Programming Techniques Part 1-JAVA BCNS1102C

Lecture 2-3: Pseudocodes, IDEs, Java Program Structure

Lecturer: Mr. Ajit Kumar Gopee

E-mail:ajit.gopee@utm.ac.mu

Outline

- Pseudocodes (Continued)
- IDEs for JAVA
  - BlueJ
  - DrJava
  - Eclipse
  - Jbuilder
  - NetBeans (We will use this one in class)
- Java Program Structure
  - Data types
  - Statements
  - Simple Java Program

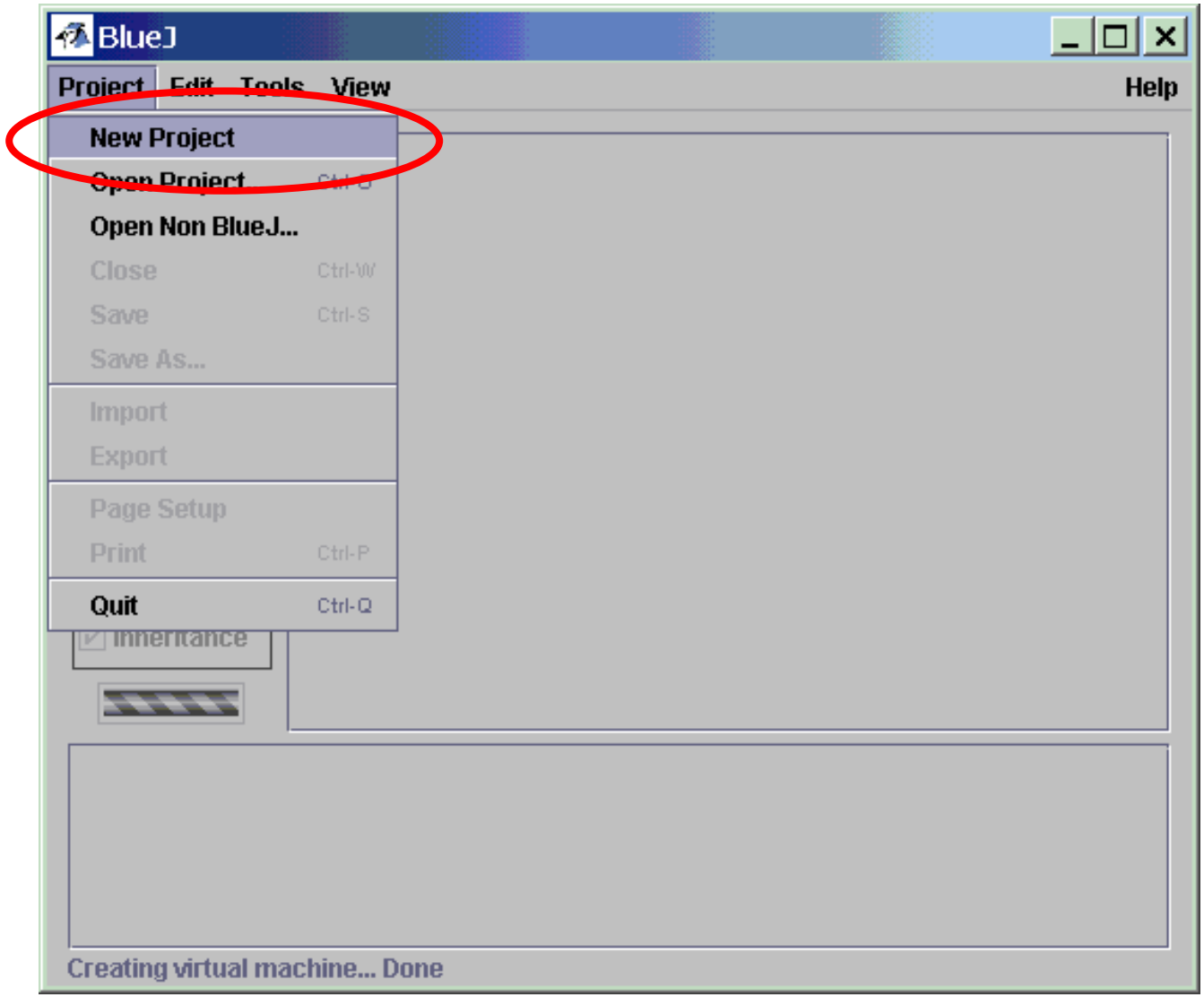
About IDEs

- An IDE is an Integrated Development Environment
- Different IDEs meet different needs
  - BlueJ, DrJava are designed as teaching tools
    - Emphasis is on ease of use for beginners
    - Little to learn, so students can concentrate on learning Java
  - Eclipse, JBuilder, NetBeans are designed as professional-level work tools
    - Emphasis is on supporting professional programmers
    - More to learn, but well worth it in the long run
- We will use NetBeans, but other professional IDEs are similar
- The following slides are taken from
  www.eclipse.org/eclipse/presentation/eclipse-slides.ppt

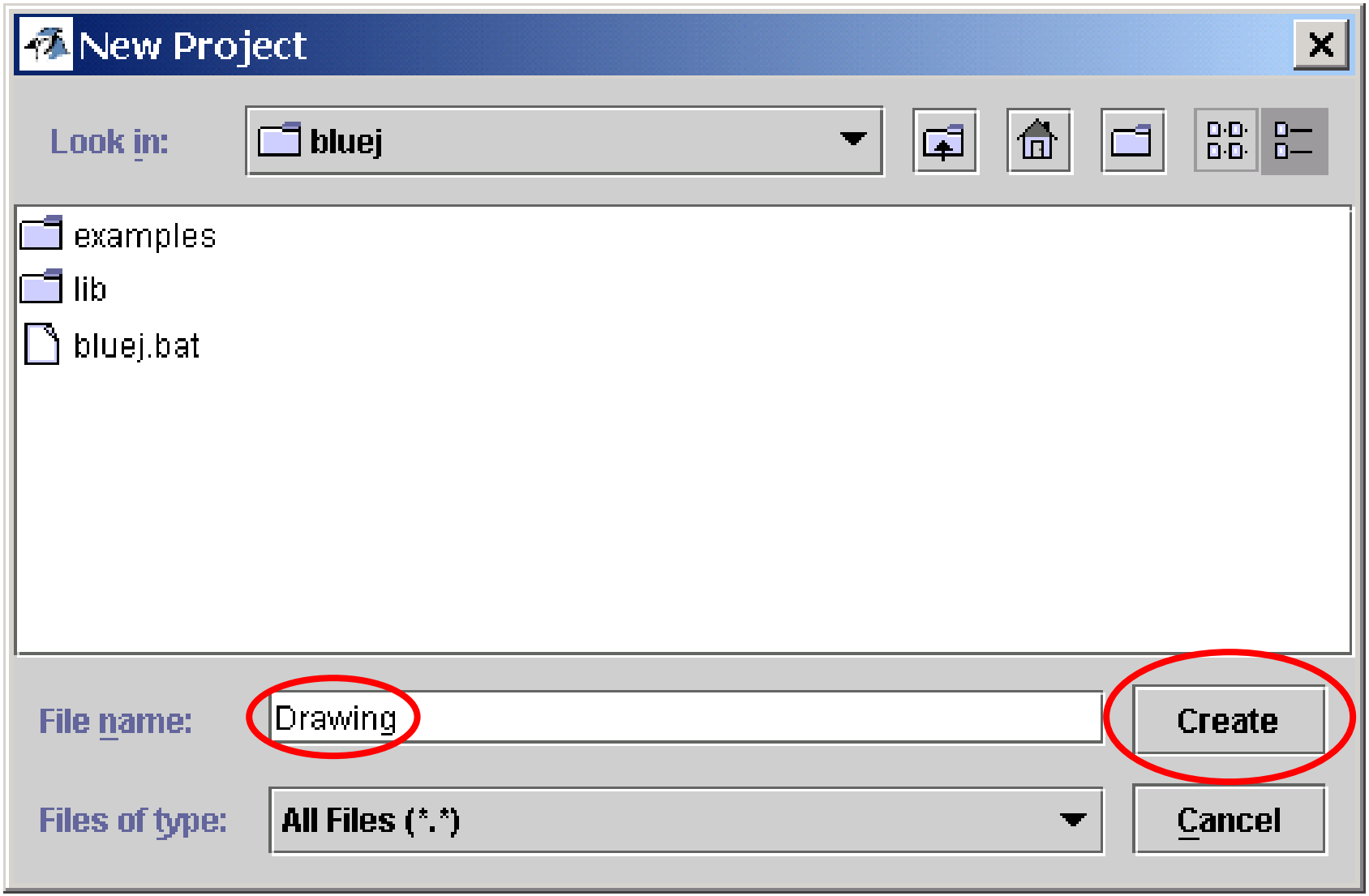
BlueJ

- BlueJ is an IDE (Integrated Development Environment). It includes
  - an editor, which you use to write your programs
  - a debugger, to help you find your mistakes
  - a viewer, to see the parts of your program
  - an easy way to run Java programs
  - an easy way to view documentation

