# NetBeans

This Wiki page provides an overview of the features of NetBeans for creating mobile Java™ applications. It then provides a simple introduction to creating a mobile Java application.

## Introduction

26 Apr 2009

The NetBeans IDE is an open-source integrated development environment based on the NetBeans Platform and has been written entirely in the Java language. The NetBeans IDE supports development for all Java application types (such as Java SE, Web, EJB, and Java ME mobile applications). Among other key features are an Apache Ant-based project system, version control, and refactoring tools.

For mobile development, NetBeans provides many Rapid Application Development (RAD) features, such as a Visual Flow Designer, Visual Form Designer, and tools for multitarget development.

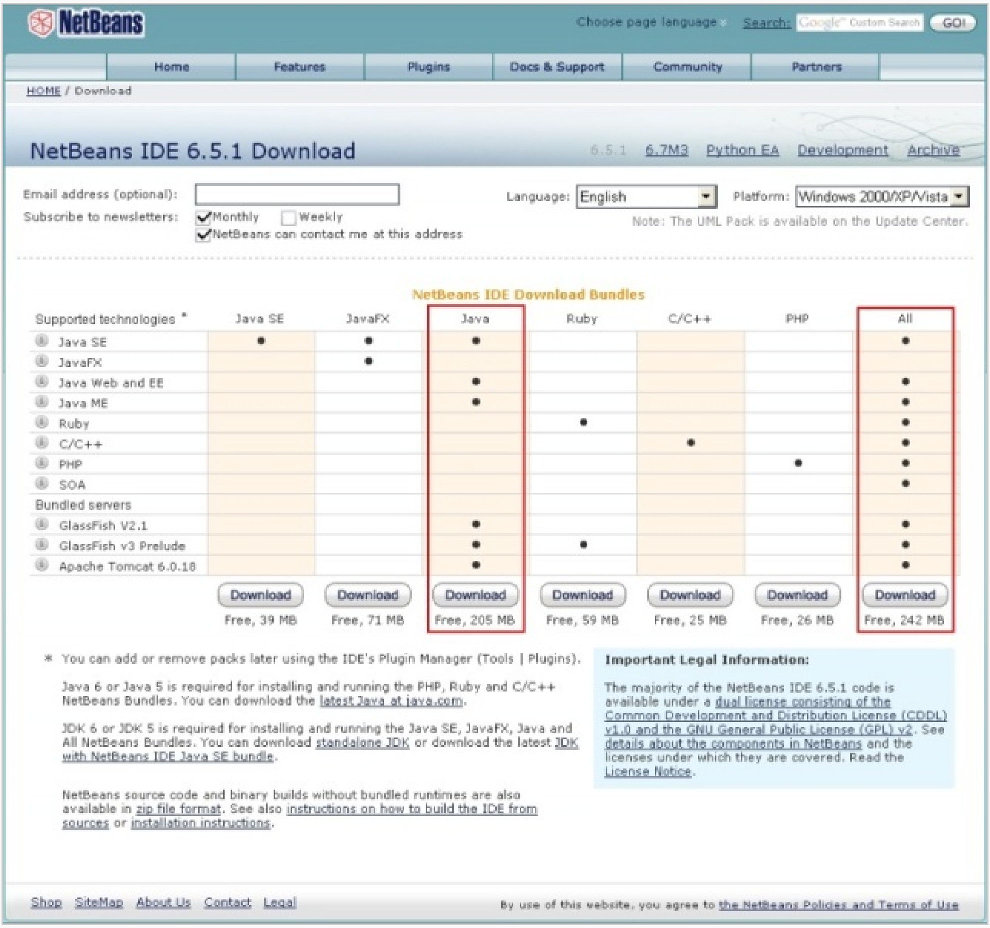
To enable the use of NetBeans by developers building mobile Java applications for devices, Nokia has provided support for NetBeans in many of the Series 40 and S60 SDKs, including all the recent Series 40 5th Edition and S60 3rd Edition SDKs.

Mobile Java developers may also be interested in the UML plug-in. This plug-in enables developers to design UML diagrams as Class Diagrams and to synchronize code with them. The UML plug-in is available in the **All** package of NetBeans or can be loaded into NetBeans using the IDE's plug-in manager (by selecting **Tools** -> **Plugins** from the menu).

