

Setup programming environment

Function of

```
/* Hello World Example */  
class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello World");  
    }  
}
```

Source code

Source code is the text that you write to tell the computer what to do. For this class, we write in the language Java. Source code is written in pure ASCII. That means that consists of only letters, numbers and some punctuation marks. It does not contain information about italics or bolding nor does it specify fonts, e.g. Times Roman or Arial. The example at the left is the “Hello World” program written. “Hello world” dates back to the beginning of the language C.

Compiler

A *compiler* is a program that translates human readable source code into machine or byte code.

Editor

An editor is very similar to a word processor. An editor allows you to type in a new program or to change an existing program. Most word processors can work as editors if you are careful about how you save your files. You must save them as text files not as word processing documents.

Most programming editors have extensive search and replace features that are helpful in programming. Newer programming editors also use color to highlight different parts of the code. The editor we will be using puts comments in green, reserved words in blue, strings in purple, etc. The colors are configurable so that you can easily change the color scheme without causing any problems.

Loader / Linker

The loader and the linker work together to run your compiled program. The loader puts your program into the Java Virtual Machine (JVM) where it is run. The JVM can be either on your main computer or on the RCX. You choose by specifying which loader you use. The linker gathers together all the pieces needed for your program to run. In the code example, you print by calling `System.out.println` with “hello world” as its argument. The linker finds the `println` method and bundles it with your code

before the loader actually runs the program. Historically, the linker was a separate program that made a standalone executable file and this process was called *static linking*. Java follows the modern practice and does not do the linking until the program starts running (and in some cases, it does the linking while the program is running). This is called *dynamic linking*.

IDE

IDE stands for Integrated Programming Environment. An IDE forces you to organize your programming. Typically an IDE contains an editor, a compiler, a debugger and a file manager. For Java, IDE's typically also have a form designer for making *graphical user interfaces*, *GUI's*, although the IDE we will be using does not because there is no GUI on a robot.

It is possible to program in Java and LeJOS without using an IDE. It is also possible to build a house without power tools. But just because something is possible does not mean it is a good idea. A good IDE enhances your ability to get things done.

Install

JDK

Go to java.sun.com. Download the current, non-beta version of J2SE for your platform. Install it following the installation instructions. If you want to run the compiler, or any of the other tools, in a command line box, you will need to add or update several environment variables. This book, however, uses an IDE so you will not need to update these variables. See Appendix xx for more information.

JDK documentation

Go to java.sun.com and download the documentation for the JDK from the same page where you downloaded the JDK itself. Run the installation program and put the documentation in the docs folder in your JDK directory.

LeJOS

Go to lejos.sourceforge.net. Download the current version. You will have to unzip the file to install it. If you don't have a zip utility (Windows XP has it built in), go to www.winzip.com. Download the free version of winzip and use it to unzip LeJOS. Put LeJOS into the directory \LeJOS.

Set environment variable for RCXTTY.