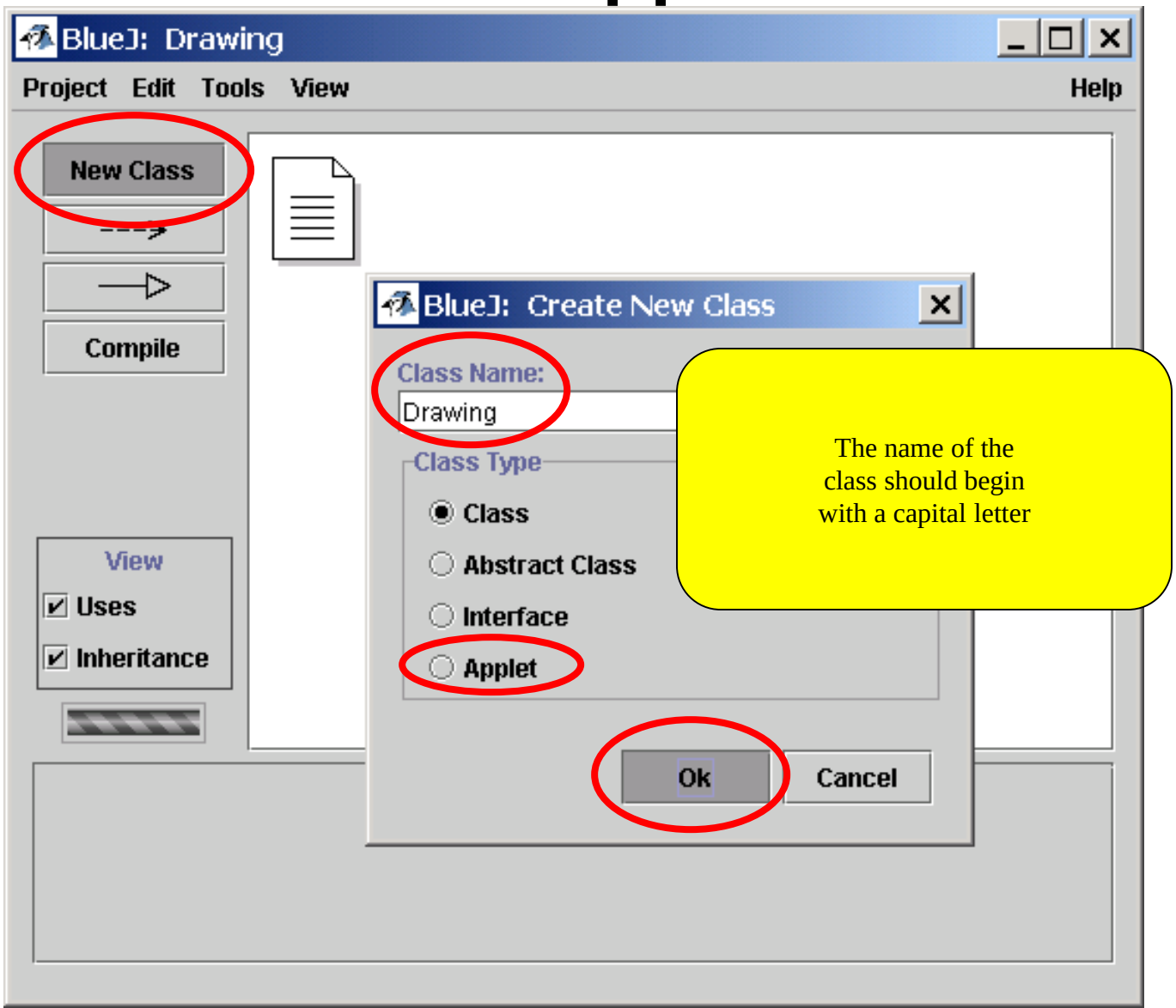
Open BlueJ, choose New Project



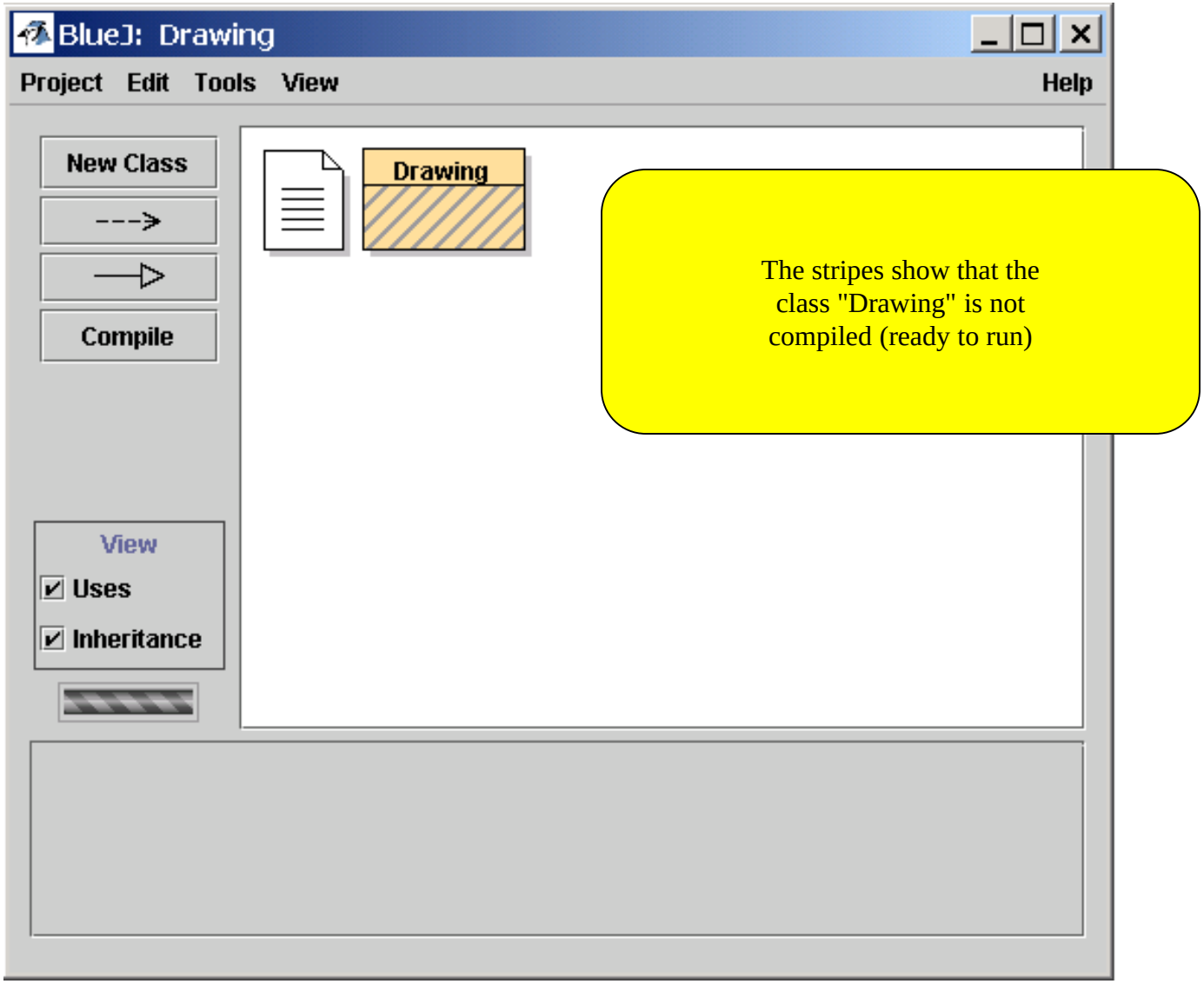
Name the project, click Create



Create and name a new Class or Applet



Double-click the new class to edit it



Edit your class, then Compile it

You must compile before you can run.

Drawing

Class Edit Tools Options

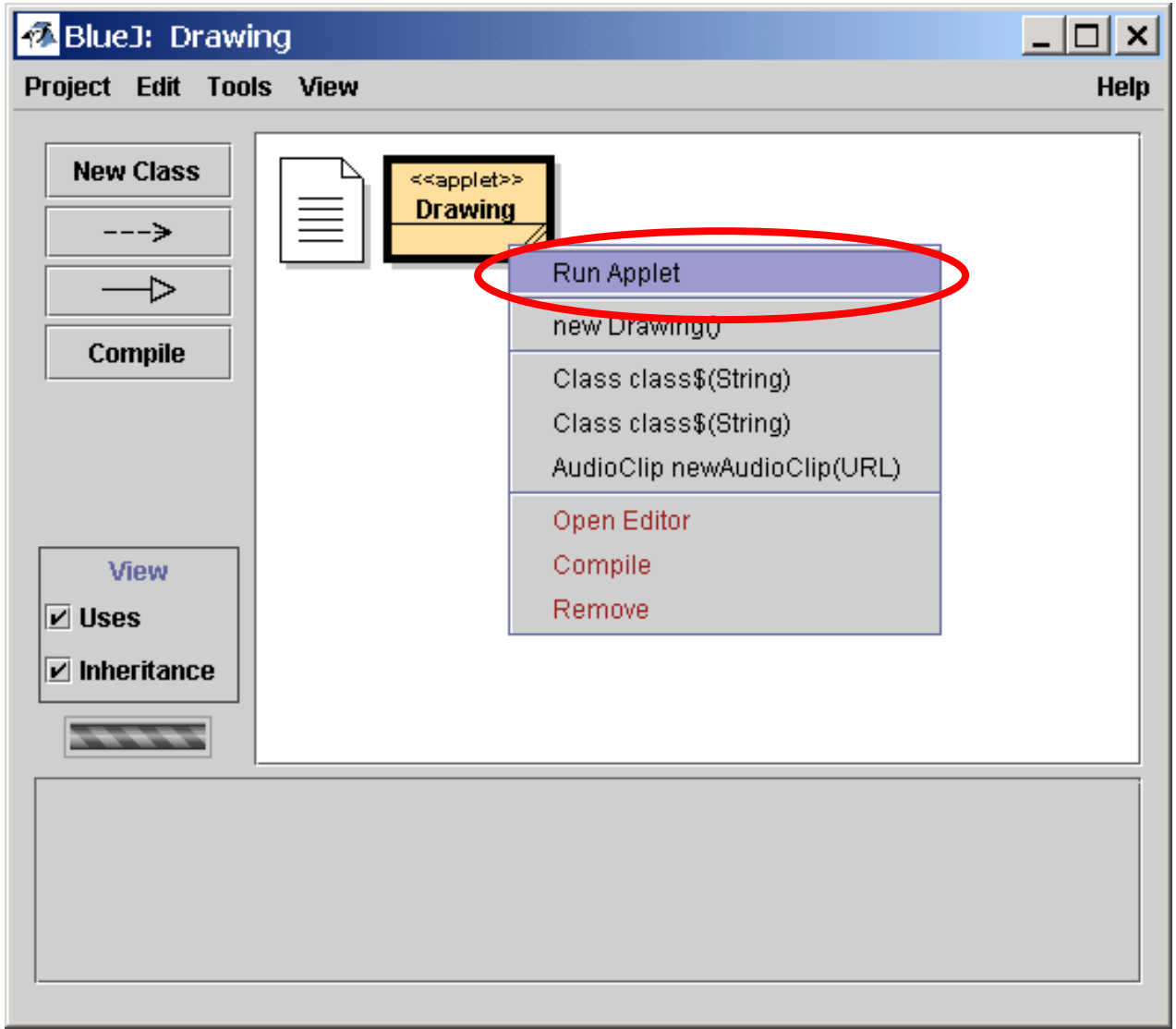
Compile Undo Cut Copy Paste Find... Find Next Close Implementation

```
import java.applet.Applet;
import java.awt.*;

// CIT 591 example
public class Drawing extends Applet {

    public void paint(Graphics g) {
        g.setColor(Color.blue);
        g.fillRect(20, 20, 50, 30);
        g.setColor(Color.red);
        g.fillRect(50, 30, 50, 30);
    }
}
```

