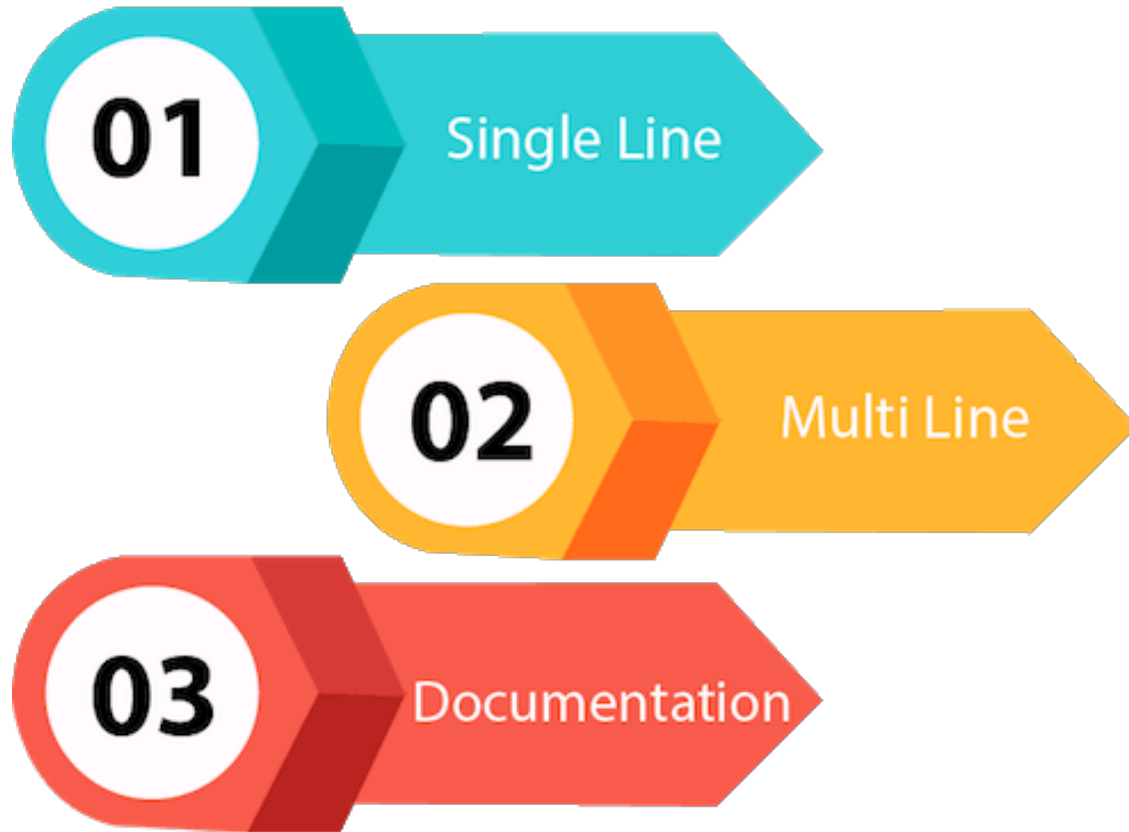




Variables

Comments

Java comments can be used to make the code more readable.



Single-Line Comments

In a single-line comments, anything that follows the two forward slash characters `//` on that line will not be processed by the JavaScript interpreter. Single line comments are often used for short descriptions of what the code is doing.

Multi-Line Comments

To write a comment that stretches over more than one line, you use a multi-line comment, starting with the `/*` characters and ending with the `*/` characters. Multi line comments are often used for descriptions of how the script works, or to prevent a section of the script from running when testing it.

```
//Java program to show single line comments
class Comment
{
    public static void main(String args[])
    {
        // Single line comment here
        System.out.println("Single line comment above");
    }
}
```

Single Line Comment

```
//Java program to show multi line comments
class Scomment
{
    public static void main(String args[])
    {
        System.out.println("Multi line comments below");
        /*Comment line 1
        Comment line 2
        Comment line 3*/
    }
}
```

Multi Line Comment

println() vs print()

