CCS0023/L

Object Oriented Programming (Java)

EXPLORING NETBEANS IDE

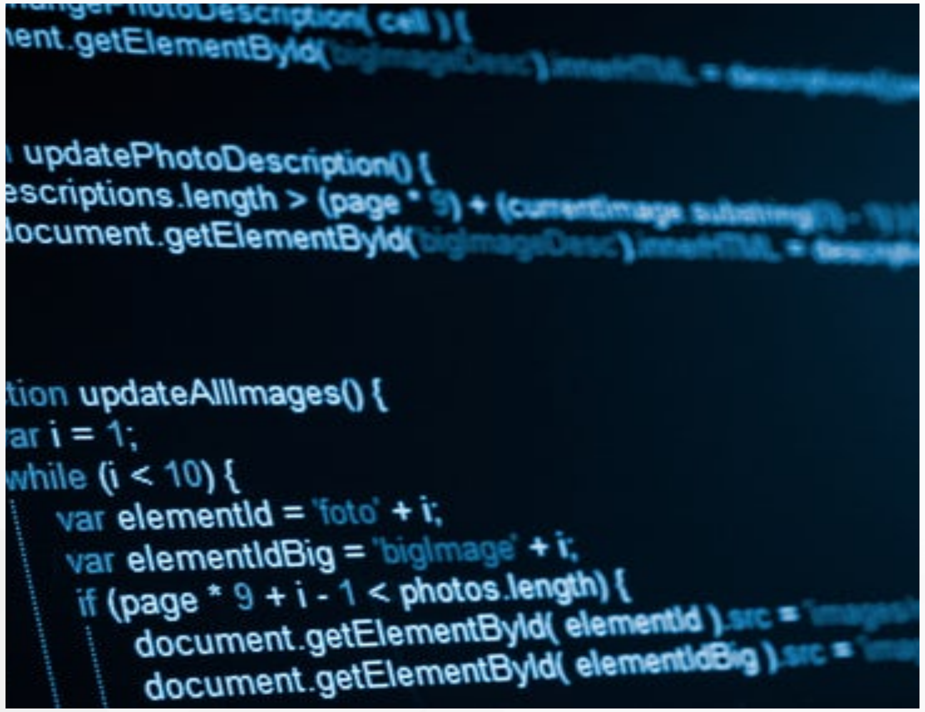
00

LESSON #0

Exploring NetBeans IDE

What you should know

- NetBeans IDE



```
ent.getElementById(

 updatePhotoDescription() {
escriptions.length > (page * 9) + (currentImage
ocument.getElementById(


tion updateAllImages() {
ar i = 1;
while (i < 10) {
   var elementId = 'foto' + i;
   var elementIdBig = 'bigImage' + i;
   if (page * 9 + i - 1 < photos.length) {
      document.getElementById( elementId ).src =
      document.getElementById( elementIdBig ).src =
```

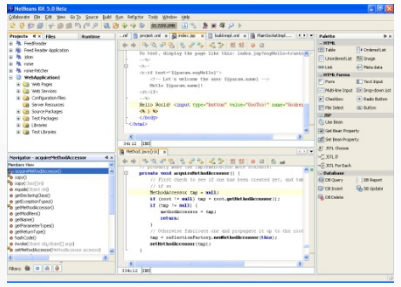
Objective

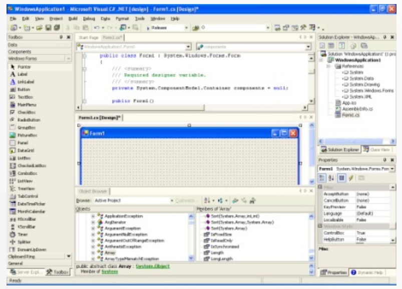
- Introduction to IDE
- NetBeans History & Overview
- NetBeans Matisse
- How To Get Started
- Detailed Demos

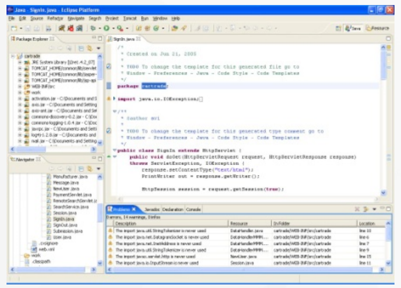
Introduction to IDE

- An **Integrated Development Environment** is a computer software to help computer programmers develop software.

- **The Leaders:**
  - NetBeans
  - Microsoft Visual Studio
  - Eclipse







Introduction to IDE

- **What does an IDE consist of:**
  - Source code Editor.
  - Compiler and/or interpreter.
  - Build- automation tools.

- **Optional Tools:**
  - Debugger.
  - Version control system.
  - **Various tools to simplify the construction of a GUI**.

Introduction to IDE

- **Tools For Object Oriented Design:**
  - Object inspector.
  - Class Browser.
  - Class hierarchy diagram.

Why Do We Need An IDE?

- IDE abstracts the configuration necessary to piece together various utilities in one unit, which could ease the learning of a language, and increases developer productivity.