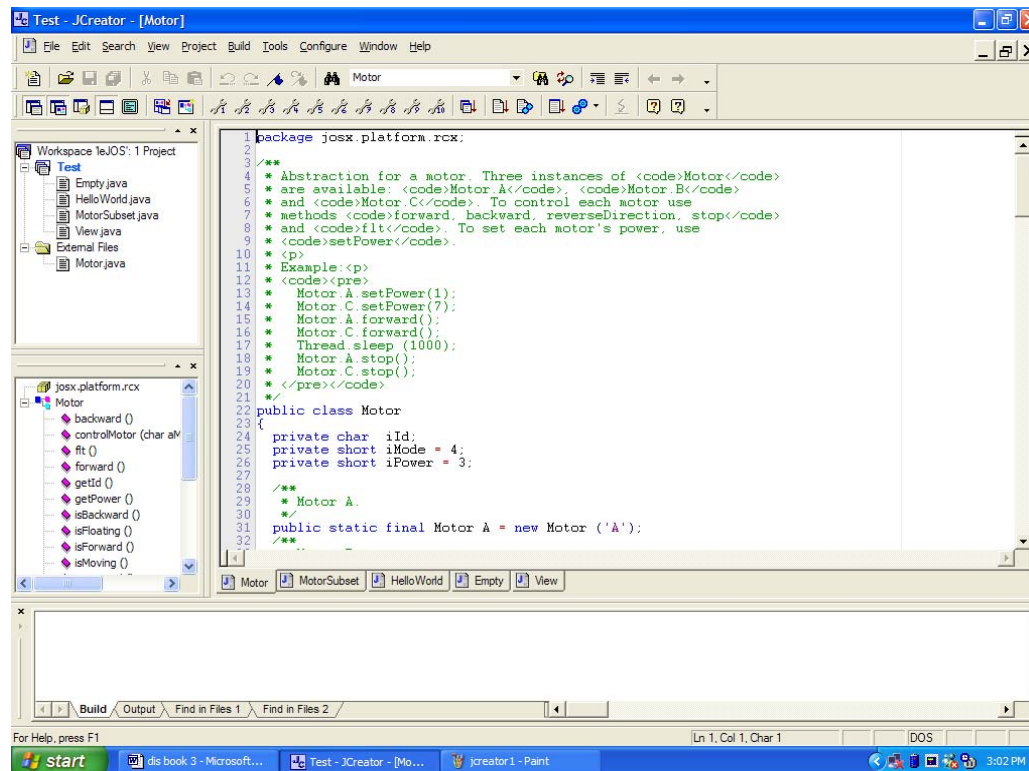
LeJOS documentation

Unzip the LeJOS documentation and put it in \\LeJOS\docs.

IDE – JCreator LE

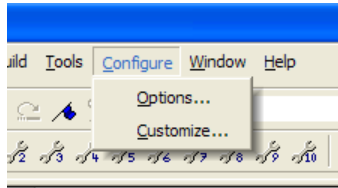
Download JCreator, the integrated development environment, IDE, from www.jcreator.com. The LE version is free for most uses and is adequate for our purposes. However, you may wish to purchase the more advanced version.

Extract the contents and run setup.

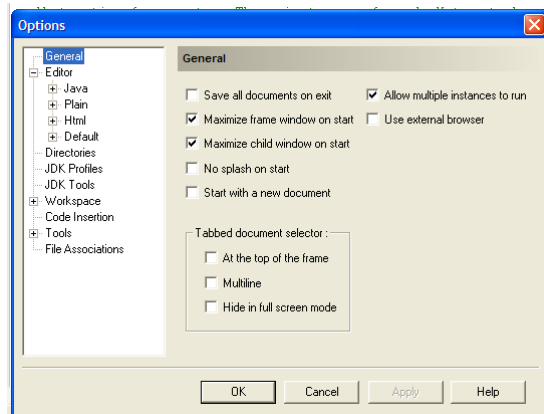


When you open JCreator for the first time, the accompanying screen appears. Notice that it has the standard menu items, File, Edit, Help, etc. and several tool bars that we use when programming. The main part of the screen has four panels: upper left is the workspace panel, middle left is the class panel, bottom is the output panel and the large panel on the right is the editing panel. In this picture, all of them are blank. The workspace panel shows the list of files that we are working with. The class panel is like a UML diagram, showing the methods of the classes we are working with. The editing panel is much like a word processor. This is where we actually type in our Java code. The output panel is where error messages from the compiler and messages from our programs are displayed. As we work through setting up JCreator, things will start appearing in these panels.

Setup legos, legosc and lejosfirmddl as tools in jcreator.

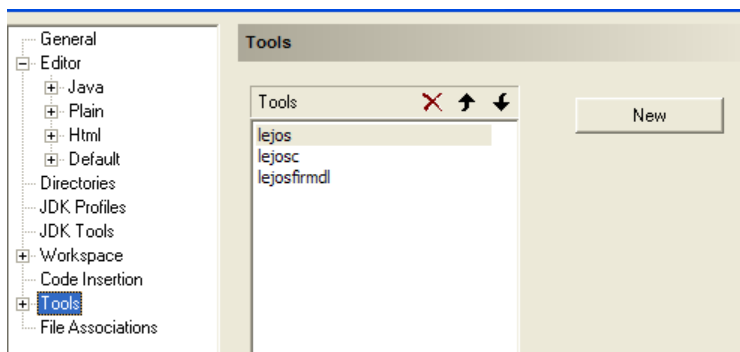


To begin, click on the configure menu item. A popup menu gives you the choice of options or customize. Click on options.