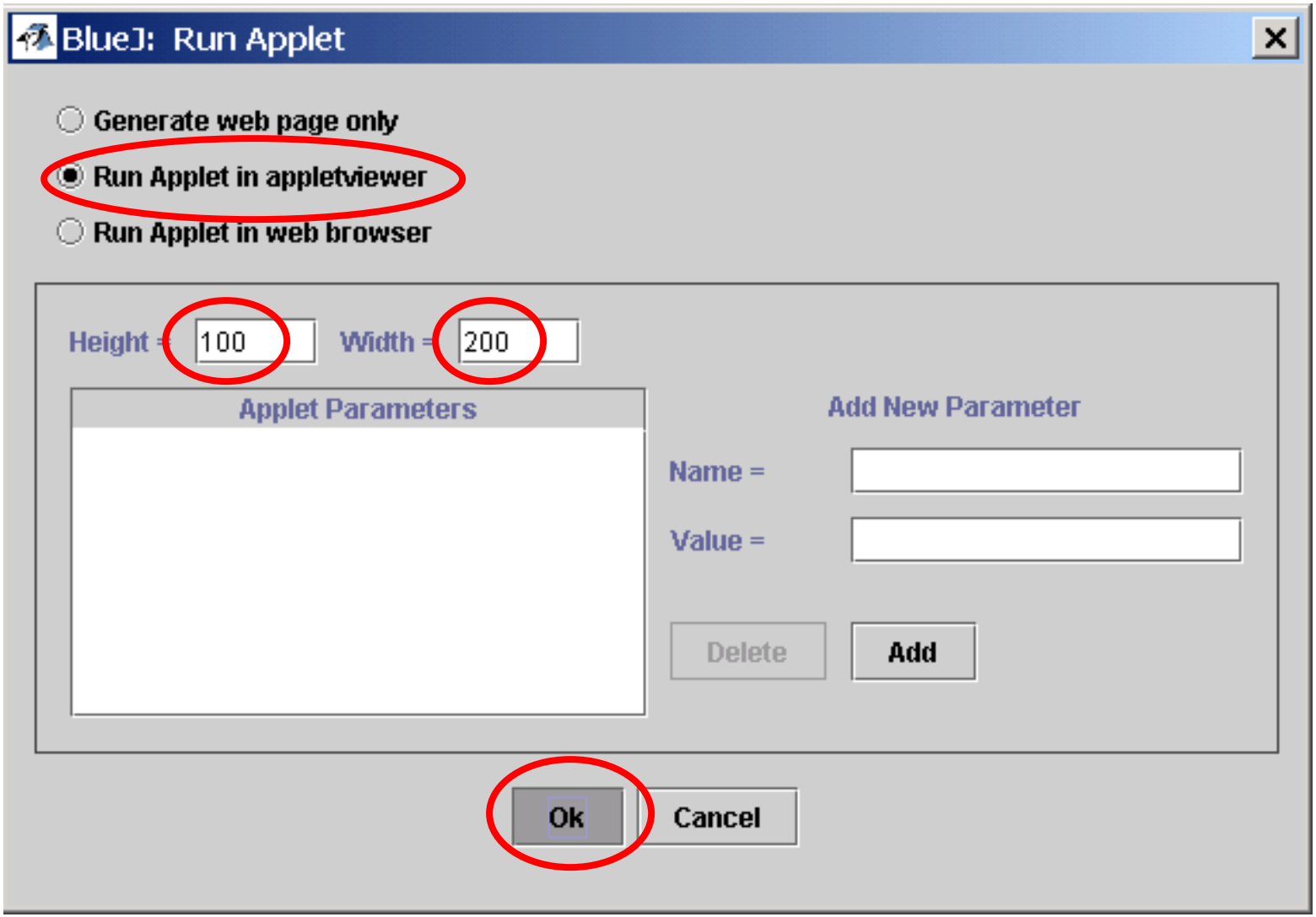
changed

BlueJ starts you with a simple class or a simple applet. You can modify what BlueJ gives you, or just replace it with your own program.

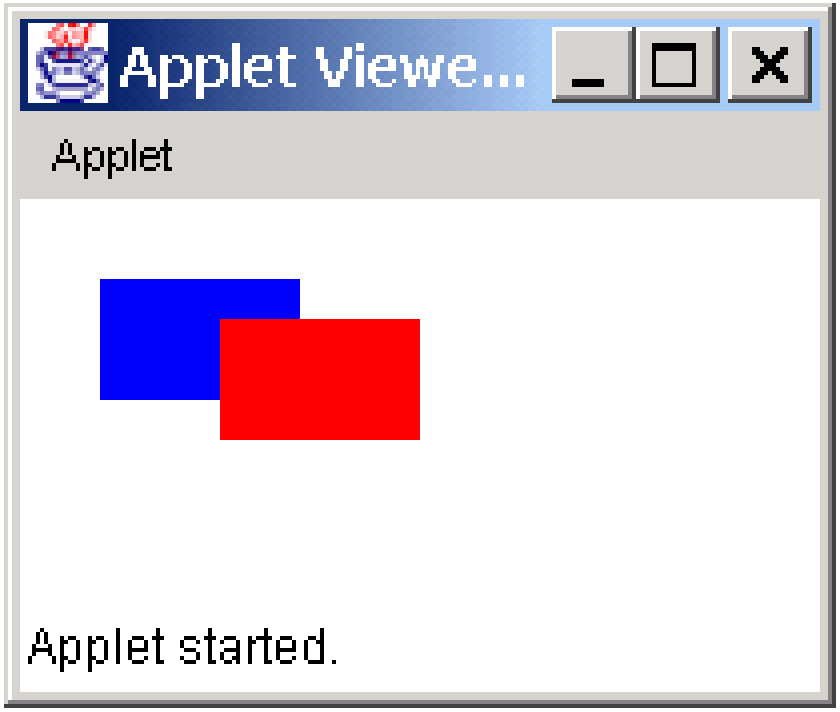
If it's an applet: Right-click the class and choose Run Applet



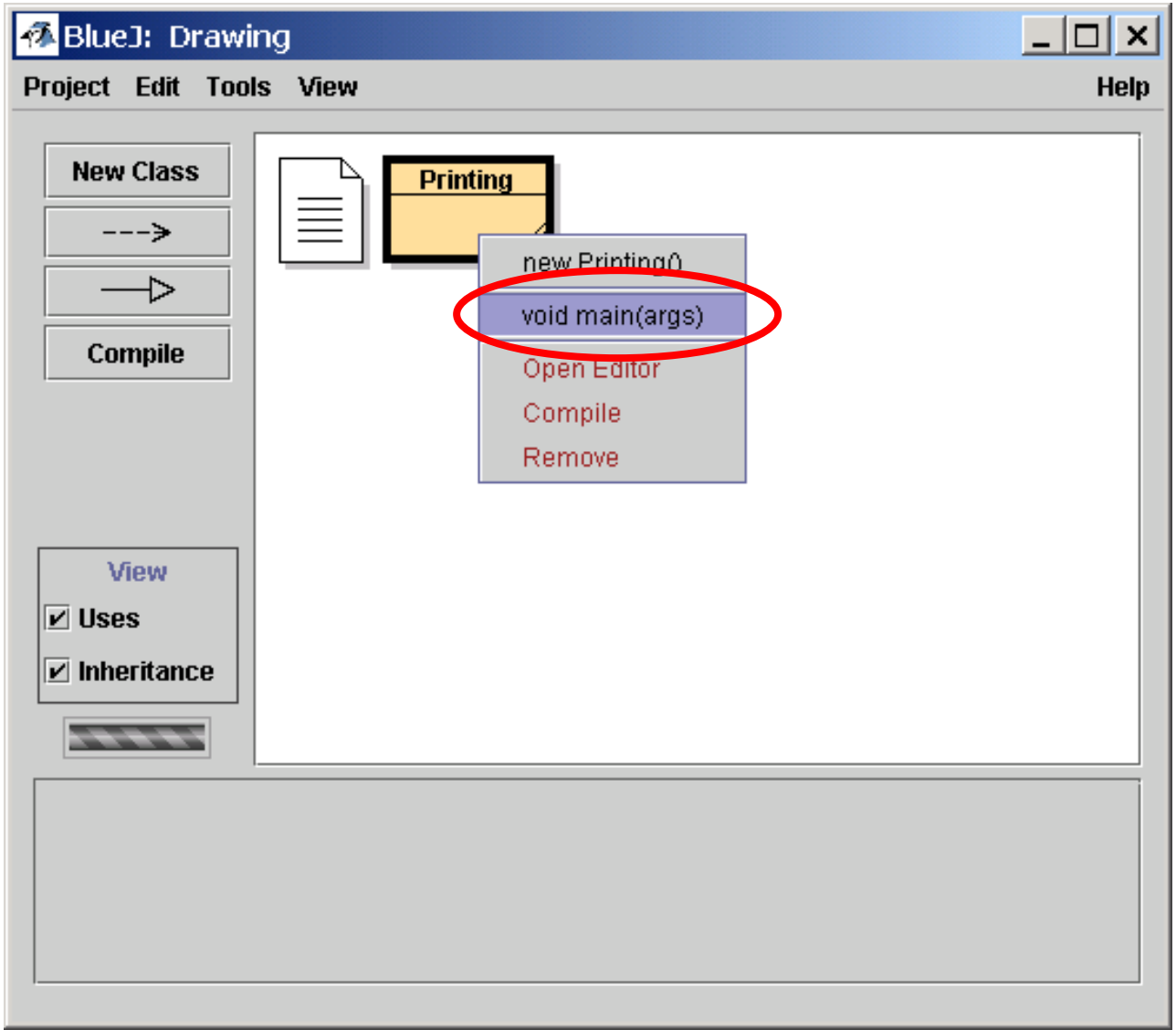
If applet: choose how to run it, and what size it should be



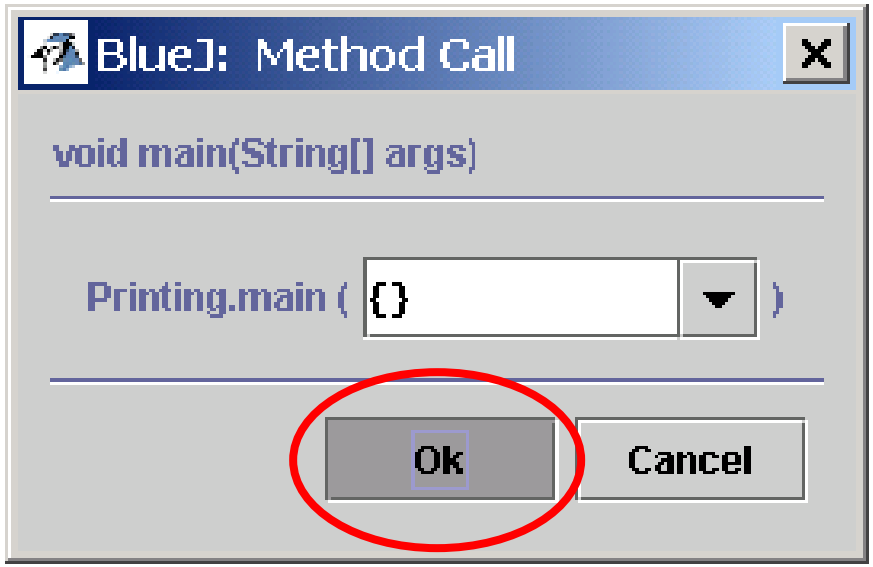
Here's the running applet!



