can open an existing workspace by going to File/Open Workspace....

JCreator keeps track of recently opened workspaces and allows you to quickly open one of them and load all its projects. To do this, choose File/Recent Workspaces.

To add a new project to the currently open workspace, go to Project/New Project... or File/New and choose Project.... Click on Empty Project, then Next:



In the dialog box that pops up, enter the project name (for example, Exercises):



Project Wizard

Project Paths

Select the location paths for the project

Name : Exercises

Location : C:\JavaMethods\Ch03\Exercises

Source Path : C:\JavaMethods\Ch03\Exercises\

Output Path : C:\JavaMethods\Ch03\Exercises\

○ Create new workspace

● Add to current workspace

● Local folder system

○ Virtual folder system

JCreator automatically adds the project name to the location lines. Click `Finish` (twice).

To delete a file from a project, click on that file in the file list and press `Delete`.

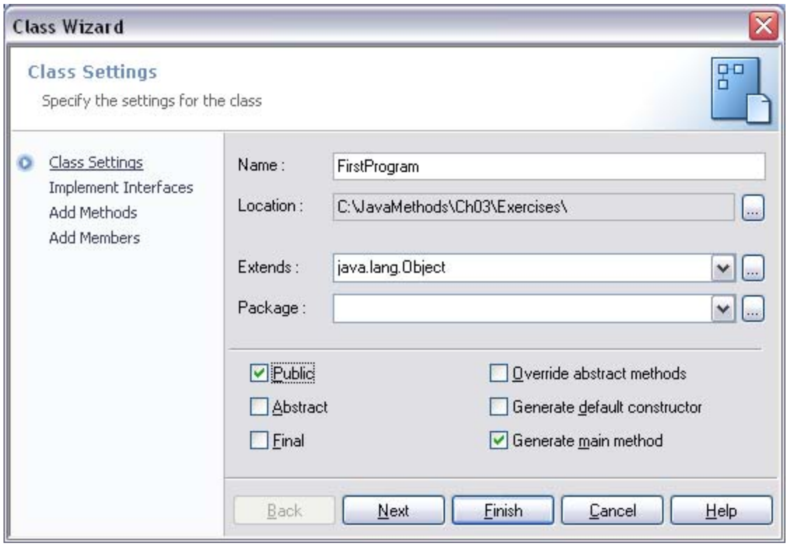
4. Entering and Running a Simple Program

Set up a workspace with a project in it, as described above. If the workspace has several projects, right-click on the project you want to work on and choose “Set as Active Project.”

You can add an existing Java source file to your project or create a new Java class (source file) from scratch. To add an existing file, go to `Project/Add files...`, navigate to the folder that holds the file, double-click on the file (or single click and click `Open`), then click on the `Add` button. The file will be copied into your project folder and added to the project description. You can add several files at once from the same folder by clicking on them while holding the `Ctrl` key down.

Adding files is a common operation, so you might want to put the `Add files` button to the toolbar (`Configure/Customize...`).

To create a new Java file, go to `Project/New class...` or click on the `New class` button on the toolbar. In the dialog box that pops up, enter the class name and check the desired options (such as “public,” “Generate main,” etc.) and click `Finish`. For example:



Name : FirstProgram

Location : C:\JavaMethods\Ch03\Exercises\

Extends : java.lang.Object

Package :

