

Professional Skills
Agile Fundamentals
Jira
Git
Databases Introduction
Java Beginner
<div><div></div>What is Java?</div>
<div><div></div>Installation</div>
<div><div></div>Hello World Example</div>
<div><div></div>Data Types</div>
<div><div></div>Packages</div>
<div><div></div>Naming Conventions Cheat Sheet</div>
<div><div></div>Flow of Control</div>
<div><div></div>Class Members</div>
<div><div></div>Operators</div>
<div><div></div>Conditionals</div>
<div><div></div>Iteration</div>
<div><div></div>Arrays</div>
<div><div></div>ArrayList</div>
<div><div></div>Enhanced For Loops</div>
<div><div></div>String Manipulation</div>
<div><div></div>Class Constructors</div>
<div><div></div>Access Modifiers</div>
<div><div></div>Installing Java & Maven To PATH</div>
<div><div></div>Object-Oriented Programming Principles</div>
<div><div></div>Encapsulation</div>
<div><div></div>Inheritance</div>
<div><div></div>Polymorphism</div>
<div><div></div>Abstraction</div>
<div><div></div>Interfaces</div>
<div><div></div>Type Casting</div>
<div><div></div>Static</div>
<div><div></div>Final</div>
<div><div></div>Garbage Collection</div>
<div><div></div>Input With Scanner</div>
<div><div></div>Pass by Value/Reference</div>
<div><div></div>JUnit</div>

Data Types

Contents

- [Overview](#)
- [Primative Data](#)
- [Return Types](#)
- [Tutorial](#)
- [Exercises](#)

Overview

When we declare a variable in Java, we have to specify the type of the value that will be stored within that variable. We do this by giving the variable a type, a reference name, and eventually a value.

```
int myNum
float myFloatNum
char myLetter
boolean myBool
String myText
```

Where the coloured text is the data type and black text is the variable name.

Each variable could be declared to have one of eight **primitive** data types. The eight **primitive** types that we could declare a variable as are as follows.

Type	Representation	Range	Default Value
boolean	N/A	true or false	false
byte	8 bit	-128 to 127	0
char	Unicode	\u0000 to \uFFFF	\u0000
short	16 bit	-32768 to 32767	0
int	32 bit	-2147483648 to 2147483647	0
long	64 bit	-922337206854775808 to 922337206854775807	0L
float	32 bit	3.4e +/- 38(7 digits)	0.0f
double	64 bit	1.7e +/- 308(15 digits)	0.0d

Everything in Java is based off of one of these eight **primitive** types in one way or another.

Primitive Data

So, what is a **primitive** type?

<div><div></div><div>Test Driven Development</div></div> <div><div></div><div>UML Basics</div></div> <div><div></div><div>JavaDoc</div></div> <div><div></div><div>Peer Programming</div></div> <div><div></div><div>Code Reviews</div></div>
Maven
Testing (Foundation)
Java Intermediate
HTML
CSS
Javascript
Spring Boot
Selenium
Sonarqube
Advanced Testing (Theory)
Cucumber
MongoDB
Express
NodeJS
React
Express-Testing
Networking
Security
Cloud Fundamentals
AWS Foundations
AWS Intermediate
Linux
DevOps
Jenkins Introduction
Jenkins Pipeline
Markdown
IDE Cheatsheet

Simply put, **primitive** types are the smallest forms of data Java can handle. All other types of data, or '**non-primitive**', are collections of **primitive** data used together to describe a more complex object.

Return Types

When executing a method, sometimes we need a value to be produced by that method. This is called a return type. In java, returns are **strongly typed**, this means that we need to be explicit with what can be returned and what cannot. If the return type expects an Integer but is sent a String, the program will fail.

```
public class DataTypes {
    public void noReturn() {
        System.out.println("No value returned");
    }

    public String withReturn() {
        System.out.println("Value returned");
        return "Hello World";
    }
}
```

As you can see in the above example, I have two methods **noReturn()** and **withReturn()**, you'll notice **withReturn()** returns "Hello World", so the return type of the method is **String**. But **noReturn()** has no return statement, but still has a return type of **void**. **void** signifies to Java that you do not expect a returned value from executing this method.

Tutorial

To declare a variable with a specific type simply specify the type at the beginning of the variable declaration, like this:

```
boolean bool;
byte bytes;
char character;
short number;
int anotherNumber;
long aLongNumber;
float decimalNumber;
double anotherDecimalNumber;
```

We can also initialise these variables like so:

```
boolean bool = true;
byte bytes = 8;
char character = 'A';
short number = 16;
int anotherNumber = 32;
long aLongNumber = 64L;
float decimalNumber = 3.2f;
double anotherDecimalNumber = 6.4d;
```

We can also specify what data type a method will return. Every method needs to have a return type, even if that return type is **void**. You can declare return types like so:

```
public int methodName() {  
    return 0;  
}  
public boolean methodName() {  
    return true;  
}  
public char methodName() {  
    return 'P';  
}  
public long methodName() {  
    return 52L;  
}  
public float methodName() {  
    return 0.6f;  
}  
public String methodName() {  
    return "Hi there";  
}  
public void methodName() {  
    // no return  
}
```

Exercises

There are no exercises for this module.