Compiling a Java Program or Class

- A Java program consists of one or more classes, which must be compiled before running the program.
- Each class should be in a separate file.
- The name of the file should be the same as the name of the class.

Compiling and Running

- Use an IDE(integrated development environment) which combines a text editor with commands for compiling and running Java programs.
- When a Java program is compiled, the byte-code version of the program has the same name, but the ending is changed from **`.java`** to **`.class`**.

Compiling and Running

- A Java program can involve any number of classes.
- The class to run will contain the words

```
public static void main(String[] args)
```

somewhere in the file

Testing and Debugging

- Eliminate errors by avoiding them in the first place.
  - Carefully design classes, algorithms and methods.
  - Carefully code everything into Java.
- Test your program with appropriate test cases (some where the answer is known), discover and fix any errors, then retest.

Software Reuse

- Programs not usually created entirely from scratch
- Most contain components which already exist
- Reusable classes are used
  - Design class objects which are general
  - Java provides many classes
  - Note documentation on following slide

Basic Computation

Chapter 2

*JAVA: An Introduction to Problem Solving & Programming, 8*th Ed. By Walter Savitch
ISBN 0134462033

Variables

- Variables store data such as numbers and letters.
  - Think of them as places to store data.
  - They are implemented as memory locations.
- The data stored by a variable is called its value.
  - The value is stored in the memory location.
- Its value can be changed.

Naming and Declaring Variables

- Choose names that are helpful such as `count` or `speed`, but not `c` or `s`.
- When you declarea variable, you provide its name and type.

```
int numberOfBaskets,eggsPerBasket;
```

- A variable's typedetermines what kinds of values it can hold (`int`, `double`, `char`, etc.).
- A variable must be declared before it is used.

Syntax and Examples

- Syntax

  `type variable_1, variable_2, …;`

  (`variable_1` is a generic variable called a syntactic variable)

- Examples

```
int styleChoice, numberOfChecks;
double balance, interestRate;
char jointOrIndividual;
```

Data Types

- A class type is used for a class of objects and has both data and methods.
  - `"Java is fun"` is a value of class type `String`
- A primitive type is used for simple, nondecomposable values such as an individual number or individual character.
  - `int`, `double`, and `char` are primitive types.

Primitive Types

FIGURE 2.1 Primitive Type

Type Name	Kind of Value	Memory Used	Range of Values
`byte`	Integer	1 byte	–128 to 127
`short`	Integer	2 bytes	–32,768 to 32,767
`int`	Integer	4 bytes	–2,147,483,648 to 2,147,483,647
`long`	Integer	8 bytes	–9,223,372,036,8547,75,808 to 9,223,372,036,854,775,807
`float`	Floating-point	4 bytes	$\pm 3.40282347 \times 10^{+38}$ to $\pm 1.40239846 \times 10^{-45}$
`double`	Floating-point	8 bytes	$\pm 1.79769313486231570 \times 10^{+308}$ to $\pm 4.9406565841246544 \times 10^{-324}$
`char`	Single character (Unicode)	2 bytes	All Unicode values from 0 to 65,535
`boolean`		1 bit	True or false

Java Identifiers

- An identifier is a name, such as the name of a variable.
- Identifiers may contain only
  - Letters
  - Digits (0 through 9)
  - The underscore character (_)
  - And the dollar sign symbol ($) which has a special meaning
- The first character cannot be a digit.

Keywords or Reserved Words

- Words such as `if` are called keywords or reserved words and have special, predefined meanings.
  - Cannot be used as identifiers.
- Example keywords: `int`, `public`, `class`

Naming Conventions

- Class types begin with an uppercase letter (e.g. `String`).
- Primitive types begin with a lowercase letter (e.g. `int`).
- Variables of both class and primitive types begin with a lowercase letters (e.g. `myName`, `myBalance`).
- Multiword names are "punctuated" using uppercase letters.

Where to Declare Variables

- Declare a variable
  - Just before it is used or
  - At the beginning of the section of your program that is enclosed in `{}`.

```
public static void main(String[] args)
{ /* declare variables here */
    . . .
}
```

Primitive Types

- Four integer types (`byte`, `short`, `int`, and `long`)
  - `int` is most common
- Two floating-point types (`float` and `double`)
  - `double` is more common
- One character type (`char`)
- One boolean type (`boolean`)

Examples of Primitive Values

- Integer types

```
0  -1  365  12000
```

- Floating-point types

```
0.99  -22.8  3.14159 5.0
```

- Character type

```
'a'  'A'  '#'  ' '
```

- Boolean type

```
true  false
```

Assignment Statements

- An assignment statement is used to assign a value to a variable.

```
answer = 42;
```

- The "equal sign" is called the assignment operator.

Assignment Statements

- Syntax

  `variable = expression`

  where **`expression`** can be another variable, a literalor constant(such as a number), or something more complicated which combines variables and literals using operators
  (such as + and -)

Initializing Variables

- A variable that has been declared, but no yet given a value is said to be uninitialized.
- Uninitialized class variables have the value `null`.
- Uninitialized primitive variables may have a default value.
- It's good practice not to rely on a default value.

Assignment Evaluation

- The expression on the right-hand side of the assignment operator (=) is evaluated first.
- The result is used to set the value of the variable on the left-hand side of the assignment operator.

```
eggsPerBasket = eggsPerBasket - 2;
```

Simple Input

- Sometimes the data needed for a computation are obtained from the user at run time.
- Keyboard input requires

  `import java.util.Scanner`

  at the beginning of the file.

Simple Input

- Data can be entered from the keyboard using

```
Scanner key = new Scanner(System.in);
```

followed, for example, by

```
eggsPerBasket = key.nextInt();
```

which reads one `int` value from the keyboard and assigns it to `eggsPerBasket`.

Simple Screen Output

```
System.out.println("The count is " + count);
```

- Outputs the string literal **`"the count is "`**
- Followed by the current value of the variable **`count`**.

`nextInt()` & `nextDouble()` Method

- The `nextInt()` method
  - reads an integer

- The `nextDouble()` method
  - reads a decimal number

`next()` & `nextLine()` Method

- The `next()` method reads
  - the string before a space

- The `nextLine()` method reads
  - The remainder of the current line,
  - Even if it is empty.

Constants

- Literal expressions such as `2`, `3.7`, or `'y'` are called constants.
- Integer constants can be preceded by a `+` or `-` sign, but cannot contain commas.
- Floating-point constants can be written
  - With digits after a decimal point or
  - Using e notation.

Named Constants

- Java provides mechanism to …
  - Define a variable
  - Initialize it
  - Fix the value so it cannot be changed

```
public static final Type Variable = Constant;
```

Example

- `public static final double PI = 3.14159;`
- `public static final String MOTTO = "The customer is always right";`

By convention, uppercase letters are used for constants.

Using Named Constants

- To avoid confusion, always name constants (and variables).

```
area = PI * radius * radius;
```

  is clearer than

```
area = 3.14159 * radius * radius;
```

- Place constants near the beginning of the program.