Following are some of the key features of NetBeans for development of Java applications:

- Code completion.
- Javadoc™ help.
- Many refactoring features.
- Code assistant.
- Project view.
- The availability of many additional tools and plug-ins.

The current version of NetBeans is 6.5.1 and can be downloaded from http://netbeans.org/downloads/index.html. To develop mobile Java applications, developers should download the **Java** or **All** packages, which support Java ME development.



NetBeans IDE 6.5.1 Download

Email address (optional):
Subscribe to newsletters: ☑ Monthly □ Weekly
☑ NetBeans can contact me at this address

Language: English
Platform: Windows 2000/XP/Vista

Note: The UML Pack is available on the Update Center.

NetBeans IDE Download Bundles

| Supported technologies * | Java SE | JavaFX | Java | Ruby | C/C++ | PHP | All |
|---|---|---|---|---|---|---|---|
| Java SE | ● | ● | ● | | | | ● |
| JavaFX | | ● | | | | | |
| Java Web and EE | | | ● | | | | ● |
| Java ME | | | ● | | | | ● |
| Ruby | | | | ● | | | ● |
| C/C++ | | | | | ● | | ● |
| PHP | | | | | | ● | ● |
| SOA | | | | | | | ● |
| Bundled servers | | | | | | | |
| GlassFish V2.1 | | | ● | | | | ● |
| GlassFish v3 Prelude | | | ● | ● | | | ● |
| Apache Tomcat 6.0.18 | | | ● | | | | ● |
| | Download | Download | Download | Download | Download | Download | Download |
| | Free, 39 MB | Free, 71 MB | Free, 205 MB | Free, 59 MB | Free, 25 MB | Free, 26 MB | Free, 242 MB |

* You can add or remove packs later using the IDE's Plugin Manager (Tools | Plugins).

Java 6 or Java 5 is required for installing and running the PHP, Ruby and C/C++ NetBeans Bundles. You can download the latest Java at java.com.

JDK 6 or JDK 5 is required for installing and running the Java SE, JavaFX, Java and All NetBeans Bundles. You can download standalone JDK or download the latest JDK with NetBeans IDE Java SE bundle.

NetBeans source code and binary builds without bundled runtimes are also available in zip file format. See also instructions on how to build the IDE from sources or installation instructions.

**Important Legal Information:**

The majority of the NetBeans IDE 6.5.1 code is available under a dual license consisting of the Common Development and Distribution License (CDDL) v1.0 and the GNU General Public License (GPL) v2. See details about the components in NetBeans and the licenses under which they are covered. Read the License Notice.

By use of this website, you agree to the NetBeans Policies and Terms of Use

Download NetBeans

Following are some of the key features of NetBeans for development of mobile Java applications:

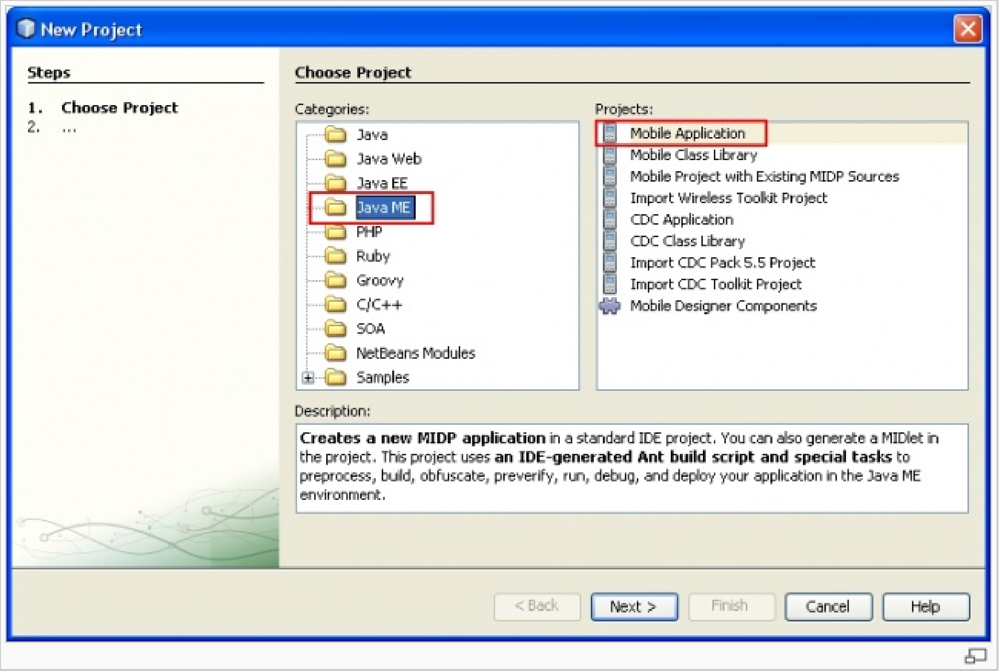
- Application Flow Designer.
- Screen Designer.
- Code analyzer.
- A modular build system.
- Application porting support.
- Wireless connection wizard (available in the **All** package).
- Scalable 2D Vector Graphics API for J2ME™ (JSR 226) support.
- Various deployment options, including deployment to a Nokia device using Nokia PC Suite.
- Project support.
- Integrated development tools.
- J2MEUnit testing support.
- Integrated obfuscation and optimization support.