- Most IDEs today have GUI modeling utilities that simplify the development of UIs, which is critical for commercial software today.

The History Of NetBeans

- It all started as a student project called Xelfi
- The Goal was to write a Delphi- like Java IDE in Java for the first time.
- The original plan was to develop network-enabled JavaBeans components, hens the name. but coming out of the spec for enterprise changed the plans.
- Sun decided it needs a more powerful Java development tool, and the rest is history.

About NetBeans

- A fast fully-featured Integrated Development Environment (IDE) with support for Java.
- Compliant applications for accelerating development across all major OS platforms.
- Provides an open source, high performance, modular, extensible, multi-platform Java IDE for GUI, mobile tools, Web, and Desktop applications.
- Written in java and therefore runs on every operating system that supports Java VM.

NetBeans Features

- **Environment:** easily configured user interface and a modular architecture extensible with additional plugins.
- **Project System:** support for multiple source roots, easy management of libraries, easily ported to other environments, all based on Apache Ant.
- **Web Development:** Web Application project type, Supports the J2EE 1.3 and 1.4 standards with web application build support based on Apache Ant.

NetBeans Features

- **Enterprise Java Beans (EJB) Development:** easy to create and deploy and import java beans.

- **Web Services Development:** wizards for creating web services and web services clients, providing the basic (java/wsdl) code needed, and easy to use testing tools of existing web services.

- **Java 2 Platform, Micro Edition (J2ME) MIDP development:** visual design editor with end-to-end support for enterprise applications.

NetBeans Features

- **Code Editor:** Syntax highlighting for Java, XML, HTML, CSS, JSP and IDL, full support of new JDK 1.5 features, live parsing/error marking, popup javadoc, code completion, and fast class importing.

- **Refactoring:** renaming, changing and moving of various objects, field encapsulation and usage finding.

- **Award Winning Debugger:** Language independent debugger core, variable modification and watches, various breakpoints and "Fix and Continue" mechanism.

NetBeans Features

- **GUI Builder:** fully WYSIWYG designer with "Test Form" feature, extensible Component Palette pre-installed Swing and AWT components, showing a components tree and properties, automatic code generation and full JavaBeans support.

- **Version control Support:** supports command lined vcs, supplying merging and diff tools and containing a built- in CVS client.

- **XML:** XML, DTD and CSS Text Editor and XML Productivity Tools Wizards to help user generate codes.

NetBeans Extras

- **NetBeans Profiler:** provides information about the runtime behavior of applications. Allows developers to monitor the thread state, CPU performance, and memory usage of their applications. makes it easy to track down performance problems and memory leaks.

- **NetBeans Platform:** provides the services common to almost all large desktop applications such as: window, menu, settings management and storage, file access and more.

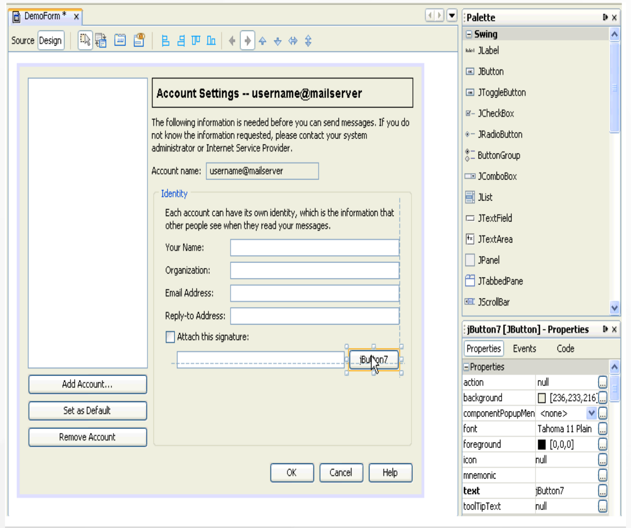
NetBeans Extras

- **NetBeans Mobility Pack:** used to write, test, and debug applications for the Java Micro Edition platform (J2ME) technology-enabled mobile devices. It integrates support for the Mobile Information Device Profile (MIDP) 2.0, the Connected, Limited Device Configuration (CLDC) 1.1.

- The mobility pack allows for the unique “On-Phone” debugging mode.

NetBeans Matisse

- The biggest improvement from the previous version and the feature with the biggest impact is the new GUI- Builder, Matisse.



Account Settings -- username@mailserver

The following information is needed before you can send messages. If you do not know the information requested, please contact your system administrator or Internet Service Provider.

Account name: username@mailserver

Each account can have its own identity, which is the information that other people see when they read your messages.

action	null
background	[236,233,216]
componentPopupMen	<none>
font	Tahoma 11 Plain
foreground	[0,0,0]
icon	null
mnemonic	
text	jButton7
toolTipText	null

NetBeans Matisse

- The Goal: to take the best features from OSX and VS designers and allow the same possibilities for Java Programmers.

- In order to reach that goal there was a need to develop a new layout manager to support all the needed functionalities.

