



# Java Foundations

## 3-1

### What Is a Variable?

**ORACLE**  
Academy



Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

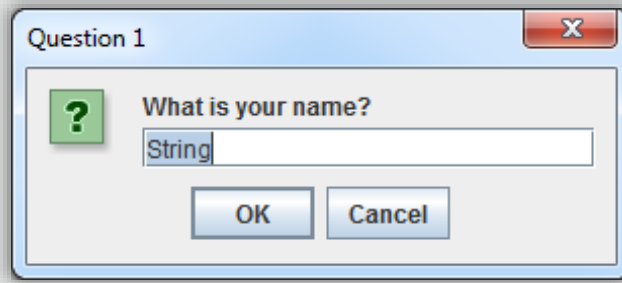
# Objectives

- This lesson covers the following objectives:
  - Understand the benefits of variables
  - Identify four main types of variables:
    - (boolean, int, double, String)
  - Declare and assign values to variables
  - Name variables according to conventions



## Exercise 1

- Run `JavaLibs.jar`
- Consider the types of data this program asks for



**Problem Set 3 is to re-create this program with your own story**  
**This section teaches everything you'll need to create this program**

## What is a Variable?

- Consider the variable  $x$  in an equation
- We can assign any value to  $x$

$$y = -2x + 5$$

$$x = 0$$

$$y = -2 \times 0 + 5$$

$$y = 0 + 5$$

$$y = 5$$

$$x = 2$$

$$y = -2 \times 2 + 5$$

$$y = -4 + 5$$

$$y = 1$$

Plug the value of  $x$  into the equation.

# What Is a Variable in Java?

- Similarly, we can assign values to Java variables

```
String x = "Alex";  
System.out.println("My name is " + x);
```



The diagram illustrates the concatenation operation in the second line of code. A red bracket is drawn under the string "My name is " and the variable x. A red arrow points from the center of this bracket down to the resulting output string "My name is Alex".

"My name is Alex"

Writing the line `String x = "Alex"` is like we're assigning a variable `x` a value of `"Alex"`. Writing `"My name is " + x` is equivalent to writing `"My name is Alex"`.

## Disadvantage Without Variables

- Code isn't flexible
- To replace the name "Alex," you must make many changes in many places:
  - Tedious editing
  - Risk of missing an "Alex"

```
System.out.println("My name is Alex");
System.out.println("Alex is so cool!");
System.out.println("Hooray Alex!");
System.out.println("Please enjoy Alex Appreciation "
    + "Day! My name is Alex. I know how excited "
    + "everyone is to start appreciating Alex on Alex"
    + "Appreciation Day! Alex, Alex, Alex! Yay "
    + "Alex!!! That's me! Alex is the best date ever!");
```

## Advantage with Variables

- Code becomes flexible
  - Remember and manipulate values
- To replace the name “Alex,” you make one change:
  - Efficient editing
  - No risk of missing an “Alex”

```
String x = "Sam";
System.out.println("My name is " + x );
System.out.println(x + " is so cool!");
System.out.println("Hooray " + x + "!");
System.out.println("Please enjoy " + x + " Appreciation "
    + "Day! My name is " + x + ". I know how excited "
    + "everyone is to start appreciating " + x
    + " on " + x + "Appreciation Day! " + x + "," + x + ","
    + x + "! Yay " + x + "!!! That's me! " + x
    + " is the best date ever!");
```

This is the Variables01 project.



## More Advantage with Variables

- Manipulate values many times in several ways:
  - Directly change values yourself (shown below)
  - Programmatically change calculated values
  - Change based on user input

```
5    String x = "Alex";
6    x = "Sam";
7    x = "Nicky";
8    x = "Mystery Date";
9
10   "backwards" = x;    //Can't do this
```



## Exercise 2

- Import and open the `Variables02` project
- Follow the steps in the exercise
- Run the program between each step and observe the output
- Your program should produce the following outputs:

– After Step 1)

```
puppy
puppy
```

– After Step 2)

```
kitty
kitty
```

– After Step 3)

```
kitty
bunny
```

## Line-by-Line Nature of Programs

- From line 8 onward, x always equals "kitty" until ...
- Line 14 onward where x always equal "bunny"

```
7 public static void main(String[] args) {  
8     String x = "kitty";  
9     System.out.println(x);           //prints "kitty"  
10    String x = "bunny";  
11    System.out.println(x);  
12    System.out.println(x);           //prints "kitty"  
13  
14    x = "bunny";  
15    String x = "kitty";  
16    System.out.println(x);  
17    System.out.println(x);           //prints "bunny"  
18    String x = "kitty";  
19  
20 }  
21 }
```

## Many Variable Types

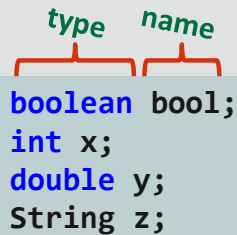
- Variables can exist for many different data types in Java
- Here are the variables that you've seen:

Type	Keyword	Example Values
Boolean	<code>boolean</code>	<code>true</code> , <code>false</code>
Integer	<code>int</code>	1, -10, 20000, 123_456_789
Double	<code>double</code>	1.0, -10.0005, 3.141
String	<code>String</code>	"Alex", "I ate too much dinner."