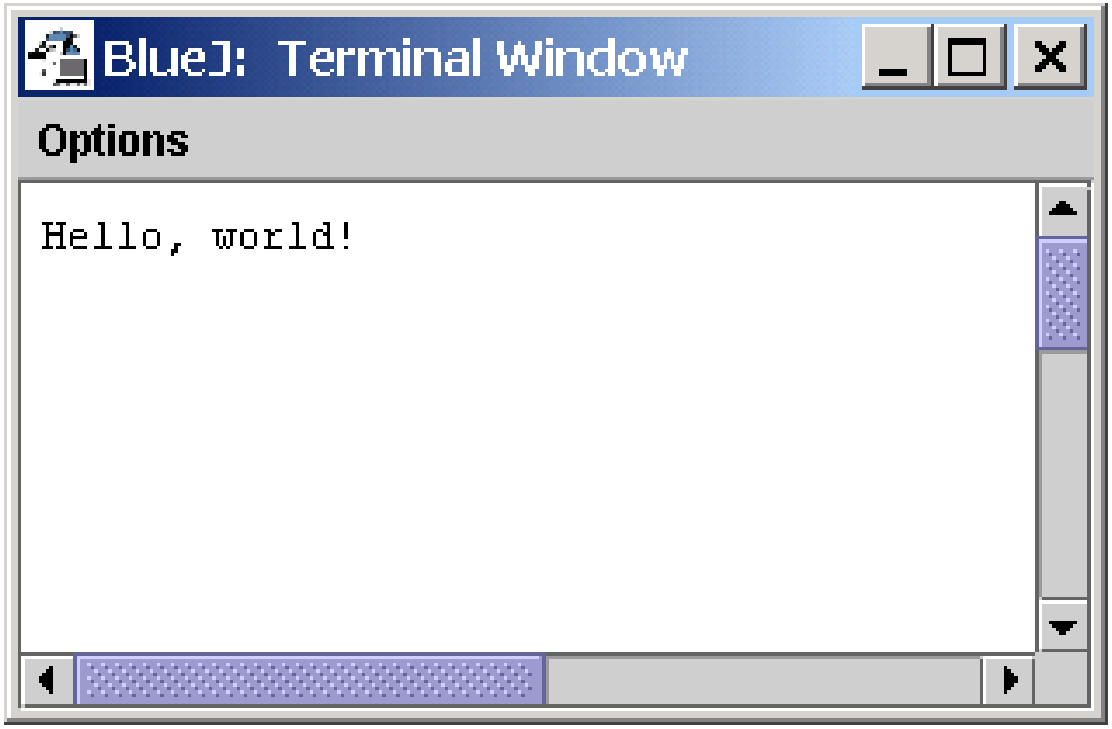
If application: right-click and choose void main(args)



Application: Just click Ok



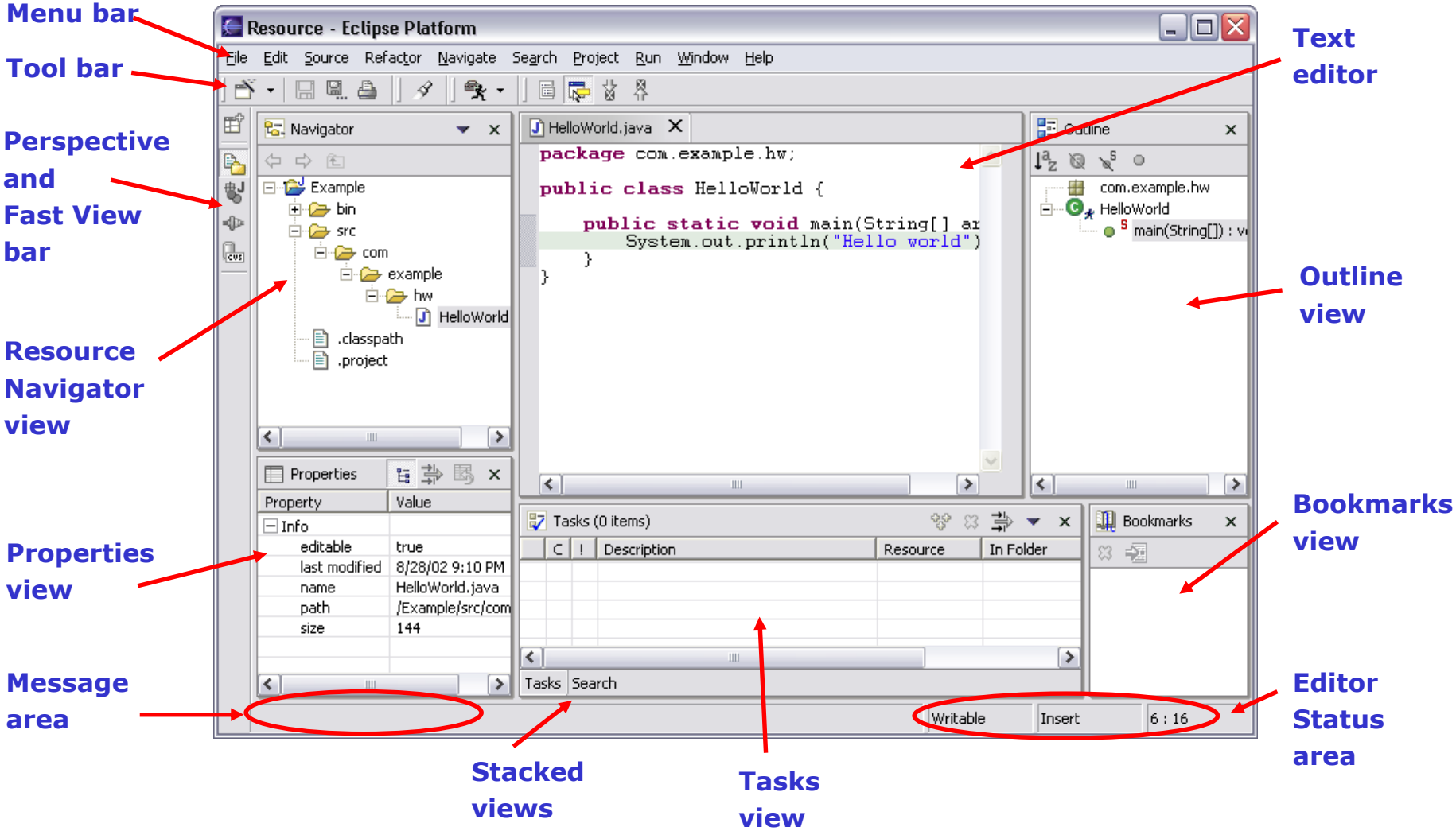
Application results



Final notes

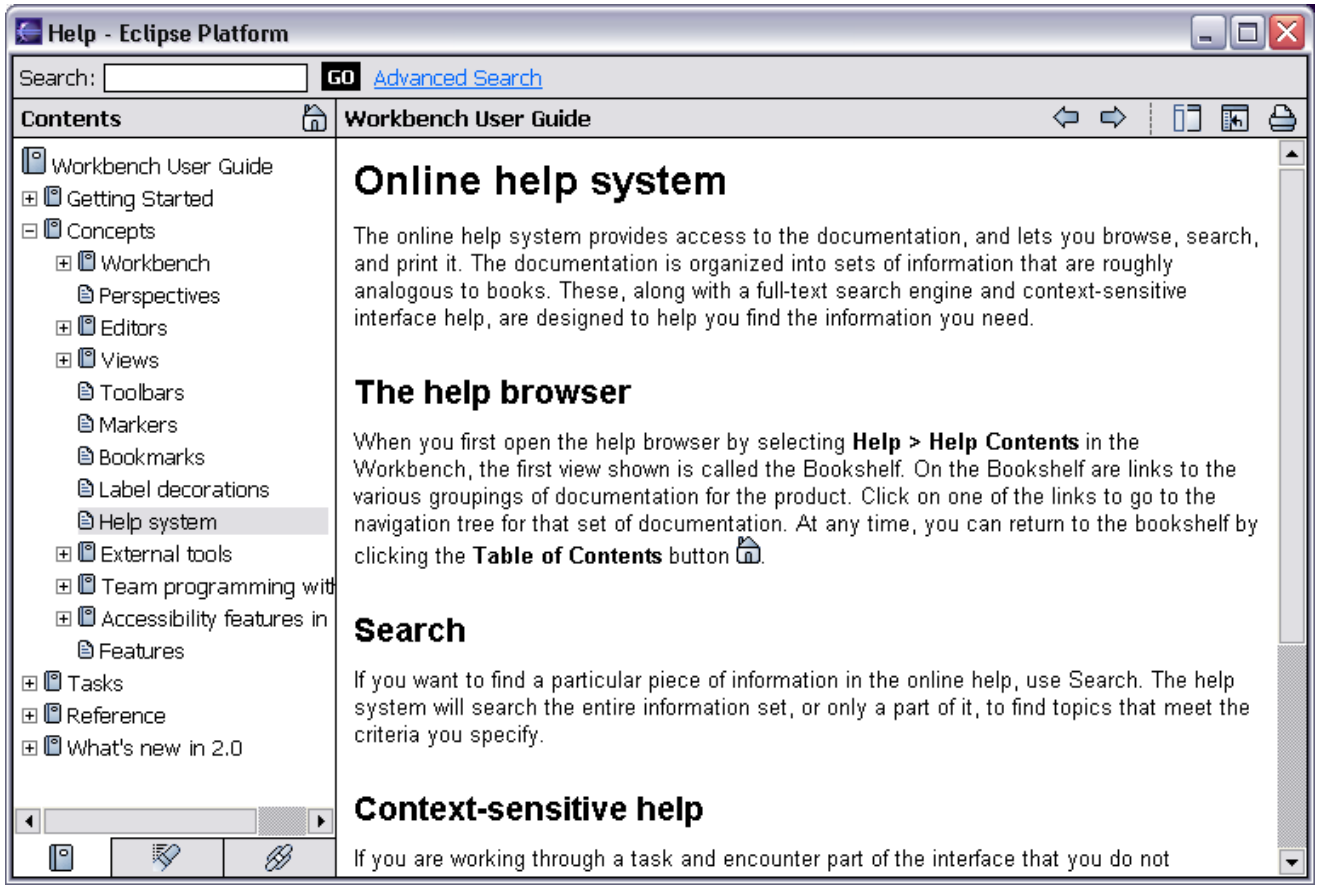
- You don't have to Save your work—BlueJ does that automatically for you when you compile
- If you quit BlueJ and come back later, use Open Project to continue where you left off
- This quick introduction does not replace working through the tutorial!