For more information on these features, see the Nokia Developer NetBeans page.

# Getting started with NetBeans

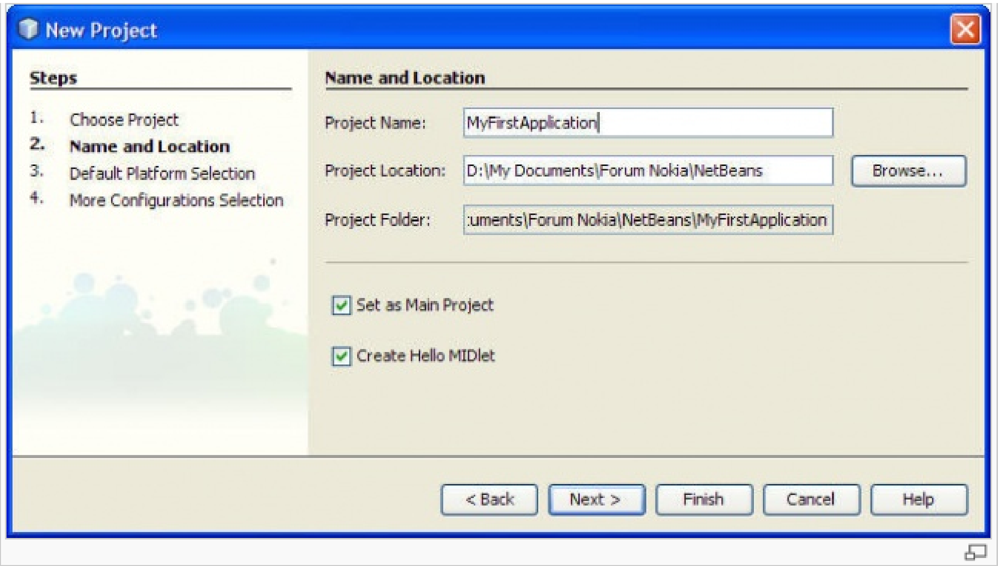
## Creating a new Java ME project with NetBeans

A NetBeans project contains information about program files and IDE environment settings. To create a new project, select **File -> New Project...** from the NetBeans menu or use the **Ctrl+Shift+N** keyboard shortcut. The window shown in the following figure will open.



**Creates a new MIDP application** in a standard IDE project. You can also generate a MIDlet in the project. This project uses **an IDE-generated Ant build script and special tasks** to preprocess, build, obfuscate, preverify, run, debug, and deploy your application in the Java ME environment.

Choose **Java ME** from the **Categories** list and then **Mobile Application** from the **Projects** list. Selecting the **Next** button opens the dialog shown in the following figure.