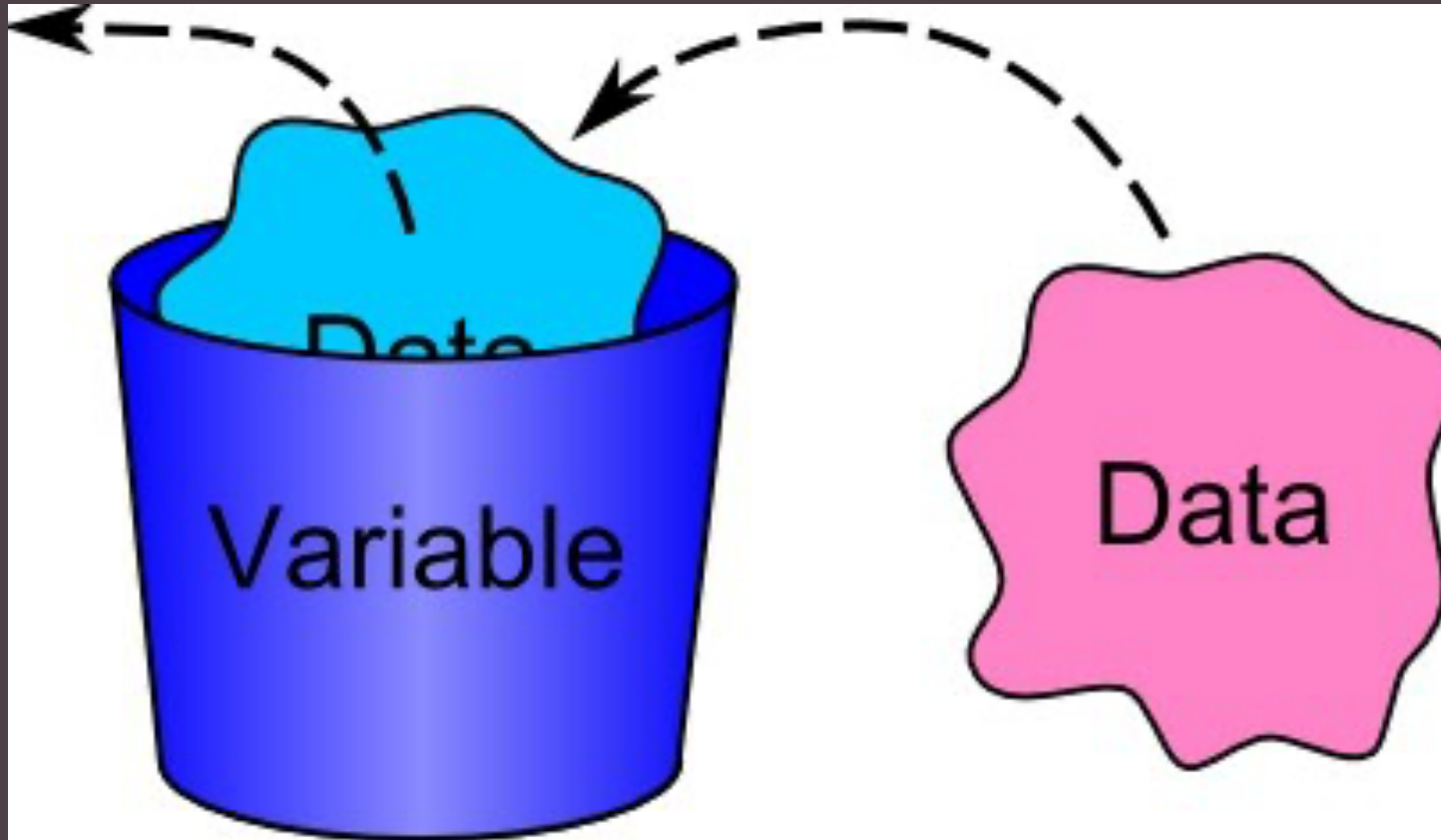
- **System.out.println()** : Appends a newline at the end of the data output
- **System.out.print()** : Does not append newline at the end of the data output

Example	Output
<pre>System.out.print("one"); System.out.print("two"); System.out.println("three");</pre>	onetwothree
<pre>System.out.println("one"); System.out.println("two"); System.out.println("three");</pre>	one two three
<pre>System.out.println();</pre>	[new line]

Common Escape Sequences

Escape Sequence	Name	Description
\n	Newline	Advances the cursor to the next line for subsequent printing
\t	Horizontal Tab	Causes the cursor to skip over to the next tab stop
\\	Backslash	Causes a backslash to be printed
\'	Single quote	Causes a single quotation mark to be printed
\"	Double quote	Causes a double quotation mark to be printed

What is variable?