Eclipse - Workbench Terminology



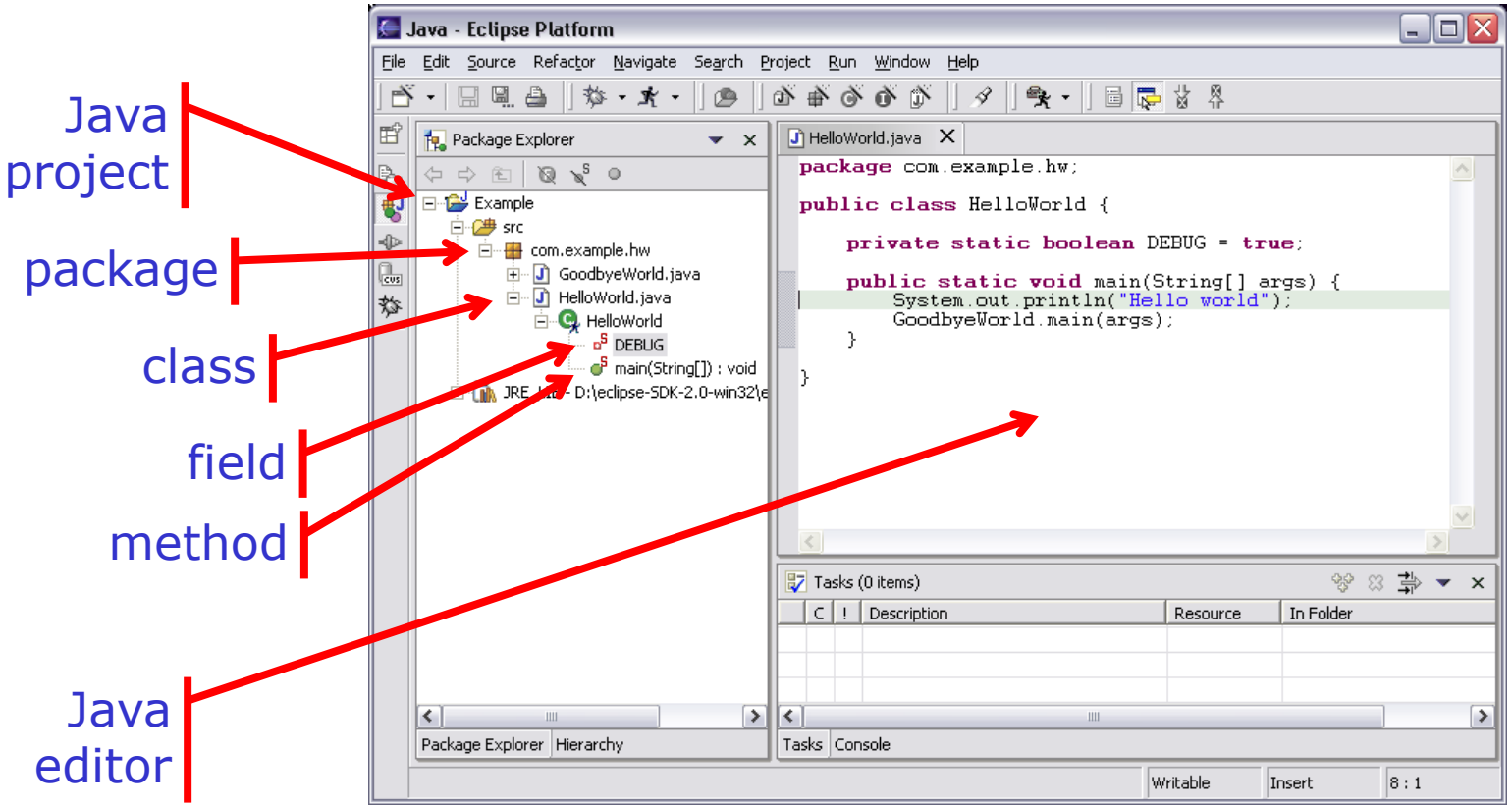
Help Component

- Help is presented in a standard web browser



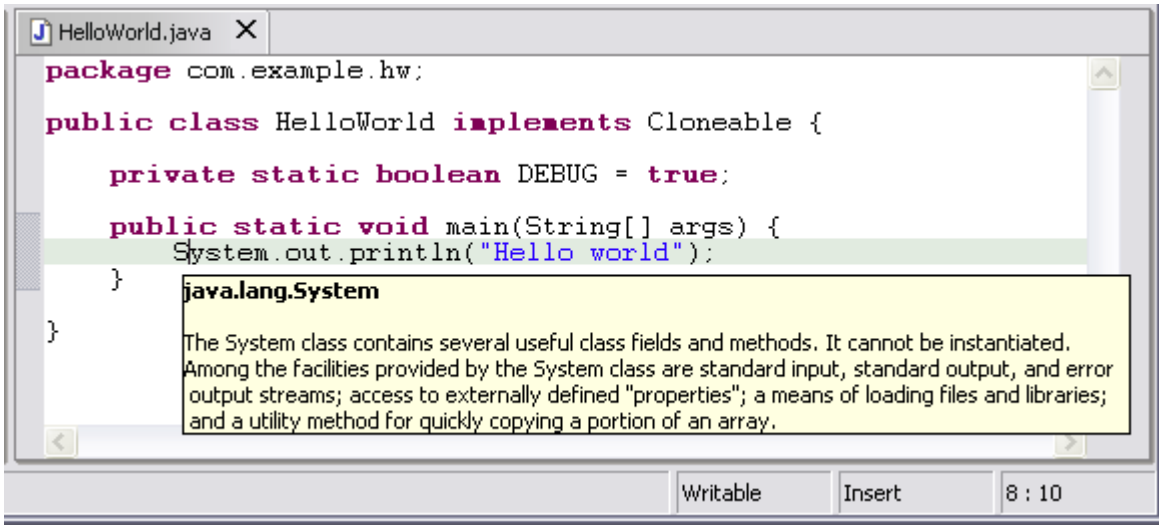
Java Perspective

- Java-centric view of files in Java projects
  - Java elements meaningful for Java programmers



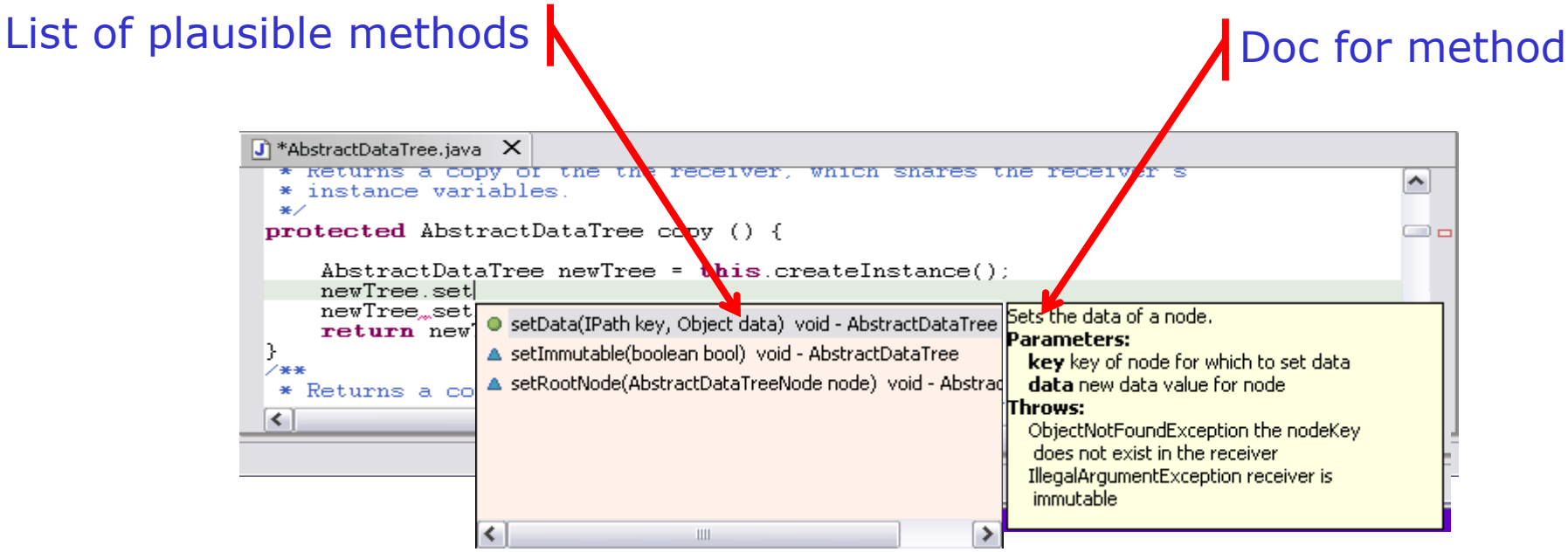
Java Editor

- Hovering over identifier shows Javadoc spec



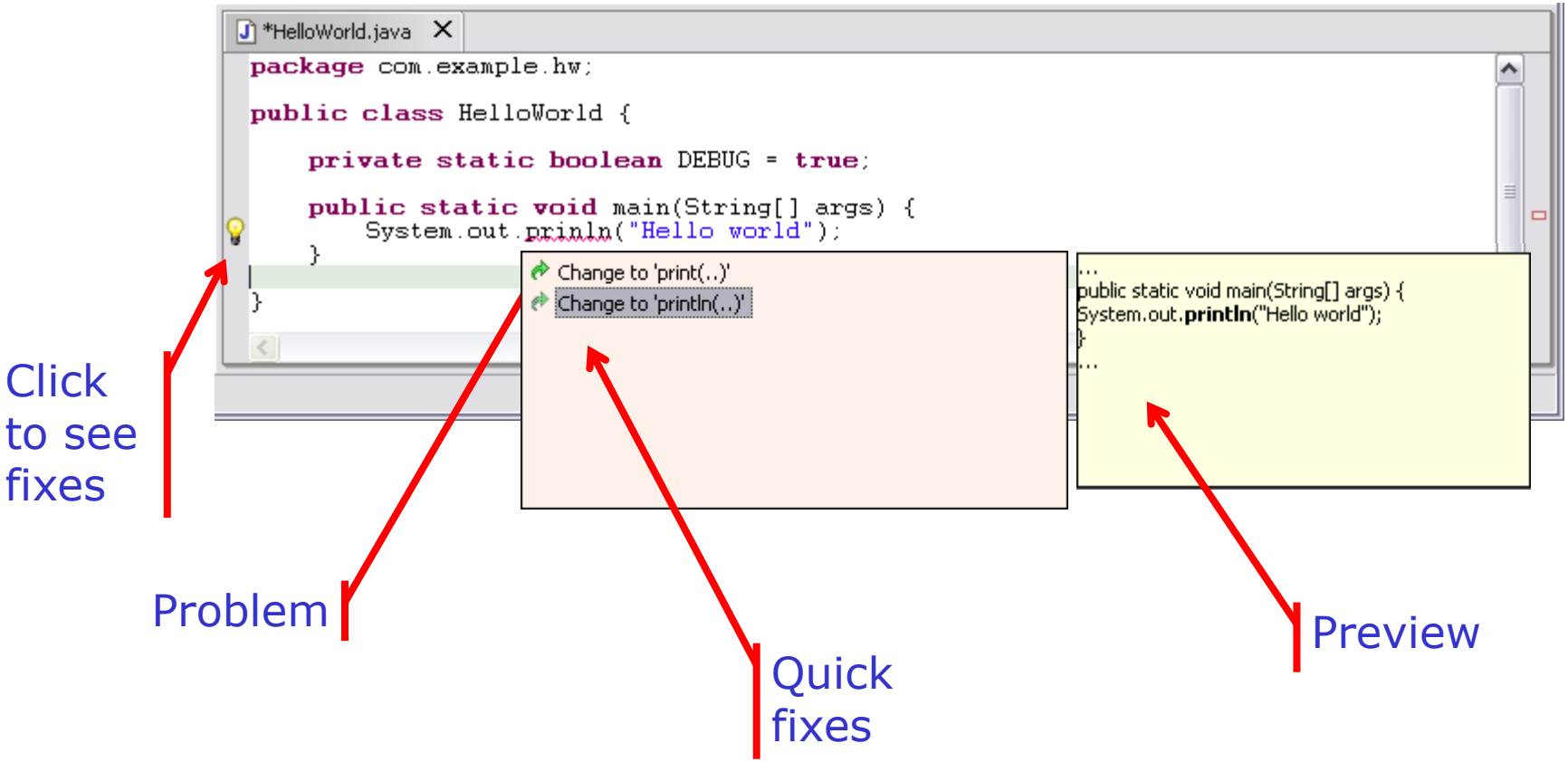
Java Editor

- Method completion in Java editor



Java Editor

- On-the-fly spell check catches errors early



Java Editor

- Java editor helps programmers write good Java code

Variable name suggestion

```
public TestResult run() {
    TestResult
    run(result)
    return res
}
/**
 * Runs the tes
 */
public void ru
    result.run(
}
/**
 * Runs the bar
```

result - TestResult
testResult - TestResult

JavaDoc code assist

```
/**
 * Runs the bare test sequence.
 * @exception
 *
 *
 *
 */
```