Steps

1. Choose Project
2. **Name and Location**
3. Default Platform Selection
4. More Configurations Selection

Name and Location

Project Name: MyFirstApplication

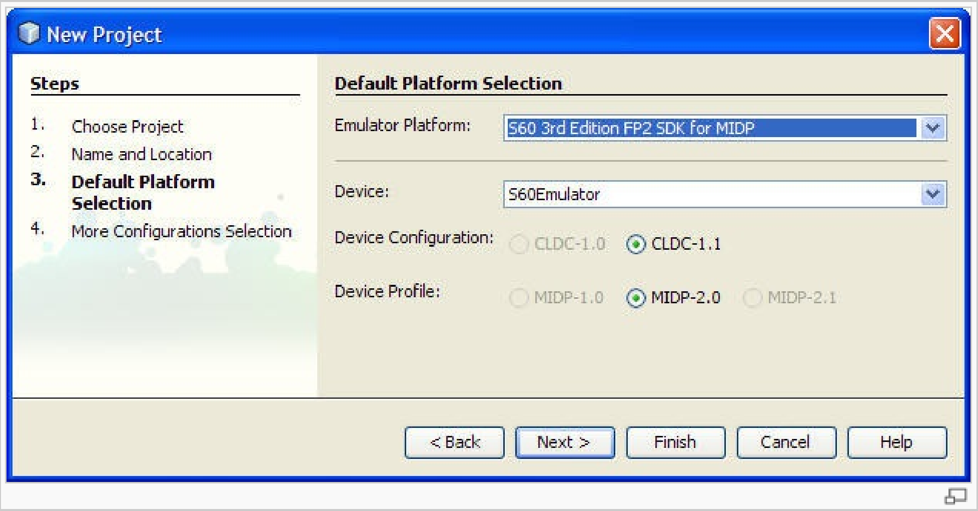
Project Location: D:\My Documents\Forum Nokia\NetBeans

Project Folder: :uments\Forum Nokia\NetBeans\MyFirstApplication

☑ Set as Main Project

☑ Create Hello MIDlet

Name the project and choose a location on the file system in which to store it. Note the option **Create Hello MIDlet**. Choosing this option will create a simple "Hello World"-style application, but more important, it will initialize the application to use the visual designers for application flow and screen layouts. Again, select the **Next** button, and the dialog shown in the following figure opens.



Steps

1. Choose Project
2. Name and Location
3. **Default Platform Selection**
4. More Configurations Selection

Default Platform Selection

Emulator Platform: S60 3rd Edition FP2 SDK for MIDP

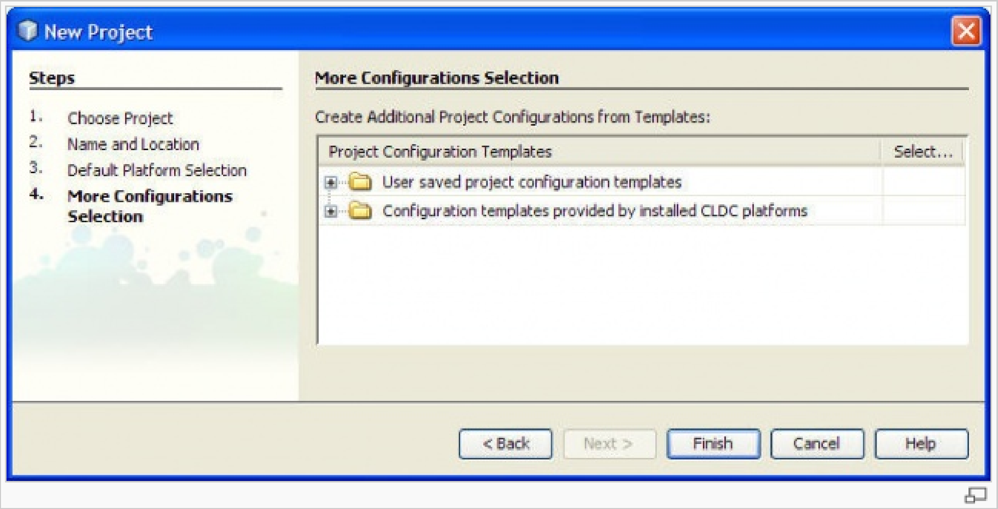
Device: S60Emulator

Device Configuration: ○ CLDC-1.0 ● CLDC-1.1

Device Profile: ○ MIDP-1.0 ● MIDP-2.0 ○ MIDP-2.1

If the **Create Hello MIDlet** option is not selected, you will have to create a MIDlet yourself after the project is created. For this, go to **File -> New File...**, select **MIDP** in the **Categories** list and **MIDlet** in the **File Types** list. Press **Next**, give your MIDlet a name and press **Finish**.

Now select the **Emulator Platform** the application will be coded against. You can use most of the S60 and Series 40 SDKs. In this case, the S60 3rd Edition, Feature Pack 2 SDK for Symbian OS, for Java has been selected. At this stage, the **S60Emulator** device is selected to enable testing of the application on a PC, using the emulator supplied in the SDK. Once again, select **Next**, and the following dialog is displayed.



