

CSC 125

Object Oriented Programming

Ch02_1_Elementary
programming

Dr. Fadi Alzhouri



Variables

- A **variable** is used to **store** a piece of data for processing.
- It is called variable because you can **change** the **value** stored.
- In reality, a variable is a named storage (memory) location.
- Each variable has a **name**, a **data type**, an **address**, and a **value**.
- The name (aka. **identifier**) is needed to **uniquely** identify and reference each variable.
 - You can use its name to save or retrieve its value.
- A variable can only store value of a particular **data type**.
 - For example, an **int** variable can store an integer value such as 12 but not a character such as "A".

Variables (cont.)

- The frequently-used Java data types are:
 - `int`: to store integers such as 34 and -45.
 - `double`: to store real numbers such as 34.58, -58.9
 - `char`: to store a **single** character, such as 'a', '8'. A char is enclosed by a pair of **single quotes**.
 - `string`: to store texts such as "Hello". Strings are enclosed within a **pair of double** quotes.
- The syntax for declaring a variable is:

`datatype variableName;`

or `datatype identifier;`

- To declare multiple variables of the same type using one line:

`datatype identifier1, identifier2,..., identifierN;`

Variables (cont.)

- Examples:

```
int age; // declare an integer variable named age.  
char grade; // declare a character variable named grade.
```

Type	Identifier /name	Address	Value
int	age	→ 34598112	• 4
double	width	→ 34598124	• 5.8
char	grade	→ 34598148	• A
string	university	→ 34598164	• Gust

String takes up many memory locations. For now, you can think of the first location of the string.

Identifiers (or Names)

- **Identifiers** are the names that identify variables, methods, and classes.
- All identifiers must obey the following rules in Java:
 - It is a sequence of characters that consists of **letters**, **digits**, underscores (**_**), and dollar signs (**\$**).
 - It must start with a letter, an underscore (**_**), or a dollar sign (**\$**) but it cannot start with a digit.
 - It cannot be a **reserved word** (such as **class**).
 - An identifier cannot be **true**, **false**, or **null**.
 - It can be of any length.

Naming Conventions

- Choose meaningful and descriptive names.
- Class names:
 - **Capitalize** the first letter of each word in the name.
 - For example, the class name `ComputeExpression`.
- Variable name is made up of one or several words with no spaces between words.
 - The first word is in **lowercase**, while the remaining words are **initial-capitalized**.
 - This convention is called *camel-notation*.
 - For examples: `radius`, `area`, `fontSize`, `studentName`.

Naming Conventions (cont.)

- Java is case-sensitive.
- This means that identifiers (names) are treated as distinct if they differ in case.
- Example: these two are not considered the same:

```
firstName;
```

```
firstname;
```

Naming Conventions (cont.)

- Naming examples: which identifier (name) is valid?

num 1

1day

myage

\$sign

last_Name

_name

first Name

x*y

Constants

- Constant indicates a data element whose value **cannot change**.
- It is declared with the keyword **final**.
- Their values cannot be changed during program execution.
- A constant must be declared and **initialized** in the same statement.
- By convention, all letters in a constant are in uppercase.
- Syntax:

```
final datatype identifier = value;
```

- For examples:

```
final double PI = 3.14159265;    // Declare and  
initialize the constant
```

Assignment (=)

- An assignment statement evaluates the RHS (Right-Hand Side) and assigns its value to the variable of the LHS (Left-Hand Side).
- It does not mean equality. It is called **assignment operator**
- Syntax:

`variable = value;`
or `variable = expression;`

- For examples:

`int number1;`

`number1 = 5;`

`int sum;`

`sum = number1 + 1;`

Assign value

Assign expression

Print text messages:

- Use `System.out.println()` or `System.out.print()` to print text messages to the display console.
- Syntax:
 - `System.out.println(aString) ;` //prints aString, and advances the cursor to the beginning of the next line.
 - `System.out.print(aString) ;` // prints aString but places the cursor after the printed string.
 - `System.out.println() ;` // without parameter prints a newline.

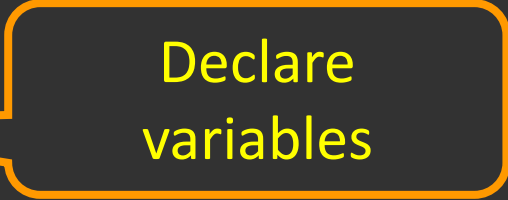
Exercise 1:

- Write a Java program to calculate the area of a circle given its radius.