Calculate the area of rectangle

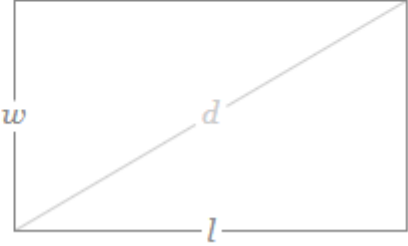
Rectangle

Solve for area ▾

$$A = w l$$

l Length

w Width



The diagram shows a rectangle with a diagonal line from the bottom-left corner to the top-right corner. The vertical side is labeled w , the horizontal side is labeled l , and the diagonal is labeled d .

Steps:

- 1-Remember the value for length
- 2-Remember the value for width
- 3-Multiply length by width to get the area
- 4-Return the result to the user

How to DECLARE variables?

```
DataType VariableName;
```

Primitive Data Types

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 $\pm 4.94065645841246544 \times 10^{-324}$
char	Single character (Unicode)	2 bytes	All Unicode values from 0 to 65,535
boolean		1 bit	True or false

Example of Variable Declarations

```
byte inches;
```

```
int speed;
```

```
short month;
```

```
float salesCommissions;
```

```
double distance;
```

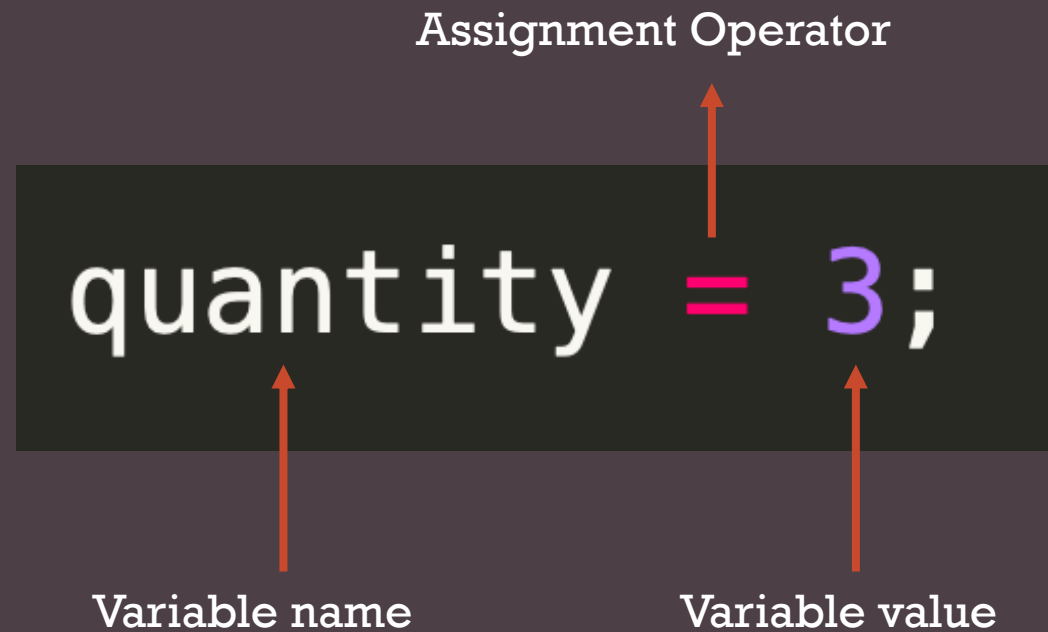
How to ASSIGN variable a value?

Assignment Operator

```
quantity = 3;
```

Variable name

Variable value

A diagram illustrating the components of an assignment statement. The code 'quantity = 3;' is displayed on a dark background. Three orange arrows point from labels to specific parts of the code: one from 'Variable name' to 'quantity', one from 'Assignment Operator' to '=', and one from 'Variable value' to '3'.

The Integer Data Types

- byte, short, int, and long are all integer data types.
- An integer variable can hold **whole numbers** such as 7, 125, -14, and 6928.

Floating-Point Data Types

- float and double are float data types.
- Whole numbers are not adequate for many jobs. If you are writing a program works with dollar amounts or precise measurements, you need a data type that allows **fractional** values.
- Values such as 1.7 and -45.316 are floating-point numbers.