Steps

1. Choose Project
2. Name and Location
3. Default Platform Selection
4. **More Configurations Selection**

More Configurations Selection

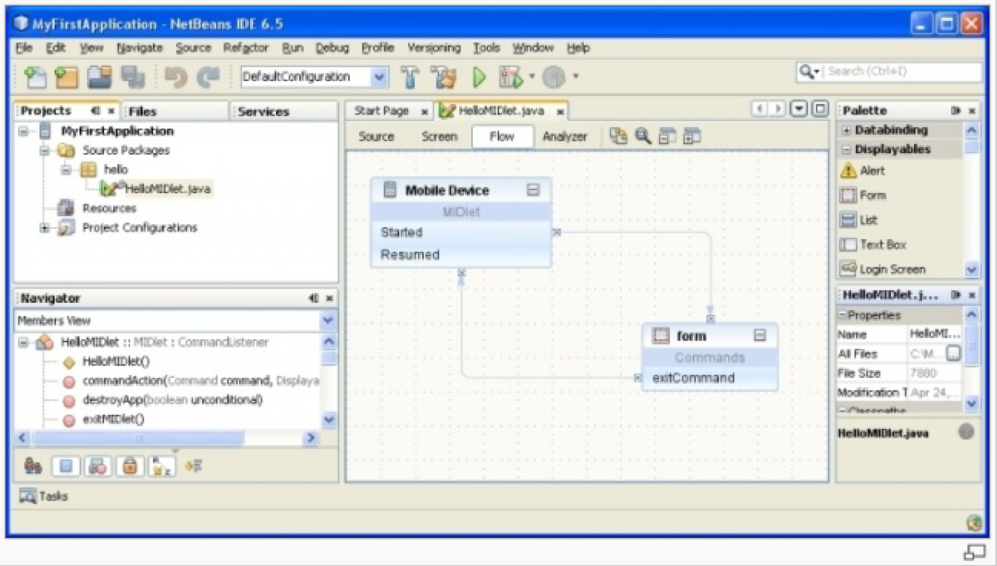
Create Additional Project Configurations from Templates:

| Project Configuration Templates | Select... |
|---|---|
| ⊞ User saved project configuration templates | |
| ⊞ Configuration templates provided by installed CLDC platforms | |

This dialog allows additional configurations to be added to a project. For a first application, no additional configurations are needed, so simply select **Finish** to complete the New Project wizard.

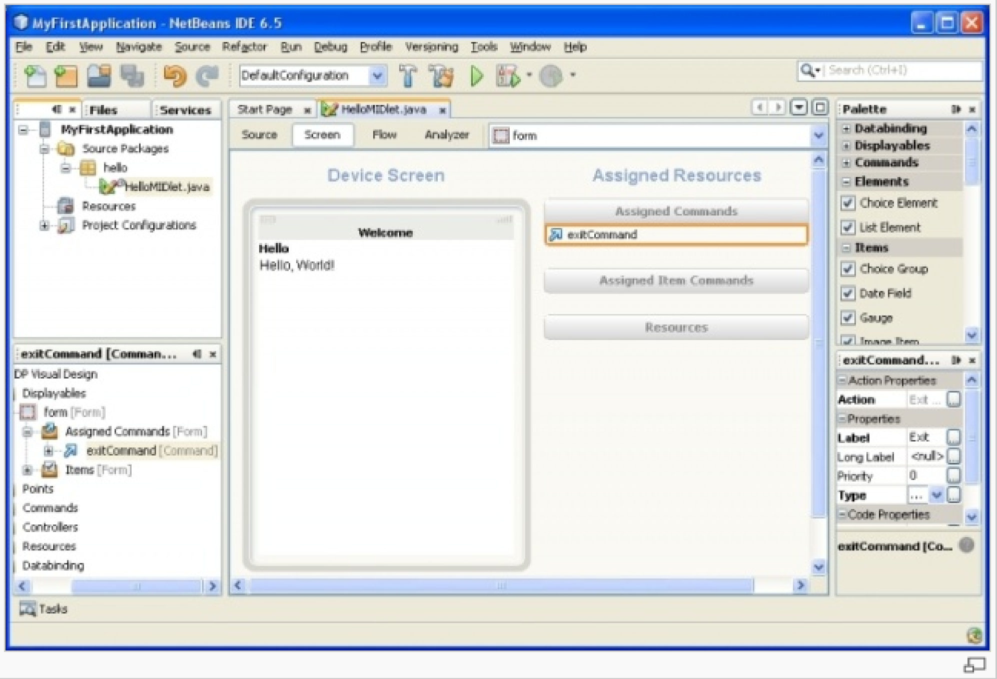
## Creating a flow, building screens, and adding code

Once the project has been created, NetBeans will display the Flow Designer in the NetBeans editor, as shown in the following figure.