NetBeans Matisse

- Matisse provides a simple and intuitive layout of GUIs without having to understand the complexities of Swing layout managers.
- As you drag and drop components into a form, the IDE automatically suggests alignment, spacing, and resizing constraints.
- By simply right clicking a UI Object you can add an event handler with a method waiting to be implemented without knowing too much about the surrounding of this object. (watch example clip in the site)

NetBeans Matisse - Advantages

- For the first time, there’s an intelligent way to build GUI for Java (unlike eclipse), that can actually compete with the Visual Studio Gui builder.
- Supports internationalization, and industrial look-and-feel rules, which is very important for large scale application meant to be spread world wide.
- Builds also GUI web applications, HTML, JSP, etc

NetBeans Matisse - Disadvantages

- No built-in support for Drag-n-Drop or double-click events.
- Matisse’s code is protected so customizing is not very easy and not always possible.
- Not all applications are easily built. For instance, an MDI Project is not that trivial to build

How To Get Started?

- How to install
- A Quick start Guide
- Importing existing applications
- Advanced

Installation

1. **Installing JDK:** in order to install NetBeans you need to first install JDK. You can easily find an installation in Sun’s web site.

1. **NetBeans installation:** you can find the installations kits for all the versions in the NetBeans home page.

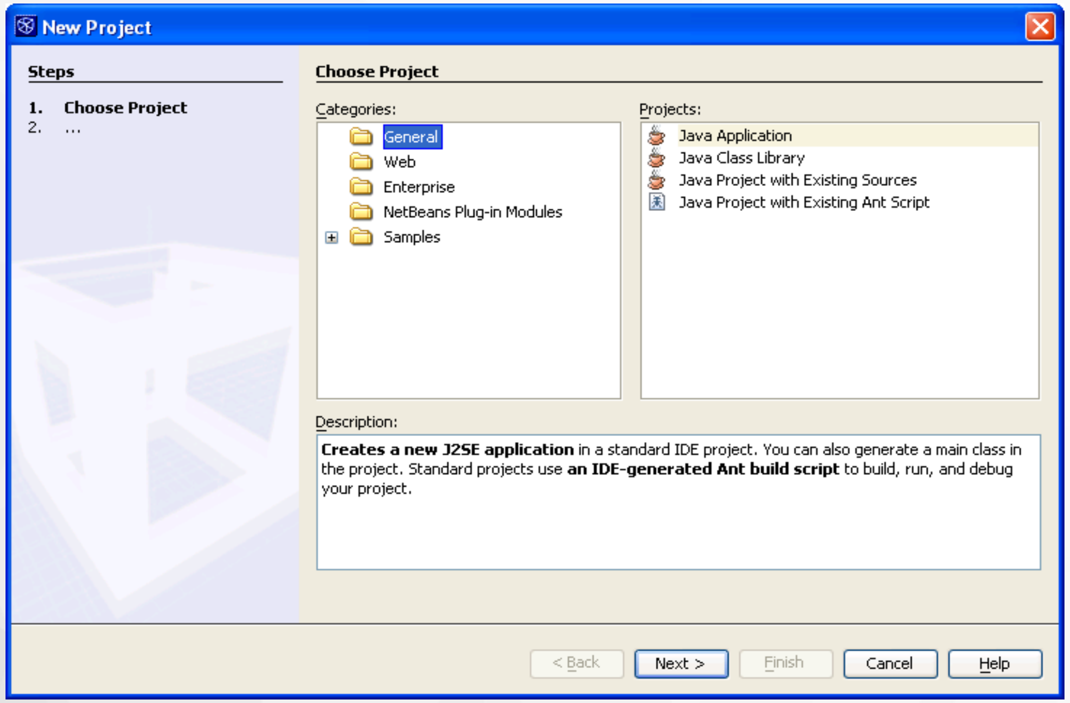
1. Installation steps: you can watch a movie describing the installation step by step in the.

Quick Start Guide

1. Starting up a project
2. Creating and editing java source code
3. Compiling and running a project
4. Testing and debugging a project

Starting Up A Project

- **Creating a new project:** in the file menu choose “new project” and look at the possibilities.



Creates a new J2SE application in a standard IDE project. You can also generate a main class in the project. Standard projects use **an IDE-generated Ant build script** to build, run, and debug your project.

Starting Up A Project

- When creating a new project, NetBeans already includes all the needed packages for compiling and testing. It also outlines the sources by the right logical directories and creates the files that are mandatory. For instance, if you create a new java application you must implement a main class and so it’s automatically created.

Editing and Refactoring

- When editing a code you can see the difference between the old version and the new version when using the Code Completion tool.

- You can use “Find Usage” tool when working on a project to predict the changes you will need to make.

