Download

Day 1: Java

Download

Programming Languages

- Assembly
- C
- Clojure
- C++
- C#
- Dart
- Elixir
- Erlang
- Fortran
- F#
- Go
- Groovy
- Haskell
- Java
- JavaScript
- Kotlin
- Lisp
- Lua
- MATLAB
- OCaml
- PHP
- Prolog
- Python
- R
- Ruby
- Rust
- Scala
- Solidity
- Swift
- TypeScript
- V
- Zig

Download

So why Java?

Download

- Class based
- Object Oriented
- Easier syntax to pick up Object Oriented concepts
- Compiled language
- Strongly Typed programming language

Download

Interpreted v Compiled

Download

Java is a **Compiled** programming language. So what does that mean?

Download

Compiled languages take the source code and compile it into machine byte code

Interpreted languages, like Python, read code line by line and execute the code line by line.

Download

Pros and Cons of Compiled

Pros:

- Code is checked when compiled.
- Errors can be found before running code.
- Code generally runs faster as it doesn't have ot execute/check the code line by line each time
- Consistency of execution

Cons:

- Additional step before running code to compile
- May take longer before we can run and test our code due to compilation step

Download

Pros and Cons of Interpreted

Pros:

- Generally more forgiving with syntax
- Generally faster to write code

Cons:

- Slower to run as it has to execute line by line
- May take longer to debug as any errors are found when executing code

Download

Java

Download

Java

A compiled, statically typed, Object Oriented programming language that we will use throughout this course.

Java compiles to bytecode, executing code through the JVM

Statically typed simply means data types are clearly defined for each variable and function

We will explore Object Oriented programming more later in the semester

Download

JVM

Download

Java Virtual Machine (JVM)

Java code is compiled into Bytecode that the Java Virtual Machine can read and execute. This means code will run the same regardless of the operating system, as it is executed through a Virtual Machine.

This allows for standardized code, where we would only have to write code once and it would execute the same everywhere.

Download

Byte Code

Download

Numeric instructions read by the Java Virtual machine that gets read as binary later.

Each instruction is a single byte, hence the term "bytecode"

Download

Uses

Download

Uses

- Web Backend Development
- Database (Neo4j)
- Android App Development
- Desktop Applications
- Enterprise Software
- Games (Minecraft)
- Embedded Systems
- Cloud Applications
- Testing Systems (Selenium)
- Data Structures

Download

Java keywords

Download

- https://www.w3schools.com/java/java_ref_keywords.asp
- 51 functional keywords in Java
- 24 functional keywords in Python

Download

Java Code Structure Intro

Download

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

Download

Breaking it down (class)

```
public class Main {
   public static void main(String[] args)
{
      System.out.println("Hello
World");
   }
}
```

Java is a class-based object-oriented programming language. This means the language requires everything to exist in a class of sorts.

We will see what this means later when we dive into creating our own classes.

We will also say the class is "public" to ensure other "classes" can see it.

Download

Breaking it down (main)

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

When we run a java program, our compiler will look for a "main" method. This highlighted line is the entire main method.

A method is essentially a function, but stored in a class.

Download

Breaking it down (main)

```
public class Main {
   public static void main(String[]
args) {
      System.out.println("Hello
World");
   }
}
```

The main method will always start with a "public" keyword at the beginning to denote that the method is available for other classes to see.

We will look at this "public" keyword more later.

Download

Breaking it down (main)

```
public class Main {
   public static void main(String[] args) {
      System.out.println("Hello World");
   }
}
```

After the public keyword, we have our "static" keyword. We will see the purpose of this more later, but it essentially means the function will stick to the class.

this will make far more sense later when we discuss using static vs. non-static/instance functions.

Download

Breaking it down (main)

```
public class Main {
  public static void main(String[] args) {
    System.out.println("Hello World");
  }
}
```

After the "static" keyword, we have our "void" keyword. This is important because functions in Java require a return type.

Unlike Python, when we don't return a value, we must define our function to have a "void" return type.

Simply put, this means there's no value being returned from the function.

Download

Breaking it down (main)

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

Since it's a function, we have parentheses after with a single parameter.

As a reminder, a parameter is the variable we define when we define the function.

Download

Breaking it down (main)

```
public class Main {
   public static void main(String[] args) {
      System.out.println("Hello World");
   }
}
```

Here, this parameter is called "args". Notice we have "String[]" before it.

In Java, when we declare variable, we must give it a type.

This means our arguments will be a list (or properly called an Array) of Strings

Download

Breaking it down (main)

```
public class Main {
   public static void
main(String[] args) {
      System.out.println("Hello
World");
   }
}
```

This parameter is what we call a "Command Line Argument". We will explore these far later in the semester. For now, just remember you have to include "String[] args" in the parentheses for the main function.

Download

Breaking it down (main)

```
public class Main {
   public static void main(String[]
args) {
      System.out.println("Hello
World");
   }
}
```

The main method will always start with a "public" keyword at the beginning to denote that the method is available for other classes to see.

We will look at this "public" keyword more later.

Download

Breaking it down (print)

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

Now this last line is how we "print" to our console.

In Java, we have to call "System.out.println" as a single function. This is actually three different parts working together.

Download

Breaking it down (print)

```
public class Main {
   public static void main(String[] args) {
      System.out.println("Hello World");
   }
}
```

System is a class that directs Java towards System wide actions.

Out tells Java we want to output something.

PrintIn will print the contents passed into the function out into the console, outputting a new line after.

Download

Breaking it down (print)

```
public class Main {
   public static void main(String[] args) {
      System.out.println("Hello World");
   }
}
```

In this case, we are printing out "Hello World" to the console

There are other alternatives to println, which we will see later with the use of:
`System.out.print()`

Download

Setup

Download

We will go through the setup together, but you can also go through the assignment at the setup assignment at /inclass/setup

Download

Post setup

Download

When you finish the setup, go ahead to the post-setup assignment at inclass/post-setup