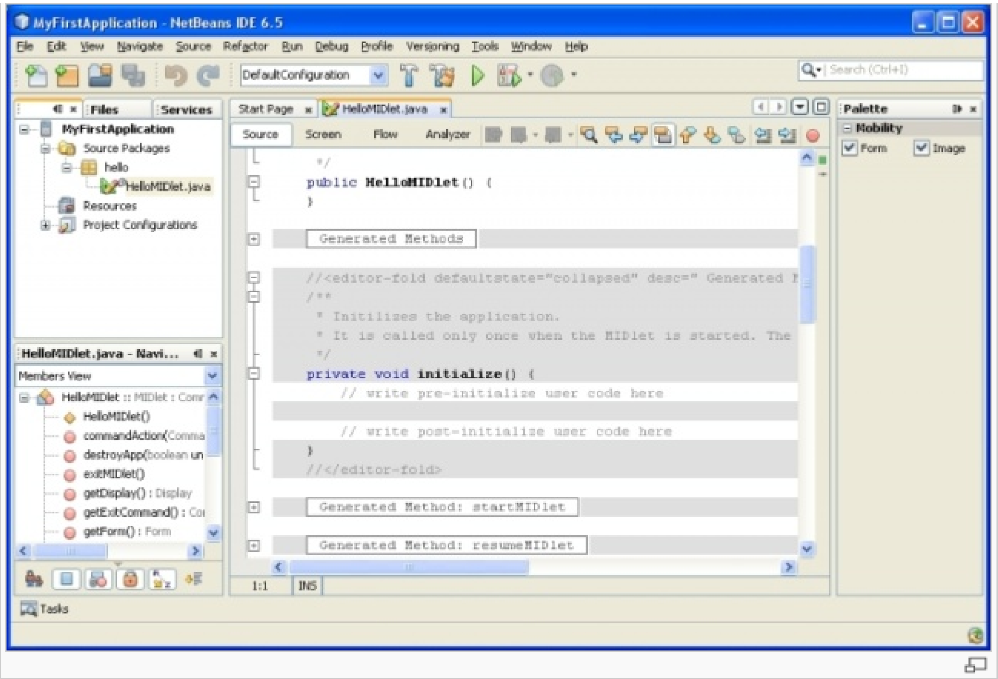


Create the application flow by dragging elements from **Displayables**, located under the **Palette** (in the right-hand window of the IDE) into the Flow Designer. Then grab commands and drag them to the displayable that they are required to flow to.

Once the flow has been created, individual screens can be created. This is done by selecting the displayable and then selecting the **Screen** tab in the Visual Designer. A screen similar to the one shown in the following figure will be displayed.



As with the Flow Designer, screen components can now be dragged into the **Device Screen** and edited to create the desired effect. Once the screen has been designed, the code can be modified to implement the application-specific business logic. Select the **Source** tab to open the code. A screen similar to the one in the following figure will be displayed.