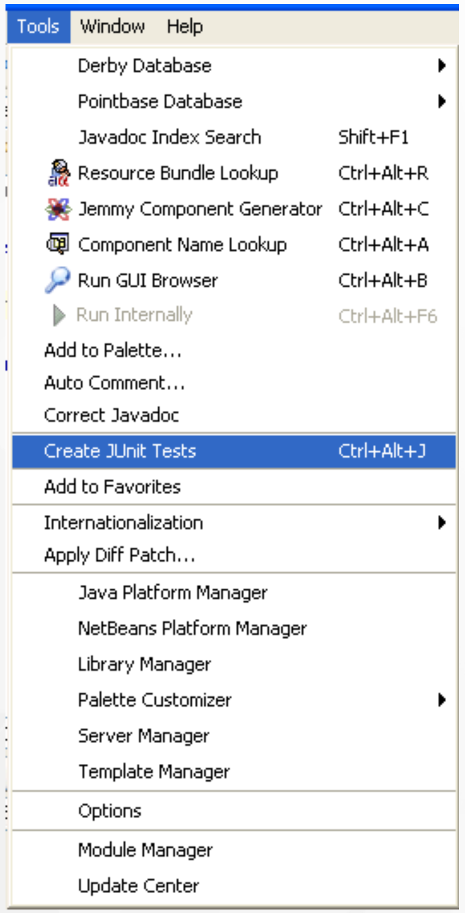
- You can use refactoring to easily change places of things without going through all the code to change it.

Building and Running

- No makefile or manual creation of an Ant file needed (by default). A simple instruction saying which class to run and what to build will suffice. Netbeans will create the Ant file automatically.
- While Building you can also generate the javadoc very easily.

Testing

- **JUnit Tests**: you simply choose the class you want to test and in the tools menu choose “create JUnit Test”. After filling the arguments NetBeans automatically creates a test class inheriting from TestCase with the default methods to implement and puts everything under the Test package.



Debugging

- NetBeans has two modes, run mode and debug mode.
- The debug mode is very easy to use.
- You can use the local variables window and watch window to follow the progress of the program.

NetBeans IDE Java Quick Start Tutorial

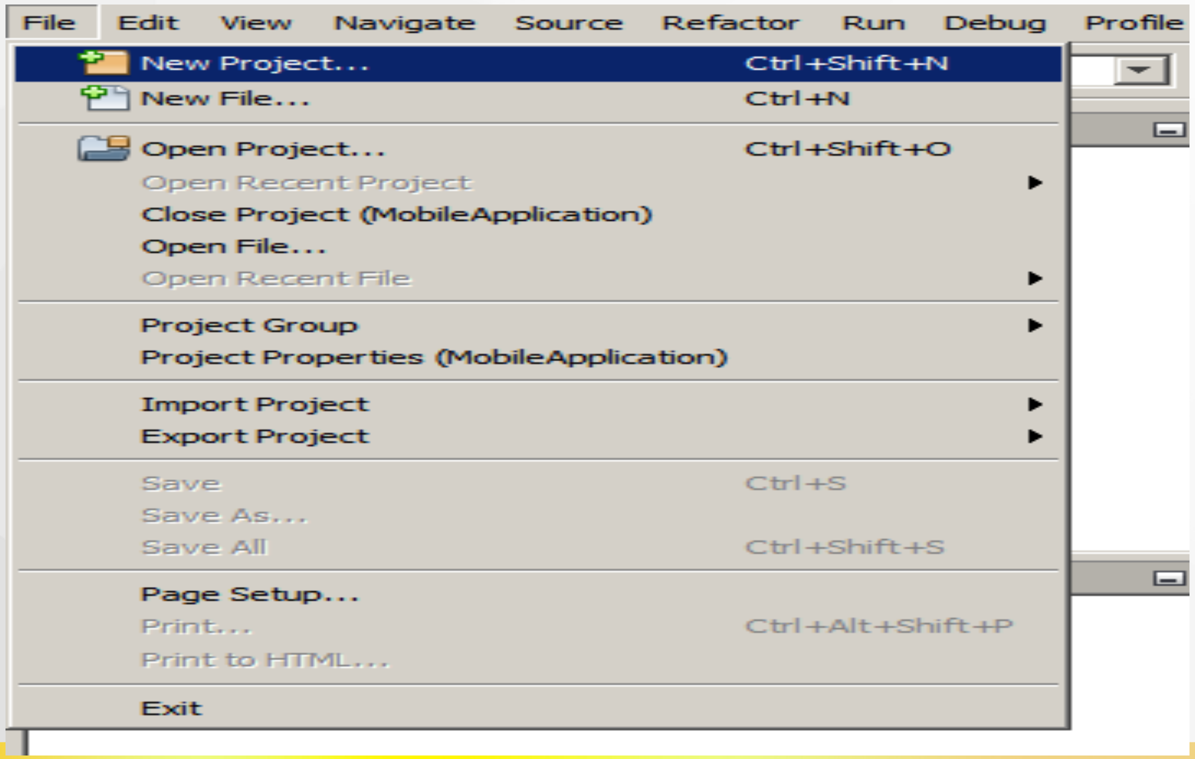
Contents:
✓Setting Up the Project
✓Adding Code to the Generated Source File
✓Compiling and Running the Application

Required Software/Resource:
✓NetBeans IDE version 7.2, 7.3, 7.4, or 8.0
✓Java Development Kit (JDK)version 6, 7, or 8