Exception
RuntimeException

```
public void runBare() throws Exception, RuntimeException {
    setUp();
```

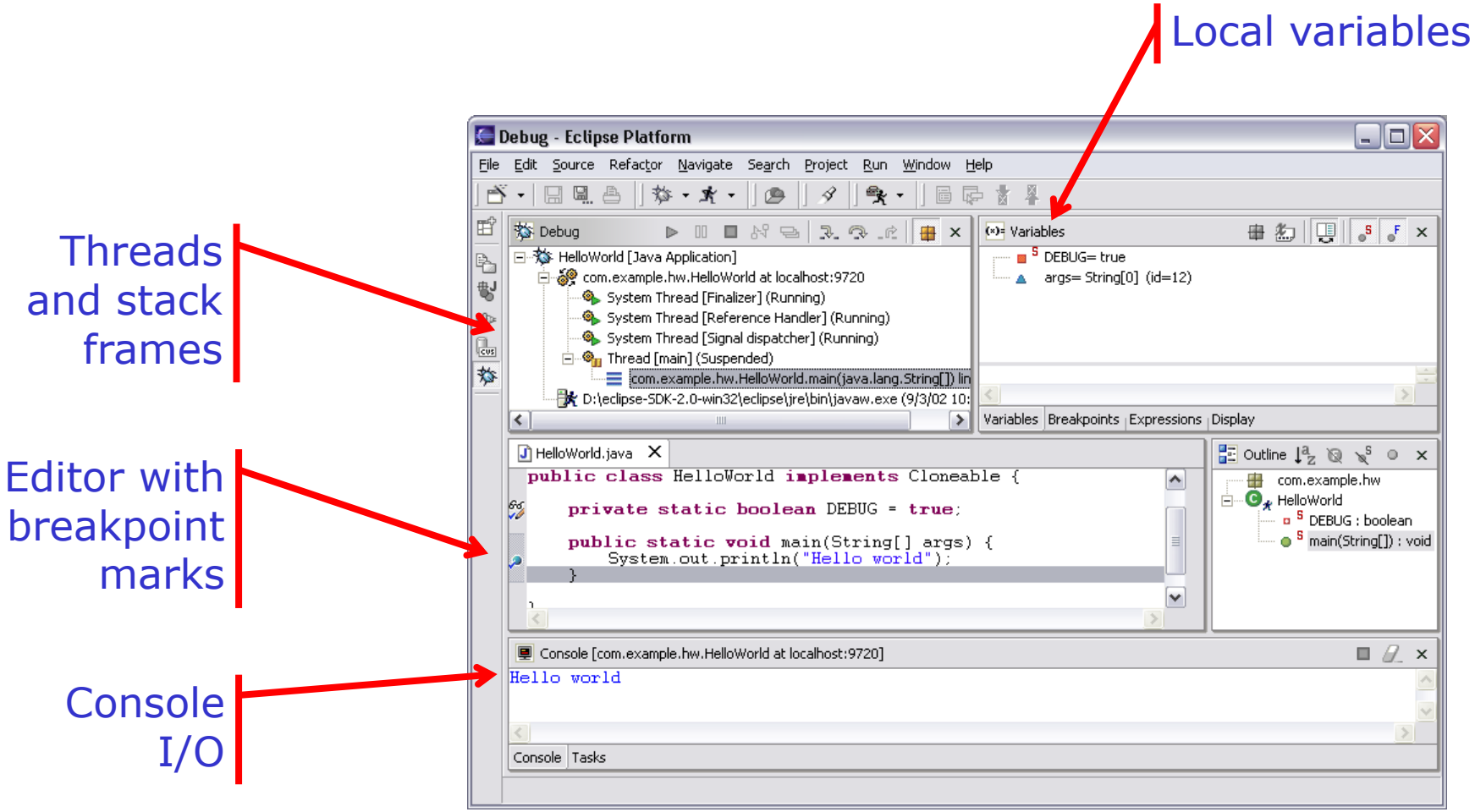
Argument hints and proposed argument names

boolean expected, boolean actual

```
assertEquals(expected, actual);
```

Eclipse Java Debugger

- Run or debug Java programs



Eclipse Java Debugger

- Run Java programs
  - In separate target JVM (user selectable)
  - Console provides stdout, stdin, stderr
- Debug Java programs
  - Full source code debugging
- Debugger features include
  - Method and exception breakpoints
  - Conditional breakpoints
  - Watchpoints
  - Step over, into, return; run to line
  - Inspect and modify fields and local variables

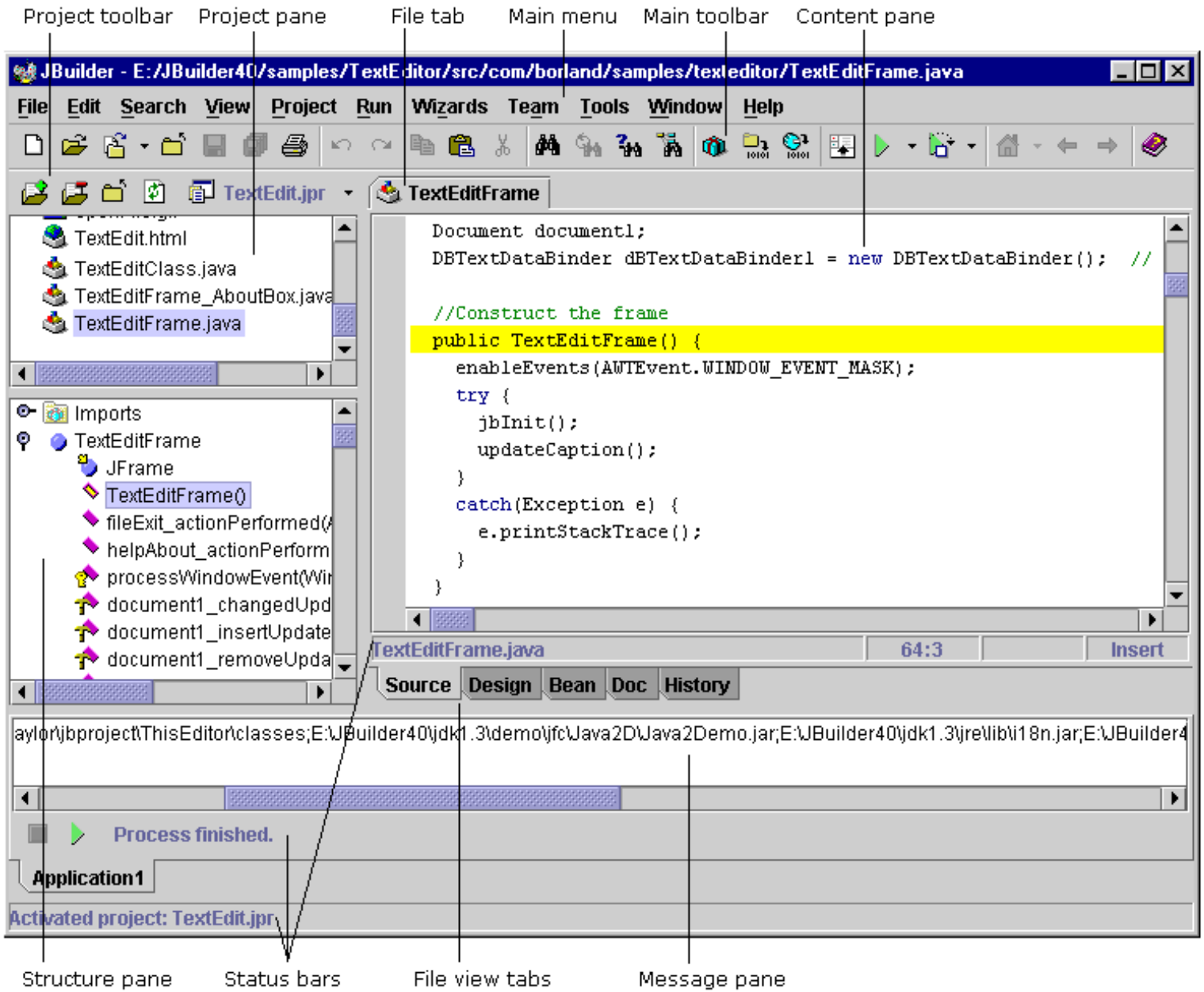
DrJava

- DrJava is a lightweight development environment for writing Java programs.
- It is designed primarily for students, providing an intuitive interface and the ability to interactively evaluate Java code.
- It also includes powerful features for more advanced users.
- DrJava is available for free and it is under active development by the JavaPLT group at Rice University.