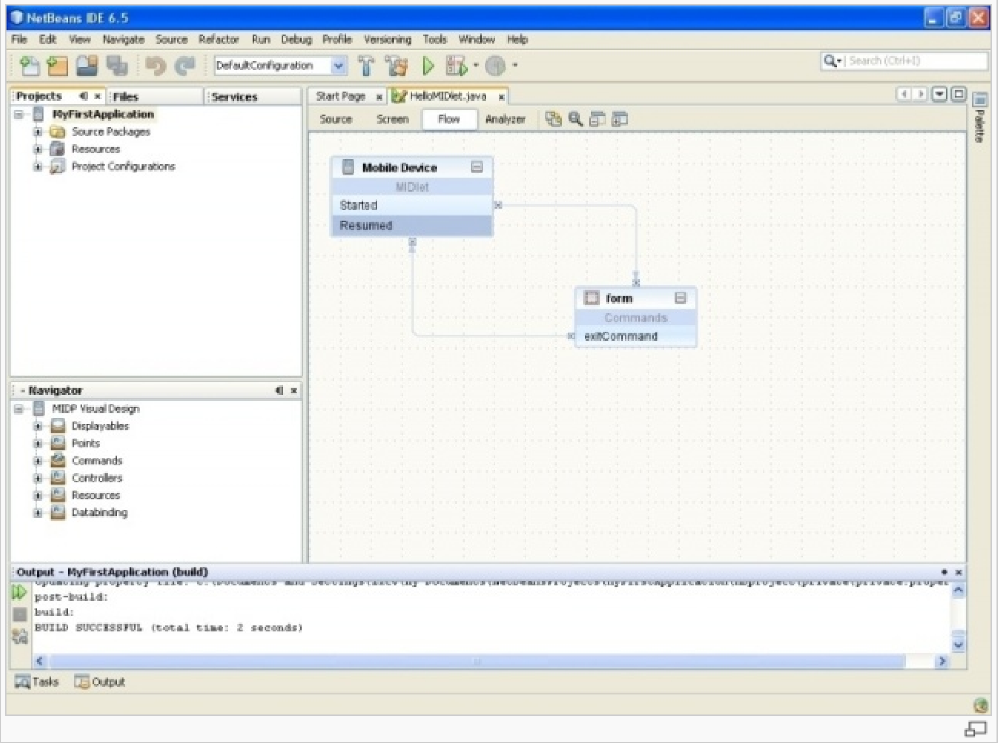
```
     */
    public HelloMIDlet() {
    }

    Generated Methods

    //<editor-fold defaultstate="collapsed" desc=" Generated M
    /**
     * Initilizes the application.
     * It is called only once when the MIDlet is started. The
     */
    private void initialize() {
        // write pre-initialize user code here

        // write post-initialize user code here
    }
    //</editor-fold>

    Generated Method: startMIDlet

    Generated Method: resumeMIDlet
```

When the source view is first opened, all the code generated by the flow and screen designers is in minimized code blocks labeled **Generated...**. Expand these code blocks to see the generated code highlighted in gray. This code cannot be changed. The application-specific code can be inserted in the sections of the code that are not highlighted.

With these tools, applications can be developed relatively quickly. In addition, it is worth noting that NetBeans provides features for taking advantage of JSR 226 and SVGT content. JSR 226 enables highly graphical content to be created with tools such as Adobe Illustrator and Ikivo Animator, eliminating the need to program graphics for menus and a range of application screens.

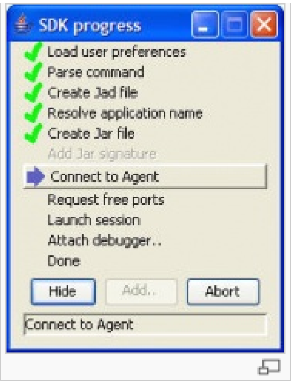
## Building the project and running it in the emulator

Having created the application with the Visual Designer and added any business logic, save the project by selecting **File -> Save** from the NetBeans menu (or by using the keyboard shortcut **Ctrl+S**). Now the project can be built. To do so, select **Run -> Build Main Project** from the NetBeans menu (or press the **F11** key). NetBeans will start to compile the project, as shown in the following figure. Information on the build process is displayed in the **Output** window at the bottom of the IDE.



```
post-build:
build:
BUILD SUCCESSFUL (total time: 2 seconds)
```

If there are no errors in the build process, the project can now be run in the S60 emulator. To do so, select **Run -> Run Main Project** from the NetBeans menu (or press **F6**). First, a dialog showing launch progress is displayed, as shown in the following figure.

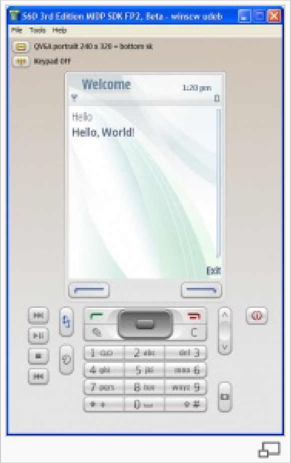


SDK progress

- ✓ Load user preferences
- ✓ Parse command
- ✓ Create Jad file
- ✓ Resolve application name
- ✓ Create Jar file
- Add Jar signature
- ➡ Connect to Agent
- Request free ports
- Launch session
- Attach debugger..
- Done

Connect to Agent

The emulator is then opened and the application started, as shown in the following figure.