Setting Up Your First Project

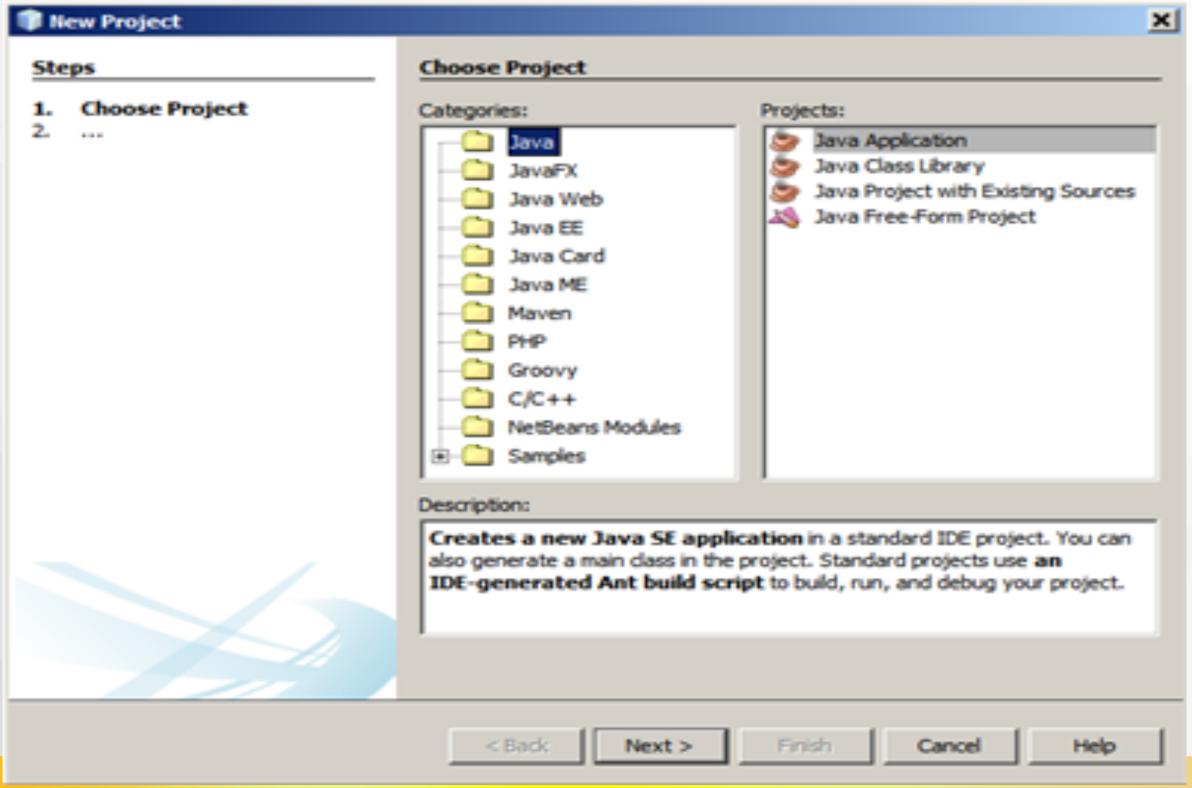
To create an IDE project:

1. Start NetBeans IDE.
2. In the IDE, choose File > New Project, as shown in the figure below.



Setting Up Your First Project

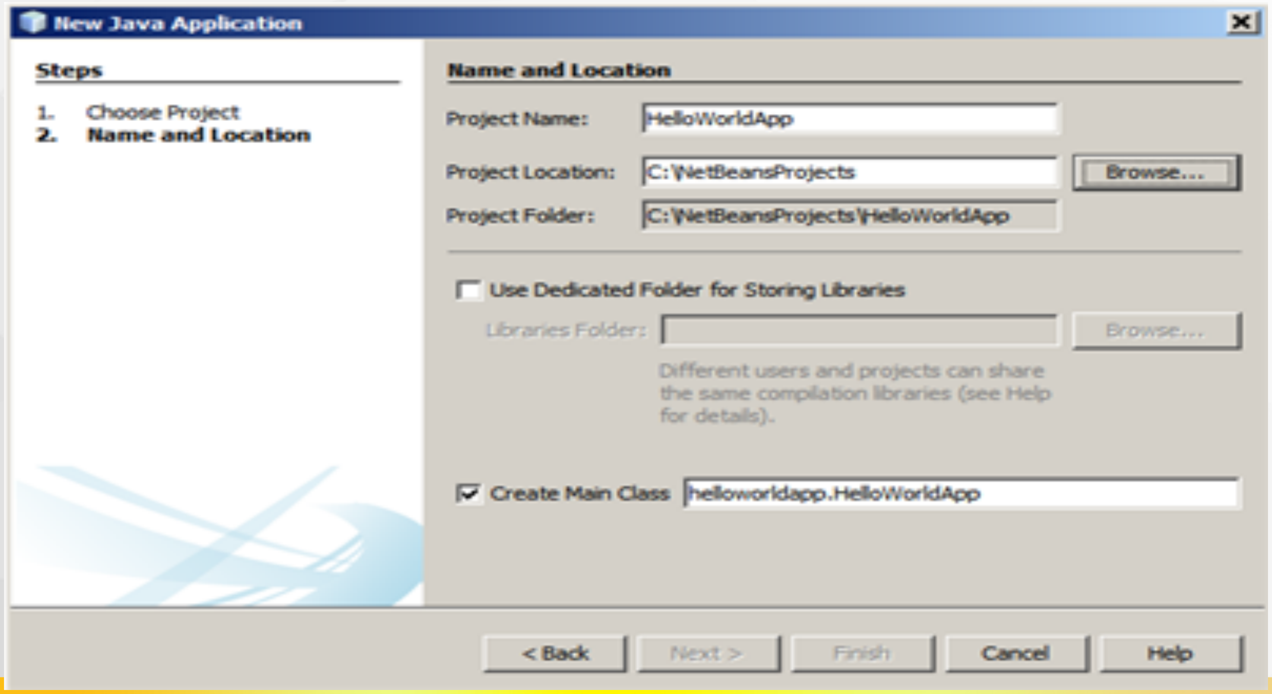
3. In the New Project wizard, expand the Java category and select Java Application as shown in the figure below. Then click Next.