JBuilder Basics

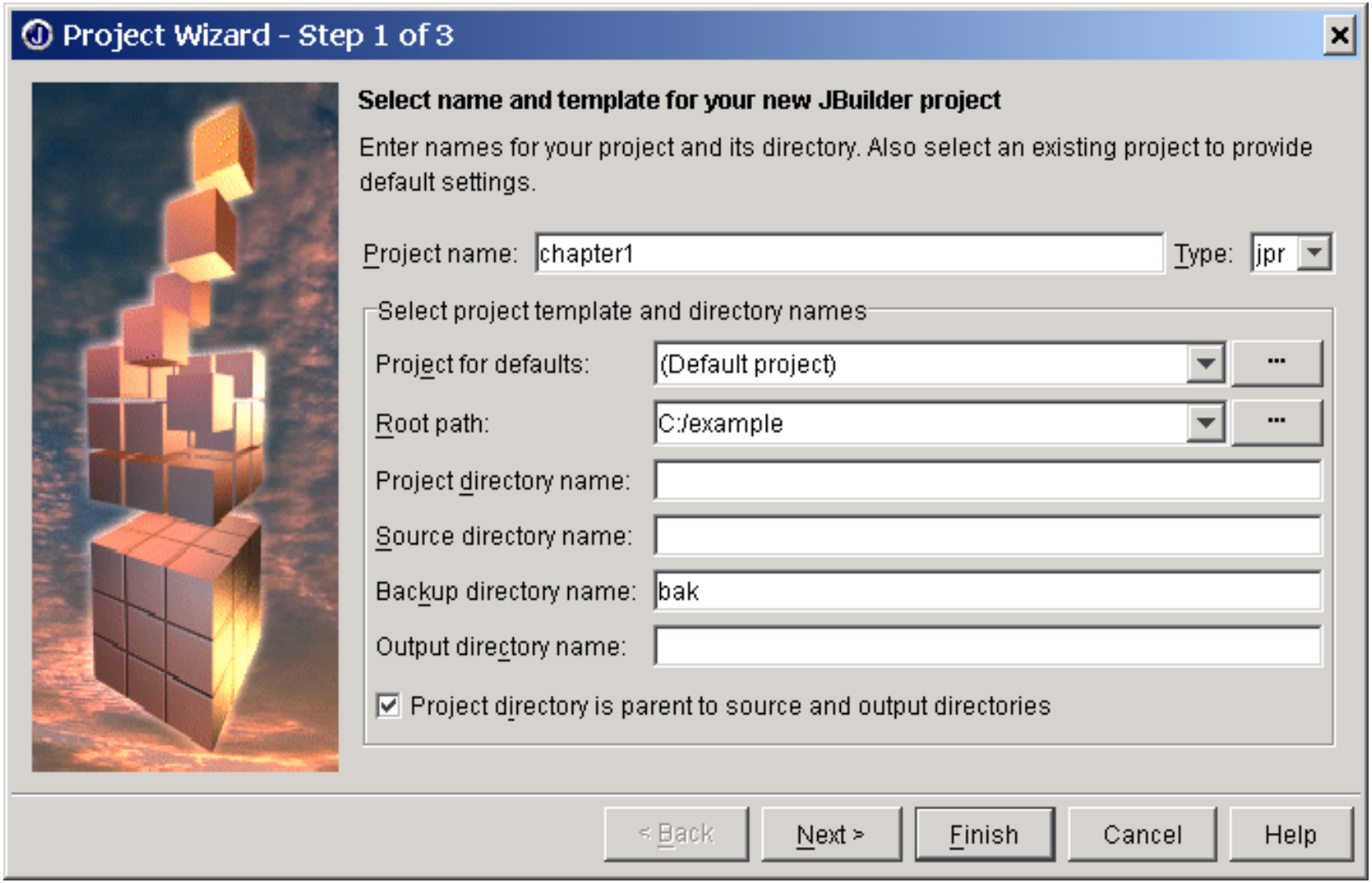
- ☞JBuilder IDE Interface
- ☞Create a JBuilder project
- ☞Create Java programs
- ☞Compile and run Java programs

JBuilder IDE Interface



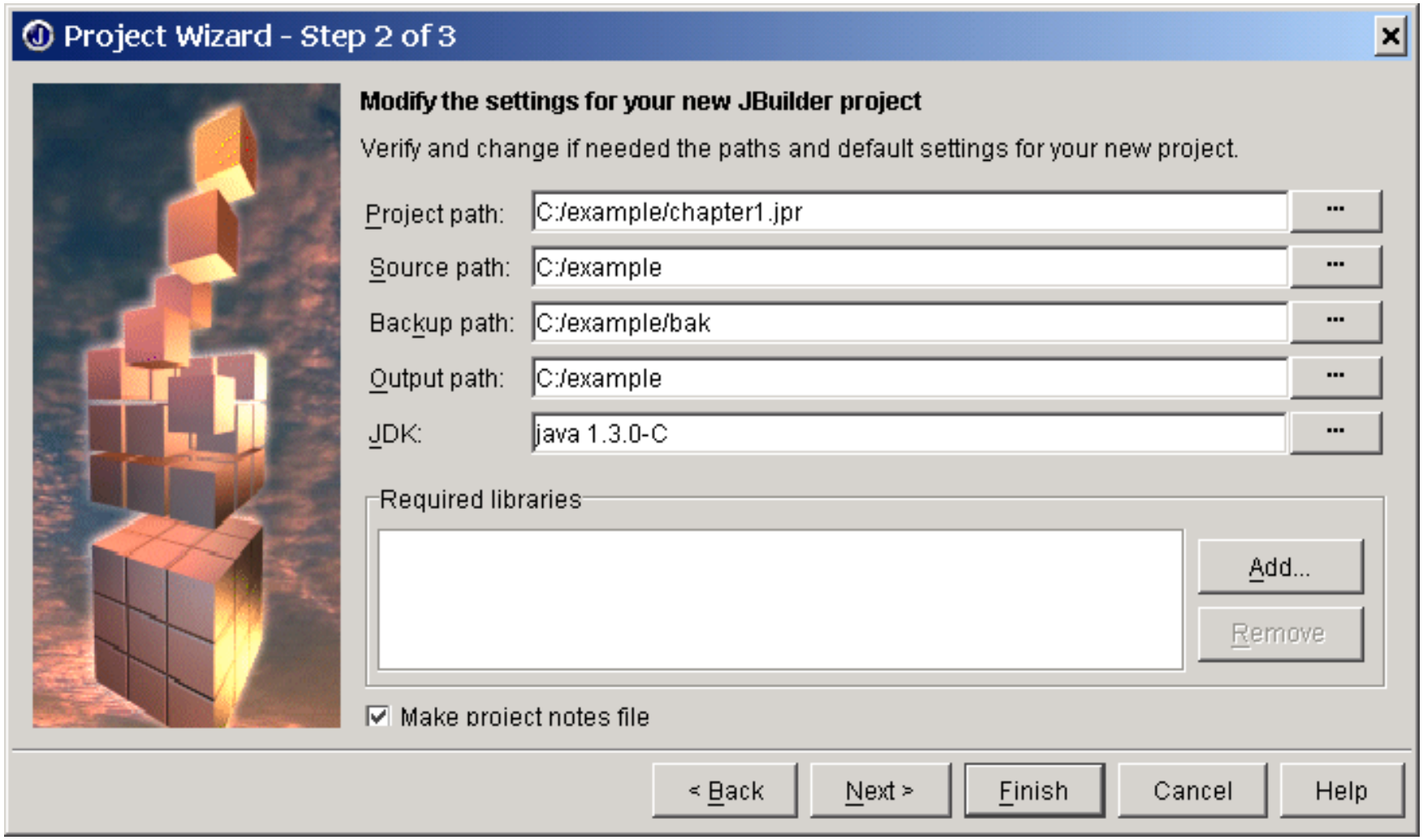
Creating a JBuilder project

Choose File, New Project to display the project wizard



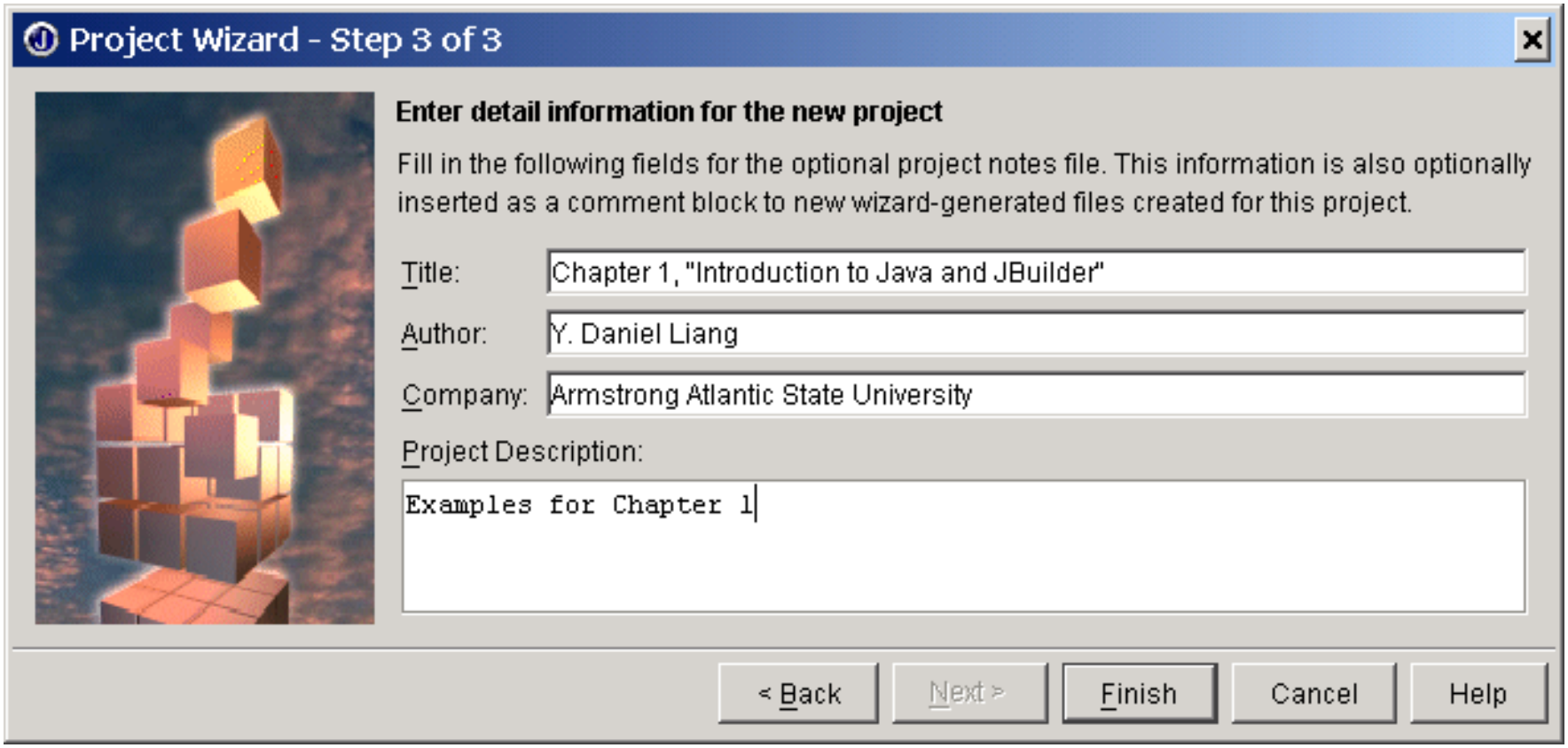
Creating a JBuilder project, cont.