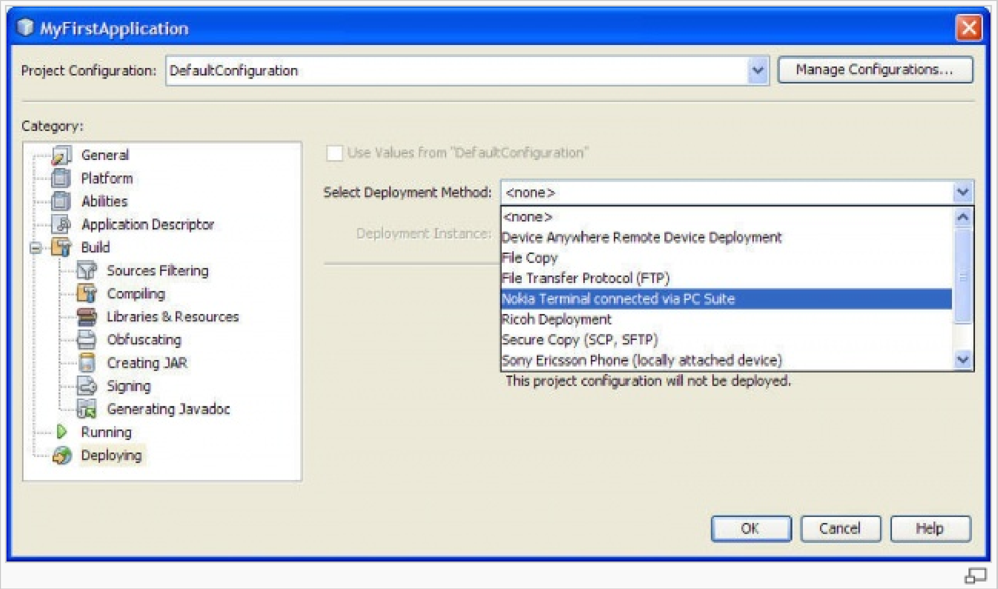
S60 3rd Edition MIDP SDK FP2, Beta - winscw udeb
File Tools Help
QVGA portrait 240 x 320 - bottom sk
Keypad Off
Welcome
1:20 pm
Hello
Hello, World!
Exit

The application can now be tested to confirm that it behaves in the manner expected. Once testing has finished, the application can be closed; however, the emulator should be left open if more testing is planned. Leaving the emulator open improves the speed of deploying the project to the emulator, because the emulator does not have to be started first.

## Deploying the MIDlet to a Nokia device

When the project is built, NetBeans creates its MIDlet Suite, consisting of a JAD file and a JAR file. These are found in the project's */dist* folder. If the Nokia PC Suite is installed on the development PC, the NetBeans deployment module can be used to deploy the application to a Nokia device connected to the PC by a Bluetooth, infrared, or USB connection.

To use this feature, the project needs to be told which deployment option to use. Do so by opening the project's properties, by selecting **File -> Project Properties** from the NetBeans menu. In the properties dialog, shown in the following figure, select **Deploying** from the **Category** list on the left of the display.



MyFirstApplication

Project Configuration: DefaultConfiguration

□ Use Values from "DefaultConfiguration"

Select Deployment Method: <none>

Deployment Instance:

<none>
Device Anywhere Remote Device Deployment
File Copy
File Transfer Protocol (FTP)
Nokia Terminal connected via PC Suite
Ricoh Deployment
Secure Copy (SCP, SFTP)
Sony Ericsson Phone (locally attached device)

This project configuration will not be deployed.

Now select **Nokia Terminal connected via PC Suite** as the method, and save the properties. Next, select **Deploy** from the project's context menu. The application will now be sent to the device connected to the PC, where it is installed in the usual way. Once installed, the application can be run in exactly the same way as it would be if it had been installed by a user.

Further information about deploying an application to a device is available here.

## More information

We hope that this short introduction has shown how powerful and convenient NetBeans is for creating mobile Java applications. There is much more information on creating mobile Java applications for Nokia devices in this Wiki and on the Nokia Developer website.

You can also find out more from the NetBeans web page.