☑ Public

☐ Abstract

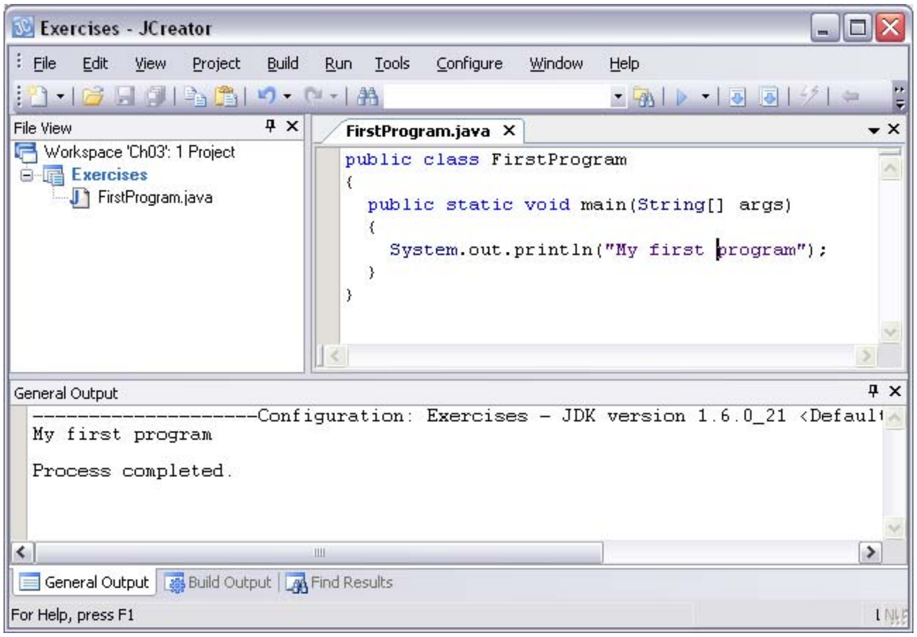
☐ Final

☐ Override abstract methods

☐ Generate default constructor

☑ Generate main method

Edit the text of your class. For example:



```
public class FirstProgram
{
  public static void main(String[] args)
  {
    System.out.println("My first program");
  }
}
```

```
--------------------Configuration: Exercises - JDK version 1.6.0_21 <Defaul
My first program

Process completed.
```

To edit a file, double-click on its name in the file list on the left. Several files from the same or different projects may be open at the same time. You can choose one of them for editing by clicking on the corresponding tab above the edit window.

To compile all the classes in the project, go to `Build/Compile Project` or press `F7`.

To run the program, from the `Run` menu choose `Run Project` or press `F5` or click on the `Run Project` button (a triangle pointing to the right).

Note: you must successfully compile the classes before you can run the program.

To redirect program's output from Command Prompt to *JCreator*'s general output window, go to Configure/Options... Select "JDK Tools" in the left column and "Run Application" in the pulldown list, click <Default>, click the Edit... button, check the "Capture output" box, click OK twice.

***JCreator* provides Java API help. To access API press Ctrl-F1.**

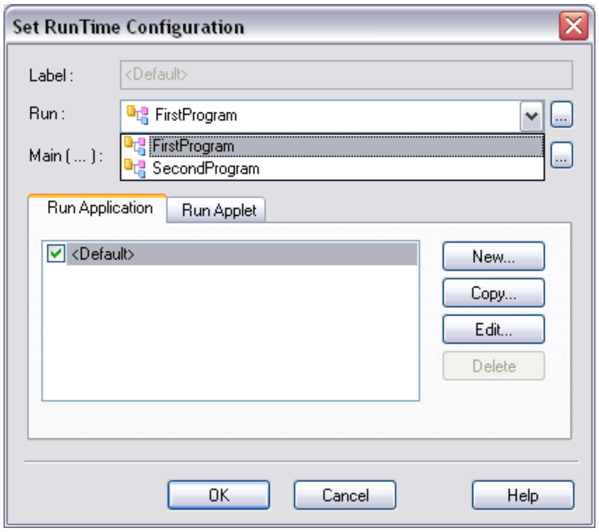
5. Choosing Which Program to Run

JCreator runs the program from the active project. If the workspace has several projects, right-click on the project you want to work on and choose "Set as Active Project."

Your project can hold several executable programs. You can choose which one to run in several ways:

1. Open the class you want to run in the editor (or make it active by clicking on its tab), then from the Run menu choose Run File. (The Run File button can be added to the toolbar.)

2. Click on the pulldown arrow next to the Run Project button and choose Configuration.... Click Default, click Edit, and pick the default class to run from the pulldown list. For example:



Set RunTime Configuration

Label : <Default>

Run : FirstProgram

FirstProgram
SecondProgram

Main (...) :

☑ <Default>

Click OK twice. The chosen class becomes the default executable for F5, etc.

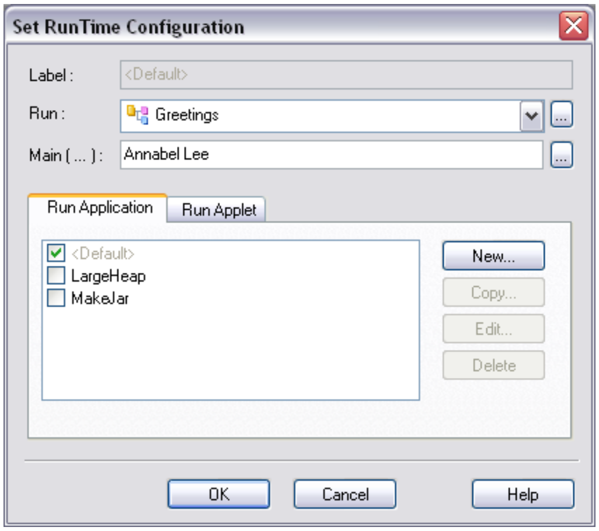
3. Create a new configuration. Click on the pulldown arrow next to the Run Project button and choose Configuration.... Click New, enter a nickname for your configuration and pick the class to run from the pulldown list.

6. Setting Up Command-Line Arguments

The Greetings program (*Java Methods* Section 2.4) expects command-line arguments. *JCreator* allows you to change the default run-time configuration for your program; in particular, it lets you set the command-line arguments.