Verify Project path, source path, backup path, outpath



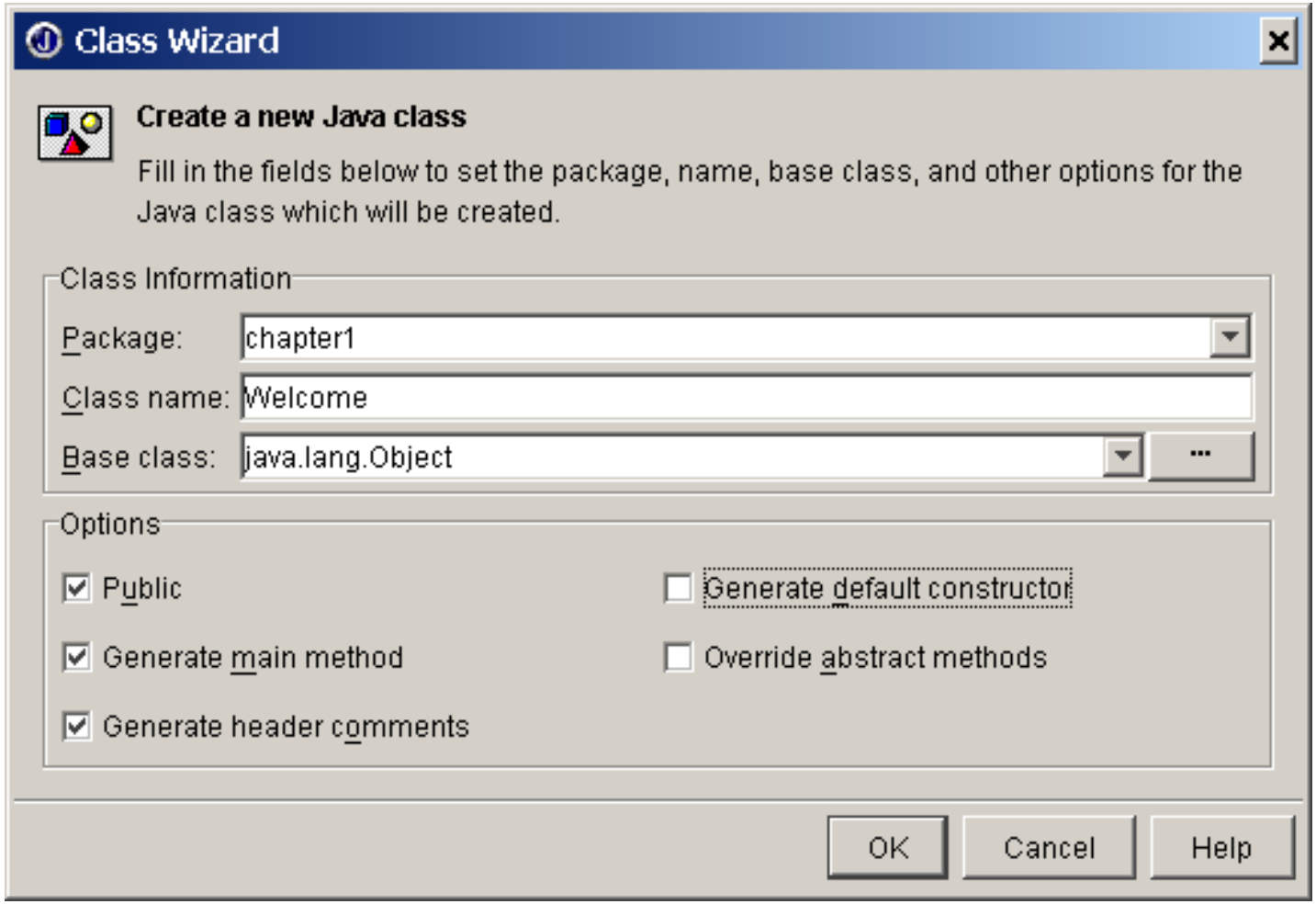
Creating a JBuilder project, cont.

Optional project description

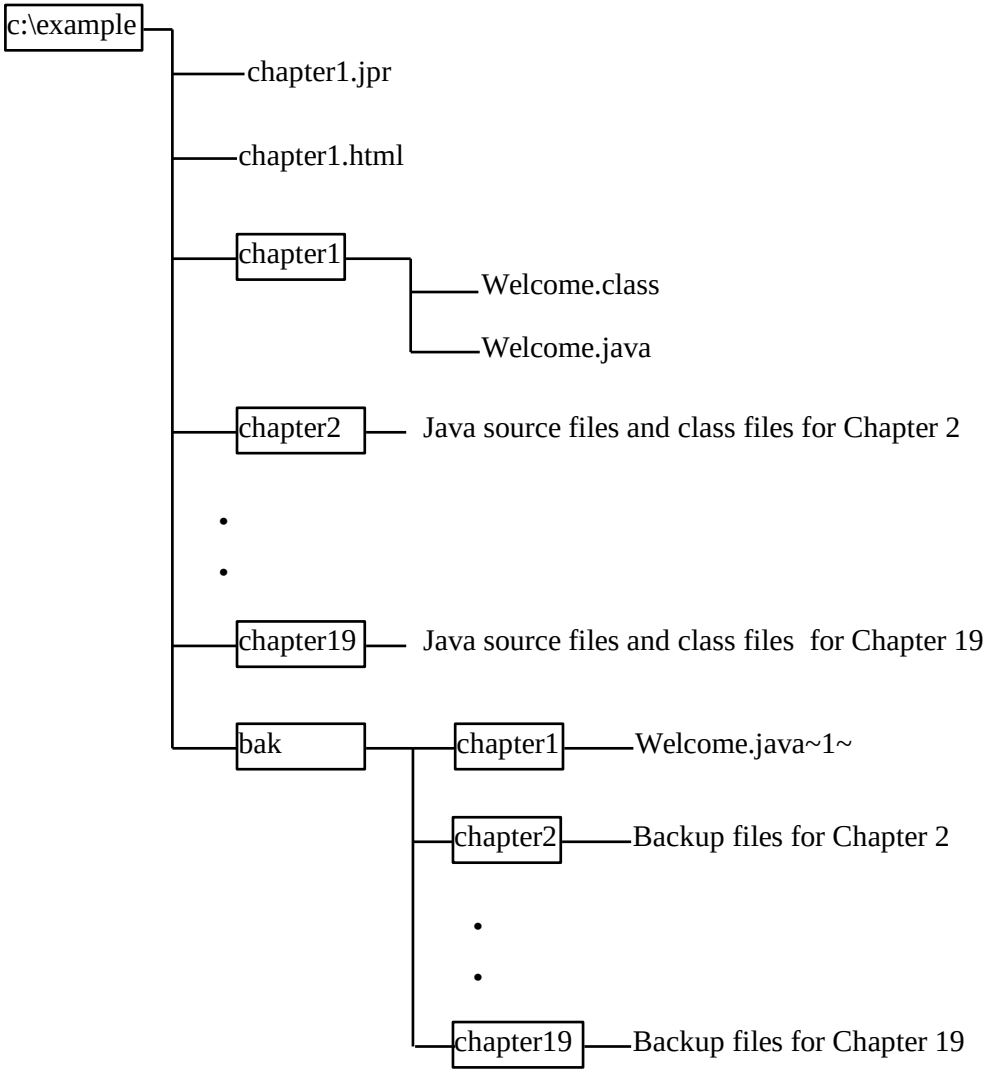


Creating a Java Program

Choose File, New Class to display the class wizard



Compiling and Running a Program



Where are the files stored in the directory?

Java Data types

- **Data types** is a classification in programming languages which indicates the type of value a variable can store. Different types of values are there & different operations are there for every data type.
- A data type is a classification of data, which can store a specific type of information. Each variable is assigned a data type that determines what type of data the variable may contain.

Classification of Java data types

- **Data types in Java are classified into two types: Primitive—which include Integer, Character, Boolean, and Floating Point.**
- **Non-primitive—which include Classes, Interfaces, and Arrays.**

Primitive Data Types

- The primitive data types are the basic data types that are available in most of the programming languages. The primitive data types are used to represent single values.
- Primitive types are declaration types and the primitive type is already defined by the java programming language.
- Variables of these data types store the actual data or the primitive value. They are passed by value. There are 8 primitive datatypes in Java:- int, char, byte, short, long, float, double and boolean.

Primitive Data Types

The eight primitive data types supported by the Java programming language are:

· **byte**: The byte data type is an 8-bit signed integer.

It has a minimum value of -128 and a maximum value of 127 (inclusive).

The byte data type can be useful for saving memory in large arrays, where the memory savings actually matters.

They can also be used in place of int where their limits help to clarify your code;

the fact that a variable's range is limited can serve as a form of documentation.

· **short**: The short data type is a 16-bit signed integer.

It has a minimum value of -32,768 and a maximum value of 32,767 (inclusive).

As with byte, the same guidelines apply: you can use a short to save memory in large arrays, in situations where the memory savings actually matters.

Primitive Data Types

- **int**: The int data type is a 32-bit signed integer. It has a minimum value of -2,147,483,648 and a maximum value of 2,147,483,647 (inclusive). For integral values, this data type is generally the default choice unless there is a reason (like the above) to choose something else. This data type will most likely be large enough for the numbers your program will use, but if you need a wider range of values, use longinstead.
- **long**: The long data type is a 64-bit signed integer. It has a minimum value of -9,223,372,036,854,775,808 and a maximum value of 9,223,372,036,854,775,807 (inclusive). Use this data type when you need a range of values wider than those provided by int.

Primitive Data Types

int: The int data type is a 32-bit signed integer. It has a minimum value of -2,147,483,648 and a maximum value of 2,147,483,647 (inclusive). For integral values, this data type is generally the default choice unless there is a reason (like the above) to choose something else. This data type will most likely be large enough for the numbers your program will use, but if you need a wider range of values, use longinstead.

· **long**: The long data type is a 64-bit signed integer. It has a minimum value of -9,223,372,036,854,775,808 and a maximum value of 9,223,372,036,854,775,807 (inclusive). Use this data type when you need a range of values wider than those provided by int.

Primitive Data Types

· **float**: The float data type is a single-precision 32-bit IEEE 754 floating point. Its range of values is beyond the scope of this discussion, but is specified in the Floating-Point Types, Formats, and Values section of the Java Language Specification. As with the recommendations for byte andshort, use a float (instead of double) if you need to save memory in large arrays of floating point numbers.

This data type should never be used for precise values, such as currency. For that, you will need to use the java.math.BigDecimal class instead. Numbers and Strings covers BigDecimal and other useful classes provided by the Java platform.

· **double**: The double data type is a double-precision 64-bit IEEE 754 floating point. Its range of values is beyond the scope of this discussion, but is specified in the Floating-Point Types, Formats, and Values section of the Java Language Specification. For decimal values, this data type is generally the default choice. As mentioned above, this data type should never be used for precise values, such as currency.

Primitive Data Types

- **boolean**: The boolean data type has only two possible values: true andfalse. Use this data type for simple flags that track true/false conditions. This data type represents one bit of information, but its "size" isn't something that's precisely defined.
- **char**: The char data type is a single 16-bit Unicode character. It has a minimum value of '\u0000' (or 0) and a maximum value of '\uffff' (or 65,535 inclusive).

Primitive Data Types

The size and range of Data Types:

Data Type	Size	Range
int	32-bit	-2,147,483,648 to 2,147,483,647
short	16-bit	-32,768 to 32,767
long	64-bit	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
byte	8-bit	-128 to 127
float	32-bit	1.40129846432481707e-45 to 3.40282346638528860e+38
double	64-bit	4.94065645841246544e-324d to 1.79769313486231570e+308d
char	16-bit	0 to 65,535
boolean	1-bit	True or False only

Default Values

Data Type	Default Value (for fields)
byte	0
short	0
int	0
long	0L
float	0.0f
double	0.0d
char	'\u0000'
String (or any object)	null
boolean	false

Non-Primitive Data Types

- Not defined by the programming language, but are instead created by the programmer.
- Sometimes called “reference variables” or “object references”; since they reference a memory location, which stores the data.
- In the Java programming language, non-primitive data types are simply called “objects”; because they are created, rather than predefined.
- Non-primitives are "Reference-types". These include class-based types, either core library defined or user defined, or array types (no class definition).

(Later – to be continued)

- **Objects** (Classes & Interfaces) and **Arrays** are the reference or non-primitive data types in Java. They are so called because they are handled “by reference” i.e. variables of their type store the address of the object or array is stored in a variable.