There are more variable types, but these are the types we'll be using most in this course.

# Declaring a Variable

- Java is a “strongly typed language”
  - You must declare what type of data your variable will handle by using keywords



A diagram illustrating the components of a variable declaration. It shows the words 'type' and 'name' in green, with red brackets underneath them. Below these, four lines of Java code are shown: 'boolean bool;', 'int x;', 'double y;', and 'String z;'. The first two words of each line ('boolean', 'int', 'double', 'String') are in blue, corresponding to the 'type' bracket, and the last word of each line ('bool', 'x', 'y', 'z') is in black, corresponding to the 'name' bracket.

```
boolean bool;  
int x;  
double y;  
String z;
```

- After you declare a variable ...
  - That variable exists
  - There’s no need to declare it again

# Options for Declaring and Assigning Values

- Declare and assign a variable in a single line

*type*   *name*   *value*  
`boolean bool = true;`

- Declare a variable in one line and assign a value later

```
boolean bool;  
// ... later ...  
bool = true;
```

## Assigning Bad Values

- Assigned values must be appropriate for the data type you've declared



```
int x = 3;
```



```
int z = "Puppies!";
```

## Inappropriate Math Values

- We can assign any number value to  $x$
- We can't assign a String value to  $x$ 
  - This doesn't make sense!

$$y = -2x + 5$$

$x = \text{"Puppies!"}$

$y = -2(\text{"Puppies!"}) + 5$

$y = ???$







## Exercise 3, Part 1

- Import and open the `Variables03` project
- There are six mistakes in this program
- Can you fix these mistakes so that the program produces the following output?

```
bool = true
intVar1 = 1
intVar2 = 2
intVar3 = 3
doubleVar1 = 1.1
doubleVar2 = 2.1
doubleVar3 = 3.1
doubleVar4 = 4.1
stringVar1 = 11
stringVar2 = 22
```



## Exercise 3, Hints 1

- NetBeans underlines problematic code
  - Hold the cursor over the code or icon in the left margin for details
  - NetBeans may hint at possible solutions
  - Click the icon in the left margin

```
4 public class Variables03 {  
5  
6     public static void main(String[] args) {  
7         incompatible types: boolean cannot be converted to int  
8         ----  
9         (Alt-Enter shows hints)  
10  
11         int intVar1 = true;  
12         int intVar2 = 2;  
13         intVar3 = 3;  
14  
15         double doubleVar1, doubleVar2, doubleVar3, doubleVar4;  
16         doubleVar1 = 1.1;  
17         doubleVar2 = 2.1;  
18         double doubleVar3 = 3.1;  
19     }  
20 }
```



## Exercise 3, Hints 2

- NetBeans suggested solutions are sometimes bad
  - Don't rely entirely on NetBeans hinted solutions
- Your own problem-solving skills can be a wonderful resource



## Mistakes with Variables

- Assigning inappropriate values for a variable type

```
int intVar1 = true;
```

- Forgetting to declare a variable's type

```
intVar3 = 3;
```

- Misspelling a variable

```
double doubleVar2;  
doublevAr2 = 2.1;    //Java is case-sensitive
```

# Mistakes with Variables

- Declaring the same variable twice

```
double doubleVar3;  
double doubleVar3 = 3.1;
```

- Forgetting to assign a value before using a variable

```
double doubleVar4;  
System.out.println(doubleVar4);
```

*Assigning an initial value to a variable is called initialization.*

## You May Have Noticed ...

- It's possible to declare many variables in a single line

```
double doubleVar1, doubleVar2, doubleVar3;
```

- It's possible to assign values when declaring many variables

```
double doubleVar1, doubleVar2, doubleVar3 = 3.1;
```

- It's a matter of personal preference either to ...
  - Declare every variable on separate lines
  - Declare all variables of a given type in a single line

# Bad Variable Naming



- You can name a variable almost anything you want

```
int dsfdsfspoop = 20;    //Ha ha!
```

- This might be funny, but ...
- Will you or a friend understand what data dsfdsfspoop represents when you read the code?

- Tiny names are usually discouraged

```
int x = 20;
```

- This is useful for testing ...
- And commonly found in small loops (covered later), but ...
- Will you or a friend understand what data x represents when you read the code?

# Very Bad Variable Naming



- Variables can't share the same name

```
int x = 20;  
double x = 22.0;  
System.out.println(x); //Which x?
```

- Variables can't start with numbers

```
boolean 1337Hacker = true;
```

- Keywords can't be used for variables names

```
int continue = 20;
```

- Keywords turn blue in NetBeans
- Keywords have special meanings in Java





## Variable Naming Conventions

- Begin each variable with a lowercase letter
- Subsequent words should be capitalized:
  - myVariable
- Choose names that are mnemonic and that indicate the intent of the variable to the casual observer
- Remember that ...
  - Names are case-sensitive
  - Names can't include white space

```
int studentAge = 20;  
String myCatchPhrase = "Enjoy Alex Appreciation Day!";
```

# Summary

- In this lesson, you should have learned how to:
  - Understand the benefits of variables
  - Identify four main types of variables:
    - (boolean, int, double, String)
  - Declare and assign values to variables
  - Name variables according to conventions