Exercise 1(cont.):

- Write a Java program to calculate the area of a circle given its radius.

```
2  /*****
3  Declaring and Using Variables
4  Author: Dr. Fadi Alzhouri
5  *****/
6  public class Main
7  {
8      public static void main(String[] args) {
9
10         double radius; // Declare radius
11         double area; // Declare area
12
13         radius = 5; // radius is 5
14
15         area = radius * radius * 3.14159; // Compute area
16
17         // Display results
18         System.out.println("The area of the circle is " + area);
19     }
20 }
```



Declare variables

The area of the circle is 78.53975

Exercise 2:

- Write a Java program to calculate the area of a circle given its radius and PI.

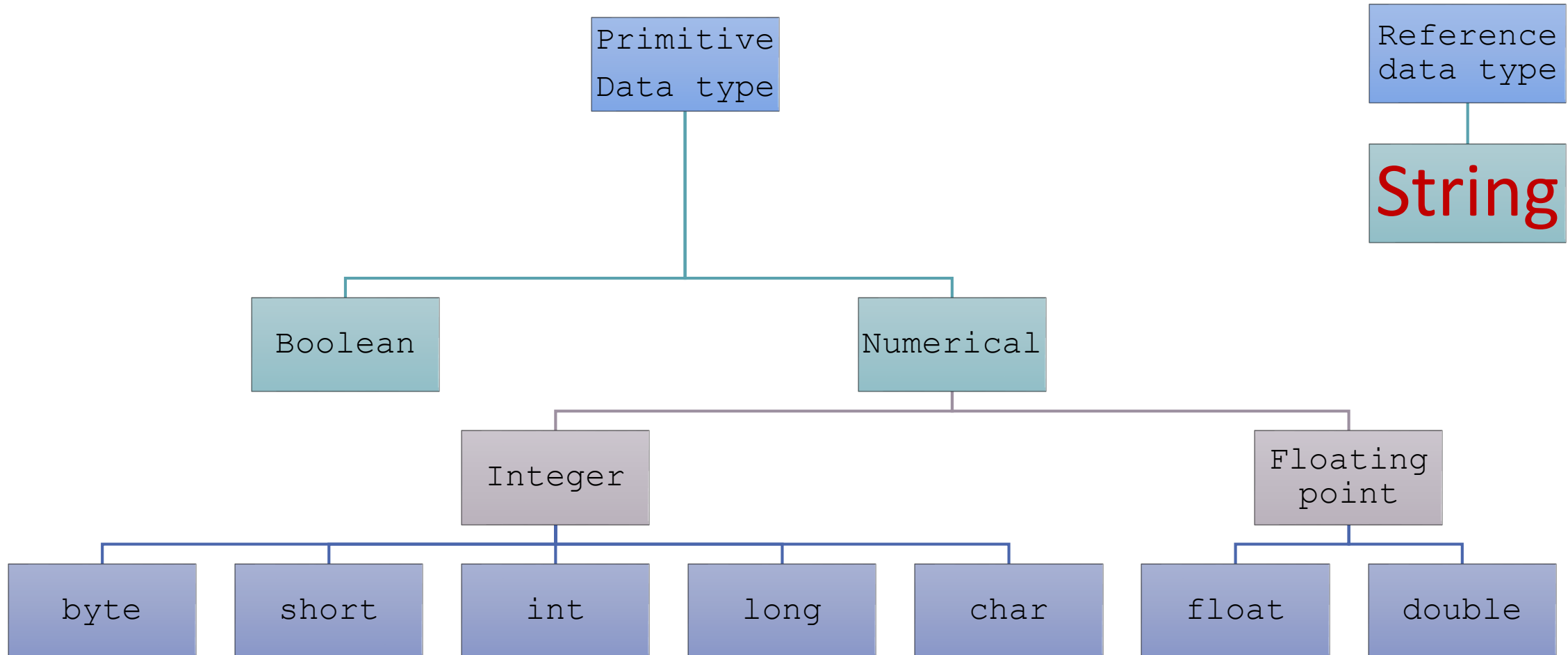
```
1  /*****
2  Declaring and Using Variables
3  Author: Dr. Fadi Alzhouri
4  Example 2: constant
5  *****/
6  public class Main
7  {
8      public static void main(String[] args) {
9
10         final double PI = 3.14159;
11
12         double radius; // Declare radius
13         double area; // Declare area
14
15         radius = 5; // radius is 5
16
17         area = radius * radius * PI; // Compute area
18
19         // Display results
20         System.out.println("The area of the circle is " + area);
21     }
22 }
```

Declare constant

Data Types:

- The Data type determines:
 - The **size** of the variable.
 - The **range** of its values
 - And the set of **operations** that can be applied.