Click on the pulldown arrow next to the `Run Project` button and choose `Configuration...`. Click `Default`, click `Edit`.

Another dialog box pops up. Enter the desired command-line arguments in the "Main(...)" field. For example:



Label : <Default>

Run : Greetings

Main (...) : Annabel Lee

☑ <Default>
□ LargeHeap
□ MakeJar

Click `OK` twice.

You can create and name a number of configurations with different run-time arguments. You can copy configurations and edit them.

JCreator also allows you to define a number of "global" configurations. To do that, go to `Configure/Options` menu and choose `JDK Tools`. The dialog box that pops up lets you copy the default configuration under a different name and edit it. In particular, you can create a configuration that prompts you for command-line arguments each time you run a program. See *JCreator*'s help (under "Contents/How to" look for "How to set the Main function arguments"). You can choose a "global" configuration you want to use for a particular project from the `Project/Project Settings...` menu.

7. Adding Jar Files to Projects

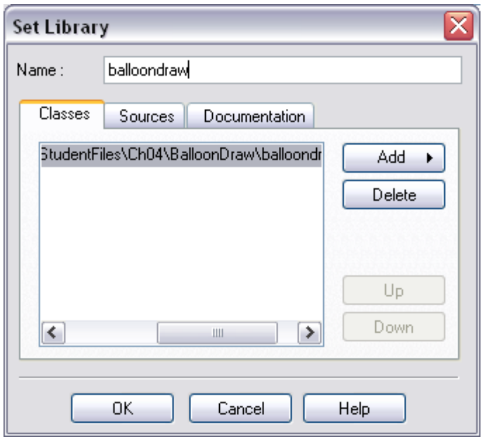
A jar (Java archive) file can hold several compiled Java classes and serve as a library for a Java project. The file name for a jar file has the extension `.jar`. Standard Java library supplied with the JDK is a jar file, and it is automatically included into Java projects. A programmer or a third party 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.

In *JCreator* you can simply add a jar file to the project. The procedure is the same as adding a Java file, except when you navigate to the folder where the jar file resides, choose "All files" from the pulldown list.

The above method, however, copies the jar file into the project folder. This is wasteful if the same jar file is used with many projects. There is a more economical way. *JCreator* allows you to define a set of jar files under a single name of your choice. Once a set is defined, you can add the whole set to any project.

Open a workspace with a project. To define a set of jar files, go to `Project/Project Settings...` and click on the "Required Libraries" tab. Click `New...` and in the dialog box that pops up enter a name of your choice for the set of jars, click on the `Classes` tab, and click on the `Add >` button. The submenu that pops-up gives you a choice of adding a "path" (a folder that contains `.class` files) or an "archive" (one or several jar files). Choose `Add Archive...`, navigate to the desired jar file(s), then click `OK`. The selected jar file(s) is (are) added to the list. For example:



Set Library

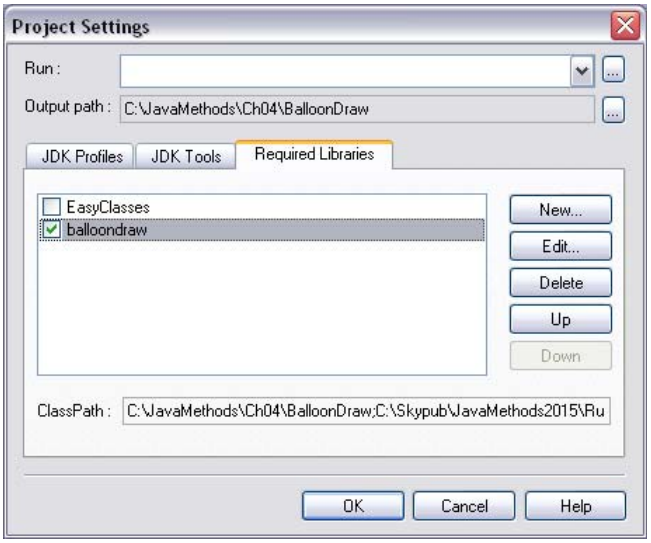
Name : balloondraw

Classes

StudentFiles\Ch04\BalloonDraw\balloondr

Repeat as necessary, then click OK.

Click on the checkbox next to the jar set to add these jars to the current project. For example:



Project Settings

Run :

Output path : C:\JavaMethods\Ch04\BalloonDraw

Required Libraries

□ EasyClasses
☑ balloondraw

ClassPath : C:\JavaMethods\Ch04\BalloonDraw;C:\Skypub\JavaMethods2015\Ru

Once a set of jars has been defined, JCreator remembers it and makes it available to any project in any workspace. Check the checkbox next to a set's name to add that set to the project.

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 takes precedence over `Balloon.class` in `balloondraw.jar`.