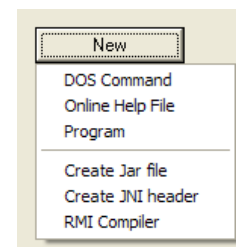


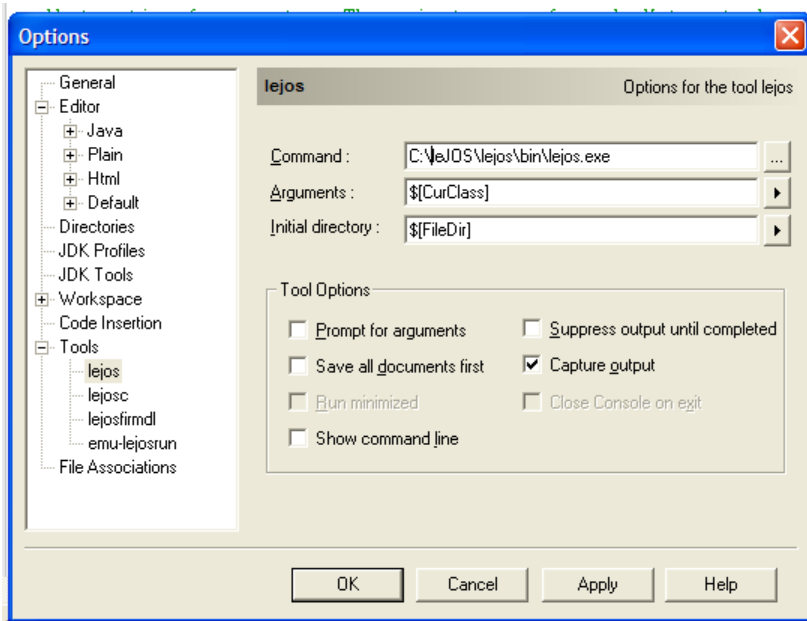
This brings up the options dialog box. The tree on the left shows all the option categories that you can set. You select a category by clicking on it. By default, the General category is selected. Categories with sub-categories are displayed preceded by a plus or minus sign. Clicking on the plus sign displays hidden sub-categories while clicking on a minus sign hides displayed sub-categories.

In order to use LeJOS inside of JCreator, we need to add the LeJOS programs to the Tools category. Click on Tools to proceed.



To add the tools, click on new. The popup menu on the right appears. You should click on Program. This brings up a file chooser dialog box. Look in your LeJOS/bin directory. You should see the programs listed at left. Pick lejosc as the first program and click open. Note that you will repeat this process four times for four different LeJOS tools, lejosc, lejos, lejosfirmddl and lejos-emu.

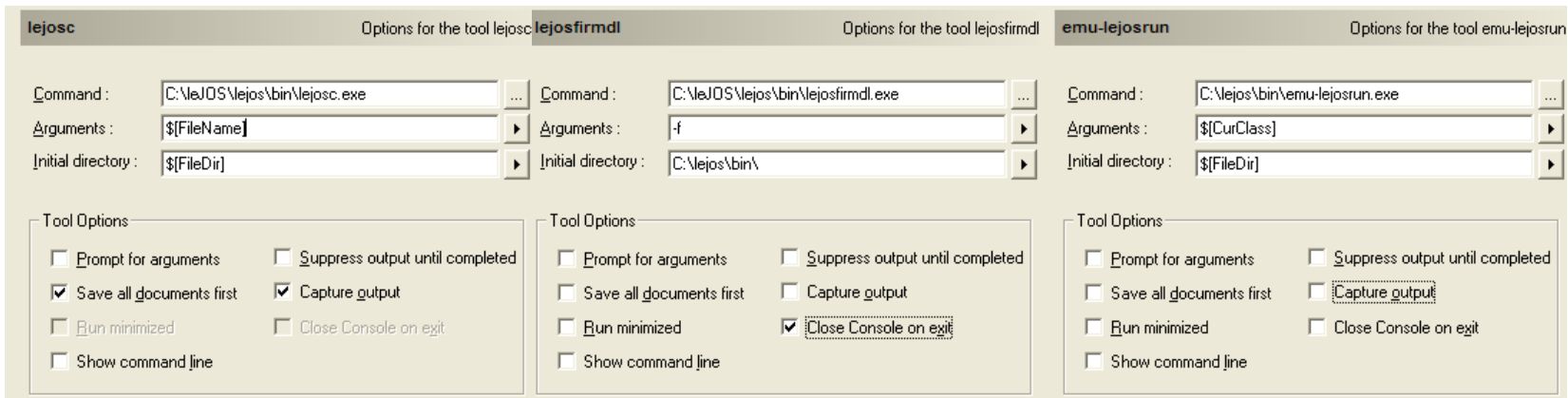




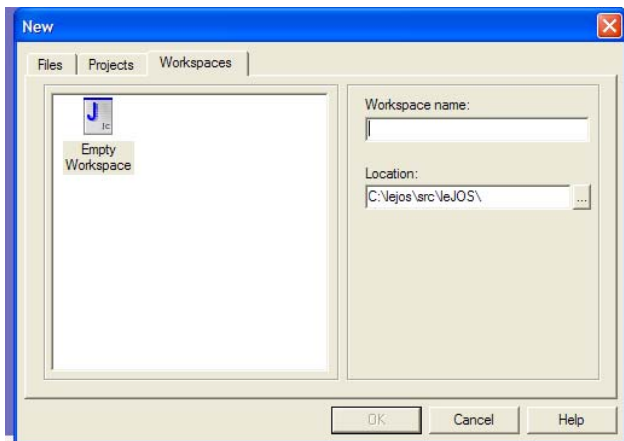
After you have created the four tools, click on lejos in the tree on the left. The options appear in the panel next to the tree. Using the buttons to the right of the fields, select arguments and the initial directory. For lejos, this current class and file directory.