Data Types in Java:



Numeric Types:

Integer

real

Name	Range	Storage Size	
byte	-2^7 to $2^7 - 1$ (-128 to 127)	8-bit signed	byte type
short	-2^{15} to $2^{15} - 1$ (-32768 to 32767)	16-bit signed	short type
int	-2^{31} to $2^{31} - 1$ (-2147483648 to 2147483647)	32-bit signed	int type
long	-2^{63} to $2^{63} - 1$ (i.e., -9223372036854775808 to 9223372036854775807)	64-bit signed	long type
float	Negative range: $-3.4028235E + 38$ to $-1.4E - 45$ Positive range: $1.4E - 45$ to $3.4028235E + 38$	32-bit IEEE 754	float type
double	Negative range: $-1.7976931348623157E + 308$ to $-4.9E - 324$ Positive range: $4.9E - 324$ to $1.7976931348623157E + 308$	64-bit IEEE 754	double type
char	16 bits		
boolean	true/false		

1 byte

1 byte

8 bytes

Int Data Type:

- Write a program to calculate the area of a rectangle. Assume its width is 10 and height is 5.

```
3  /*****  
4  Declaring and Using Variables  
5  Author: Dr. Fadi Alzhouri  
6  Example 3: int data type  
7  *****/  
8  public class Rectangle  
9  {  
10     public static void main(String[] args) {  
11  
12         int width = 10;  
13         int height = 5;  
14         int area;  
15  
16         area = width * height;  
17         // Display results  
18         System.out.println("The area of the rectangle is " + area);  
19     }  
20 }
```

The area of the rectangle is 50

Input From Keyboard: "Scanner"

- You can read input from keyboard via a class called **Scanner** in package **java.util**
- To import this object, use the following statement:

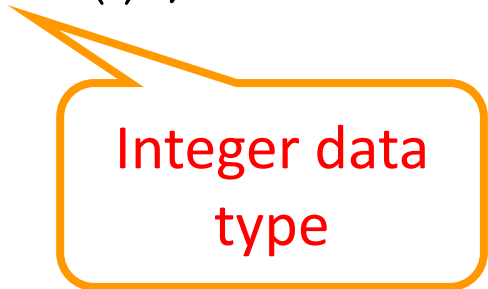
```
import java.util.Scanner;
```

- To create a scanner object, use the following:

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

- To prompt the user to enter an integer number.

```
int number = input.nextInt();
```



Integer data
type

Input From Keyboard: "Scanner" (cont.)

- You can also use the methods listed in Table 2.2 to read a number of the other data types.

TABLE 2.2 Methods for **Scanner** Objects

<i>Method</i>	<i>Description</i>
<code>nextByte()</code>	reads an integer of the byte type.
<code>nextShort()</code>	reads an integer of the short type.
<code>nextInt()</code>	reads an integer of the int type.
<code>nextLong()</code>	reads an integer of the long type.
<code>nextFloat()</code>	reads a number of the float type.
<code>nextDouble()</code>	reads a number of the double type.

If you enter a value with an incorrect range or format, a **runtime error** would occur.

Int Data Type Again:

- Write a program to calculate the area of a rectangle. Prompt the user to enter the width and the height.

Int Data Type Again (cont.):

```
2  2 /*****  
3  3 Declaring and Using Variables  
4  4 Author: Dr. Fadi Alzhouri  
5  5 Example 4: read integer value from keyboard  
6  6 *****/  
7  7 import java.util.Scanner;  
8  8 public class Rectangle2  
9  9 {  
10 10     public static void main(String[] args) {  
11 11         // Create a Scanner object  
12 12         Scanner input = new Scanner(System.in);  
13 13  
14 14         // Prompt the user to enter a width  
15 15         System.out.print("Enter a width: ");  
16 16         int width = input.nextInt();  
17 17         // Prompt the user to enter a height  
18 18         System.out.print("Enter a height: ");  
19 19         int height = input.nextInt();  
20 20         int area;  
21 21  
22 22         area = width * height;  
23 23         System.out.println("The area of the rectangle is " + area);  
24 24     }  
25 25 }
```