Creates a new Java SE application in a standard IDE project. You can also generate a main class in the project. Standard projects use **an IDE-generated Ant build script** to build, run, and debug your project.

Setting Up Your First Project

4. In the Name and Location page of the wizard, do the following:
   ✓In theProject Name field, type HelloWorldApp.
   ✓Leave the Use Dedicated Folder for Storing Libraries checkbox unselected.
   ✓In the Create Main Class field, type helloworldapp.HelloWorldApp.



Project Name: HelloWorldApp
Project Location: C:\NetBeansProjects
Project Folder: C:\NetBeansProjects\HelloWorldApp

☐ Use Dedicated Folder for Storing Libraries
Libraries Folder:
Different users and projects can share the same compilation libraries (see Help for details).

☑ Create Main Class helloworldapp.HelloWorldApp

Setting Up Your First Project

5. Click Finish.

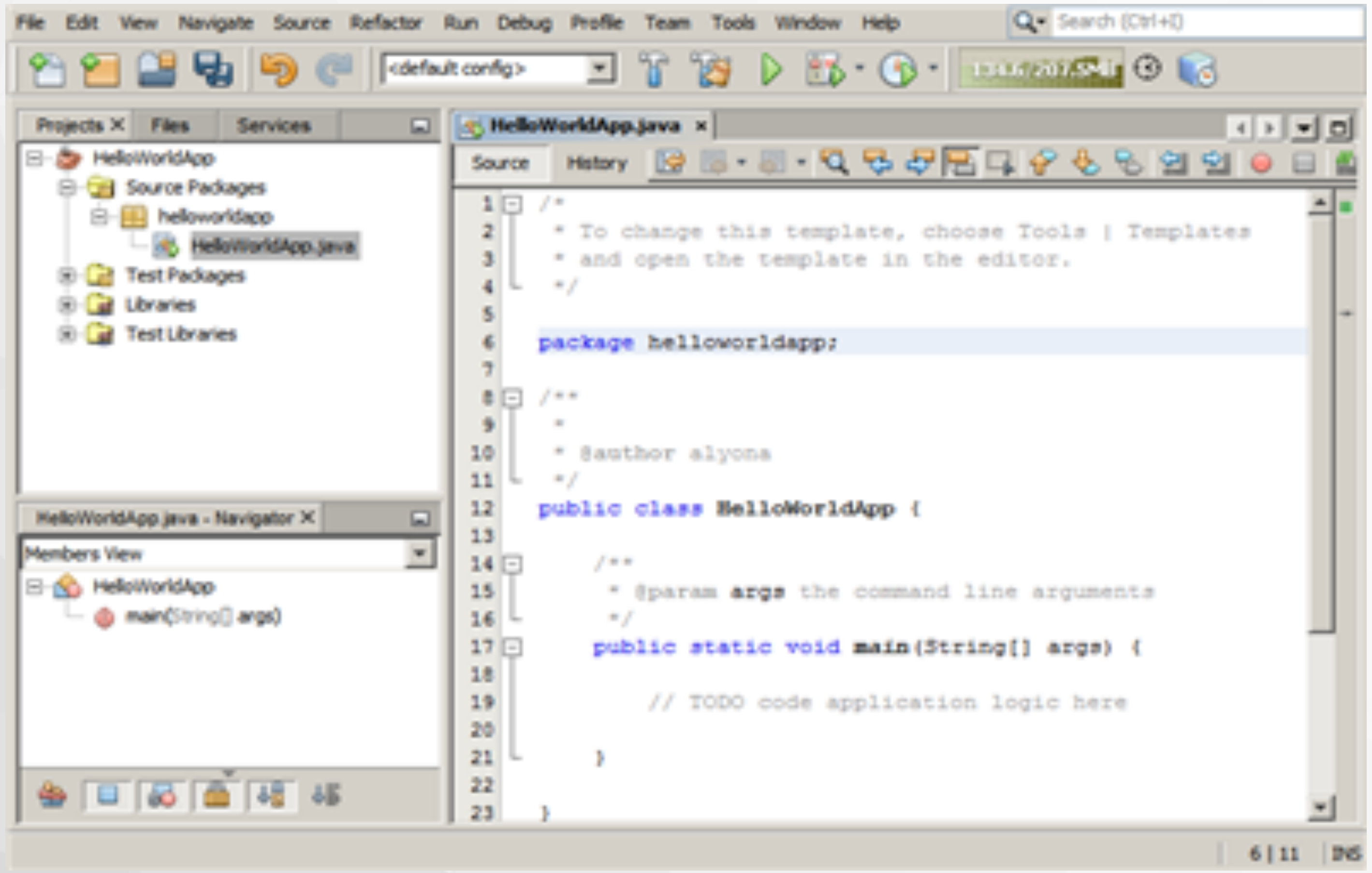
✓The project is created and opened in the IDE. You should see the following components:

✓The Projects window, which contains a tree view of the components of the project, including source files, libraries that your code depends on, and so on.

✓The Source Editor window with a file called HelloWorldApp open.

✓The Navigator window, which you can use to quickly navigate between elements within the selected class.

Setting Up Your First Project



```java
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package helloworldapp;

/**
 *
 * @author alyona
 */
public class HelloWorldApp {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {

        // TODO code application logic here

    }

}
```

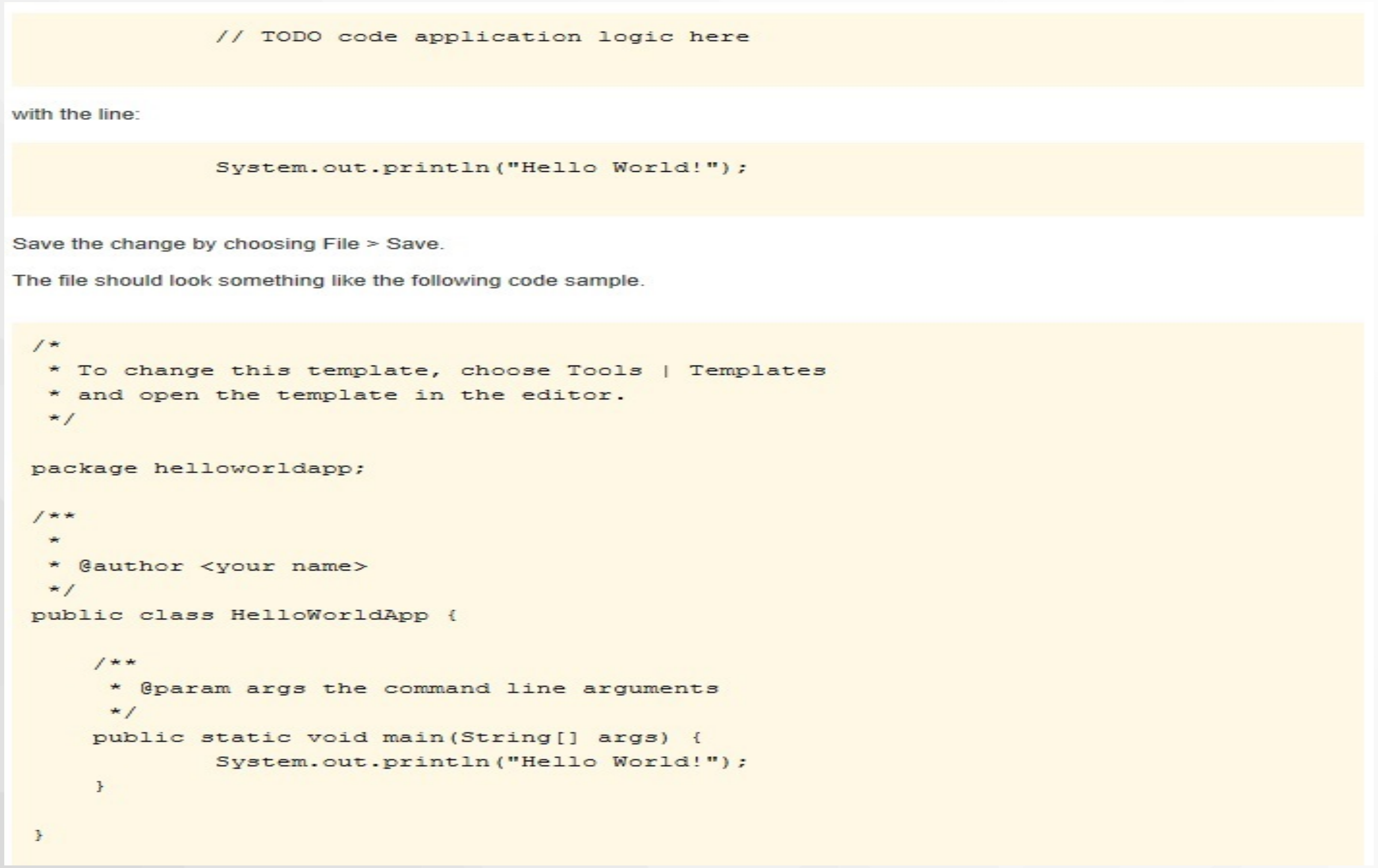
- Exploring NetBeans IDE

Adding Code to the Generated Source File

Because you have left the Create Main Class checkbox selected in the New Project wizard, the IDE has created a skeleton main class for you.