Below are the option panels for the three other tools. Note that each has different arguments and initial directories. Note also that lejos and lejosc capture output while the other two do not and that lejosfirmddl

closes the console on exit while lejos-emu does not. You should experiment with these settings to see what happens when you change them. You can't do any permanent harm by experimenting. However, when things don't work, remember to return here to learn what the settings should be.

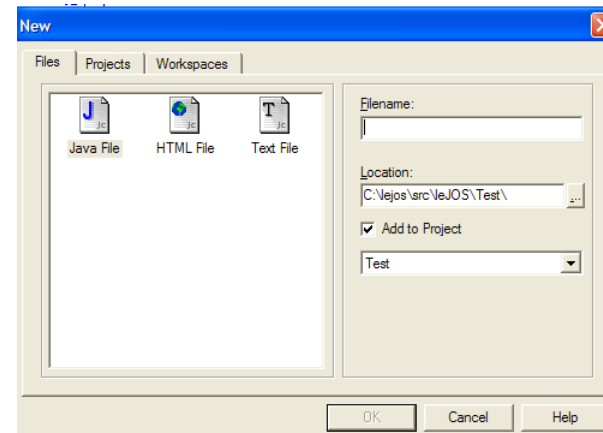


Create new workspace for JDK.



In JCreator, type `ctl-n` and the dialog box at left appears. Click on the Workspaces tab and enter a name for the new workspace. Click OK.

Type `ctl-n` again. Click on the files tab and enter the name HelloWorld in the file name box. Click OK.

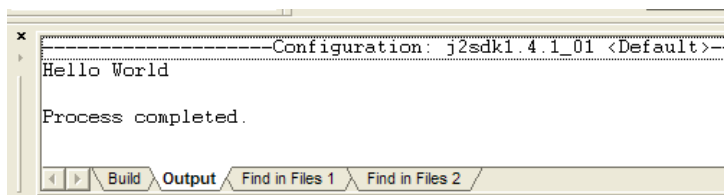


```

/* Hello World Example */
class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}

```

Copy the hello world program, at left, into the new file. Compile HelloWorld by selecting the compile file item under the build menu (`alt-b, f`). Then run HelloWorld by selecting the execute file item under the build menu (`alt-b, i`).



If you have successfully setup JCreator, then the build window that is at the bottom of JCreator will appear as at left.

```

import josx.platform.rcx.Motor;

class Testbot {

    private static void pause(int time) {
        try {
            Thread.sleep(time);
        } catch (InterruptedException e) {
        }
    }

    public static void spinRight() {
        Motor.A.backward();
        Motor.C.forward();
        pause(200);
        Motor.A.stop();
        Motor.C.stop();
    }

    public static void spinLeft() {
        Motor.C.backward();
        Motor.A.forward();
        pause(200);
        Motor.A.stop();
        Motor.C.stop();
    }

    public static void goForward() {
        Motor.A.forward();
        Motor.C.forward();
        pause(200);
        Motor.A.stop();
        Motor.C.stop();
    }

    public static void goBackward() {
        Motor.A.backward();
        Motor.C.backward();
        pause(200);
        Motor.A.stop();
        Motor.C.stop();
    }

    public static void main(String[] arg) {
        for (int idx = 0; idx < 5; idx++) {
            goForward();
            spinRight();
            goBackward();
            spinLeft();
        }
    }
}

```

Create new workspace for LeJOS.

Copy and paste the program at left into JCreator. Before you can paste, you must create a new file and name it Testbot. Compile the program using legosc, ctl-2, and download it to your robot using legos, ctl-1. If the robot does not start moving when you press the run button, you need to debug the program and the environment. Some common problems are:

- Firmware not loaded (run legosfirmddl, ctl-3).
- Robot not turned on.
- Interference on download (make sure IR tower is near RCX and protected from excess light).

Once the robot starts moving, you have succeeded in installing and testing your LeJOS and JCreator environment. The next step is experimenting with the program. Change the number in the pause statements or the count in the for statement. Change the behavior of spinRight from the RIS spin block command to the RIS turn block command.

At this point, you are ready to start learning how to really program in Java.