Part of java API

Create scanner object

call nextInt method

```
Enter a width: 12  
Enter a height: 4  
The area of the rectangle is 48
```

Float and Double Data Types: Real Values

- **Float** and **Double** are used to represent **real** numbers (decimal values).
- They allow you to work with **fractions**, scientific measurements.
- Float is a **single precision** (32 bit = 4 bytes)
- Double is a **double precision** (64 bit = 8 bytes).
- The double type values are more accurate than the float type values
- Recommendation: use double in general. Use float only if you wish to conserve storage and do not need the precision of double.

Float and Double Data Types: Real Values (cont.)

- Write a code that calculates the area of any circle.

Float and Double Data Types: Real Values (cont.)

```
6 *****/
7 import java.util.Scanner;
8 public class Main
9 {
10     public static void main(String[] args) {
11
12         final double PI = 3.14159;
13
14         // Create a Scanner object
15         Scanner value = new Scanner(System.in);
16
17         // Prompt the user to enter a radius
18         System.out.print("Enter a width: ");
19         double radius = value.nextDouble();
20
21         double area = radius * radius * PI; // Compute area
22
23         System.out.println("The area for the circle of " + " radius "
24         + radius + " is " + area);
25     }
26 }
```

Enter a width: 5.8

The area for the circle of radius 5.8 is 105.6830876

Boolean Data Type:

- It only accepts two values: true and false
- It is mostly used for control or decision-making structures.
- It is not applicable to arithmetic operations (such as addition and multiplication).

```
8 *****
9 public class Main
10 {
11     public static void main(String[] args) {
12         boolean active = true;
13         System.out.println("active is " + active);
14     }
15 }
```

active is true

Char Data Type:

- **char**: meant for a single character, such as 'a', '8', and '\$'.
- Data type char represents an individual character value for example a letter, digit, or special symbol that can be typed at the keyboard.
- A char is enclosed by a pair of **single** quotes.
- Note: char '1' is different from int 1.
- A char is 16-bit character or 16-bit unsigned integer.
- char can be treated as its underlying **integer** in the range of [0, 65535] in **arithmetic** operations.
- You can treat a char as an **int**, you can also assign an integer value in the range of **[0, 65535]** to a char variable.

Char Data Type (cont.):

- Example:

```
10 public class Main
11 {
12     public static void main(String[] args) {
13
14         char grade = 'A';
15         System.out.print("Your grade is " + grade);
16     }
17 }
```

Your grade is A

Char Data Type (cont.):

- Example 2:

```
3  /******
4  Declaring and Using Variables
5  Author: Dr. Fadi Alzhouri
6  Example 6: char vs integer
7  *****/
8  public class Main
9  {
10     public static void main(String[] args) {
11
12         char grade = 'A';
13         System.out.println("Your grade is " + grade);
14
15         grade = 100;
16         System.out.println("Your grade is " + grade);
17     }
18 }
```

Your grade is A
Your grade is d



Variable Initialization & Literals:

- Initialization: Assigning a value to a variable at the time of declaration or before its use. This value is called **literal**.
- It can be assigned **directly** to a variable; or used as part of an **expression**.

```
char grade = 'A';           // character literal
int sum = 0;                 // integer literal
int number = -20;            // integer literal
double total = 99.999;       // double literal
total = total + 32.45;        // double literal

boolean isValid = true;
```



String Data Type:

- A **String** is a sequence of characters enclosed by **double** quotes (e.g., "Hello").
- String is not a **primitive** data type.
- Examples:

```
String course = "OOP";  
String grade = "A";  
String greeting = "Hi folks";  
String street = "4";
```



First letter must be
capitalized

String Data Type (cont.):

- Example 2:

```
4  /******  
5  Declaring and Using Variables  
6  Author: Dr. Fadi Alzhouri  
7  Example 7: string type  
8  *****/  
9  public class Main  
10 {  
11     public static void main(String[] args) {  
12  
13         String firstName = "Sara";  
14         String lastName = "Adam";  
15         System.out.println("Your name is " + firstName + " " + lastName);  
16     }  
17 }
```

Your name is Sara Adam

References

- **Introduction to Java Programming, Brief Version, Global Edition, 11th edition**, Published by Pearson (June 21, 2018) © 2018