You can add the "Hello World!" message to the skeleton code by replacing the line:

Setting Up the Project



```
            // TODO code application logic here
```

with the line:

```
            System.out.println("Hello World!");
```

Save the change by choosing File > Save.

The file should look something like the following code sample.

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package helloworldapp;

/**
 *
 * @author <your name>
 */
public class HelloWorldApp {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
            System.out.println("Hello World!");
    }

}
```

Compiling and Running the Program

✓Because of the IDE's Compile on Save feature, you do not have to manually compile your project in order to run it in the IDE.
✓When you save a Java source file, the IDE automatically compiles it.
✓The Compile on Save feature can be turned off in the Project Properties window.
✓Right-click your project, select Properties. In the Properties window, choose the Compiling tab.
✓The Compile on Save checkbox is right at the top.

Compiling and Running the Program

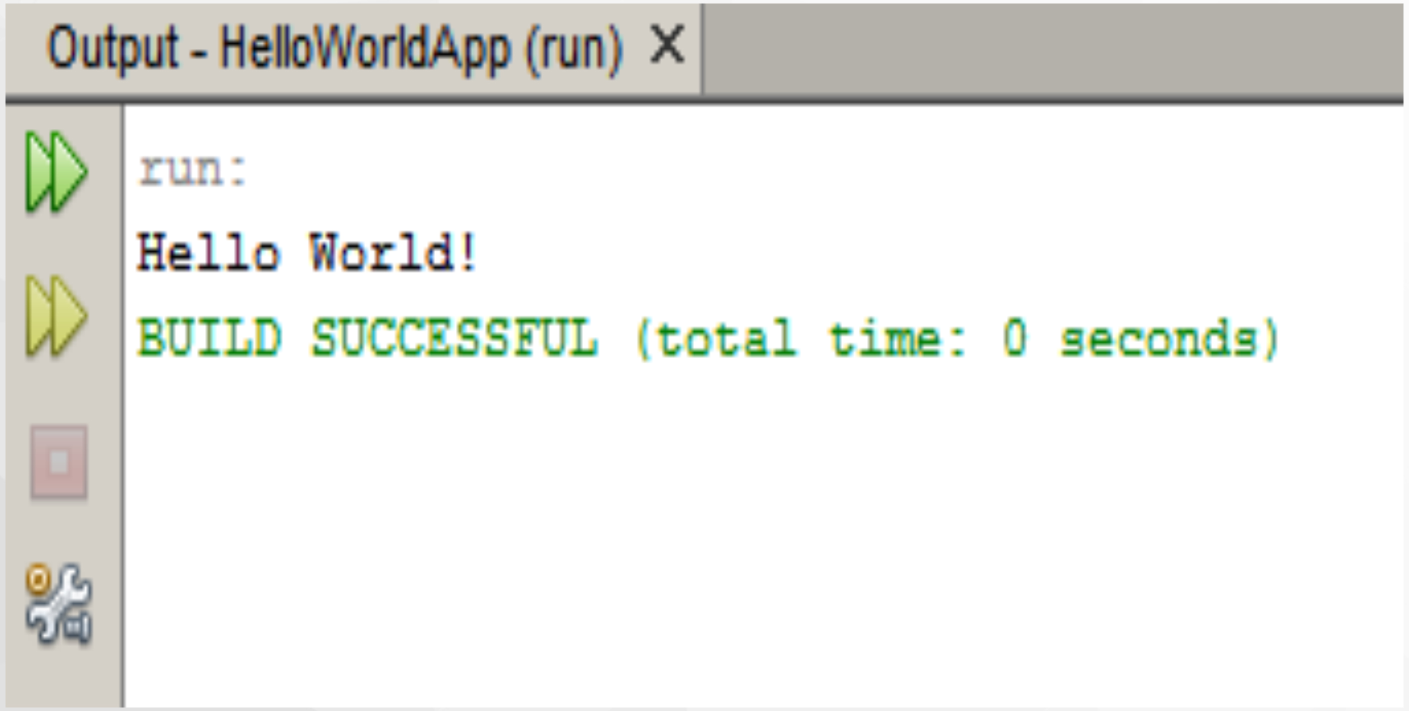
Note that in the Project Properties window you can configure numerous settings for your project: project libraries, packaging, building, running, etc.

To run the program:

Choose Run > Run Project.
Press F6/Shift+F6.
Click the run button ▷ from toolbar.
Click the play button ⏩ from output window.

The next figure shows what you should now see.

Compiling and Running the Program



```
run:
Hello World!
BUILD SUCCESSFUL (total time: 0 seconds)
```

THANK YOU