Floating-Point Data Types

- When you write a floating-point literal in your program code, Java assumes it to be of the **double** data type.
- You can force a double literal to be treated as a float, however, by suffixing it with the letter F or f.

The boolean Data Type

- The boolean data type allows you to create variables that may hold one of two possible values: `true` or `false`.

The char Data Type

- The **char** data type is used to store characters.
- A variable of the char data type can hold one character at a time.
- Character literals are enclosed in **single quotation marks**.

String

- String is a sequence of characters, surrounded by double quotes("")

```
String greeting;  
greeting="Hello World";
```

Creating Variables

1

```
int price=5;  
int quantity=14;  
int total=price*quantity;
```

2

```
int price,quantity,total;  
price=5;  
quantity=14;  
total=price*quantity;
```

3

```
int price=5,quantity=14;  
int total=price*quantity;
```

Declaring multiple variables

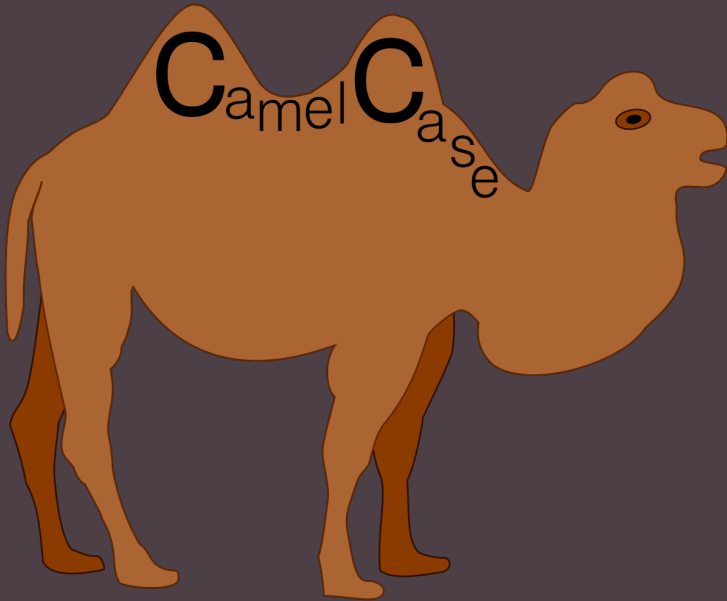
```
String s1,s2;  
String s3 = "yes", s4 = "no";
```

```
int i1,i2,i3=0;
```

```
int num,String value; //Does not compile
```

Rules for Identifiers(Variable or Class Names)

- Variable names should be readable.
- When variable names contain two or more words, use camel case.



```
int itemsOrdered;
```

```
String firstName;
```

```
String lastName;
```

```
long cardNumber;
```

Rules for Identifiers(Variable or Class Names)

- The first character must be one of the letters **a-z** or **A-Z**, an underscore(**_**), or a dollar sign(**\$**)

```
String myVar;  
String myVar1;  
String $myvar;  
String _myvar;
```

Valid Example

```
String 1myVar;  
String *myVar1;
```

Invalid Example

Rules for Identifiers(Variable or Class Names)

- After the first character, you may use the letters **a-z** or **A-Z**, the digits **0-9**, underscores(**_**), or dollar sign(**\$**)
- Identifiers can not include spaces

```
String my20thBirthday$;  
int ageOfThe_Employer100;
```

Valid Example

```
int ageOfThe@Employer100;  
String my20th Birthday$;
```

Invalid Example

Rules for Identifiers(Variable or Class Names)

➤ Variable names can not be Java reserved word.

abstract	do	if	private	this
assert	double	implements	protected	throw
boolean	else	import	public	throws
break	enum	instanceof	return	transient
byte	extends	int	short	true
case	false	interface	static	try
catch	final	long	strictfp	void
char	finally	native	super	volatile
class	float	new	switch	while
const	for	null	synchronized	continue
default	goto	package		

Conversion between Primitive Data Types

Before a value can be stored in a variable, the value's data type must be compatible with the variable's data type. Java performs some conversions between data types automatically, but does not automatically perform any conversion that can result in the loss of data.

```
int x;  
double y = 2.5;  
x=y;
```

```
int x;  
short y = 2;  
x=y;
```

Ranks

