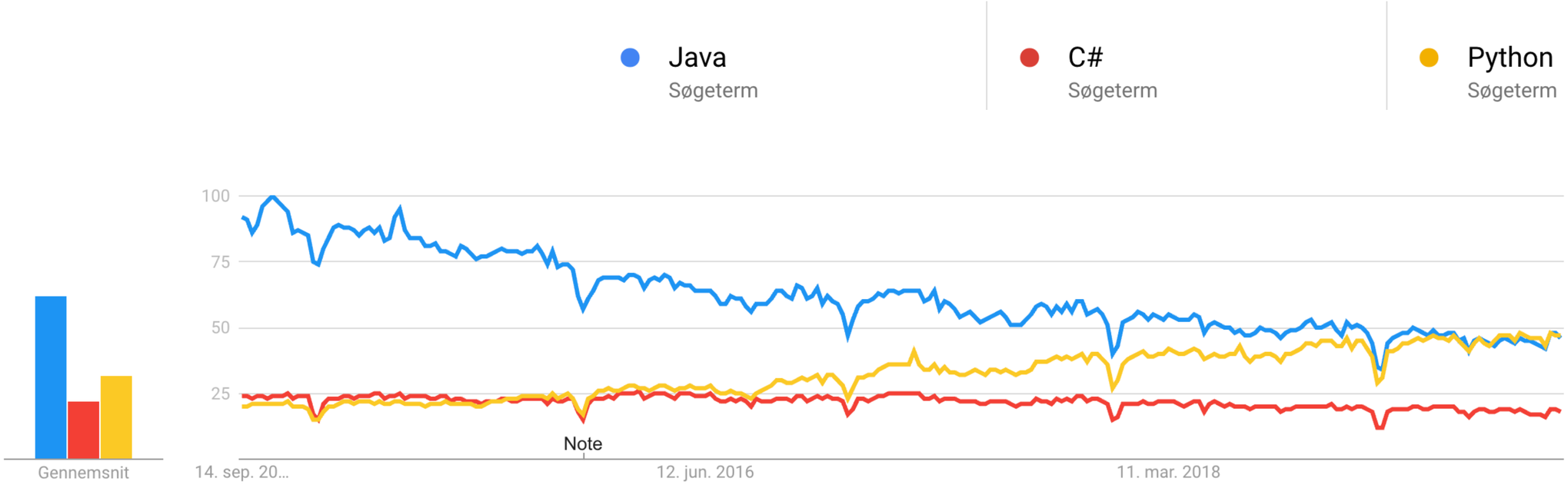
Essential Computing 1

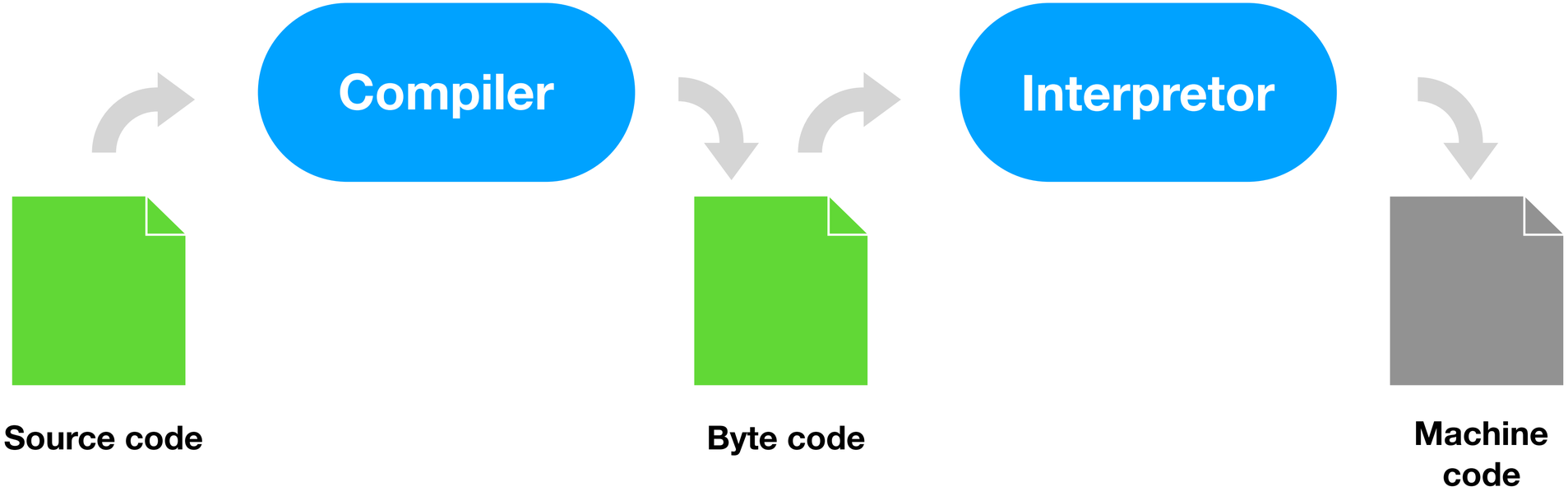
Java introduction

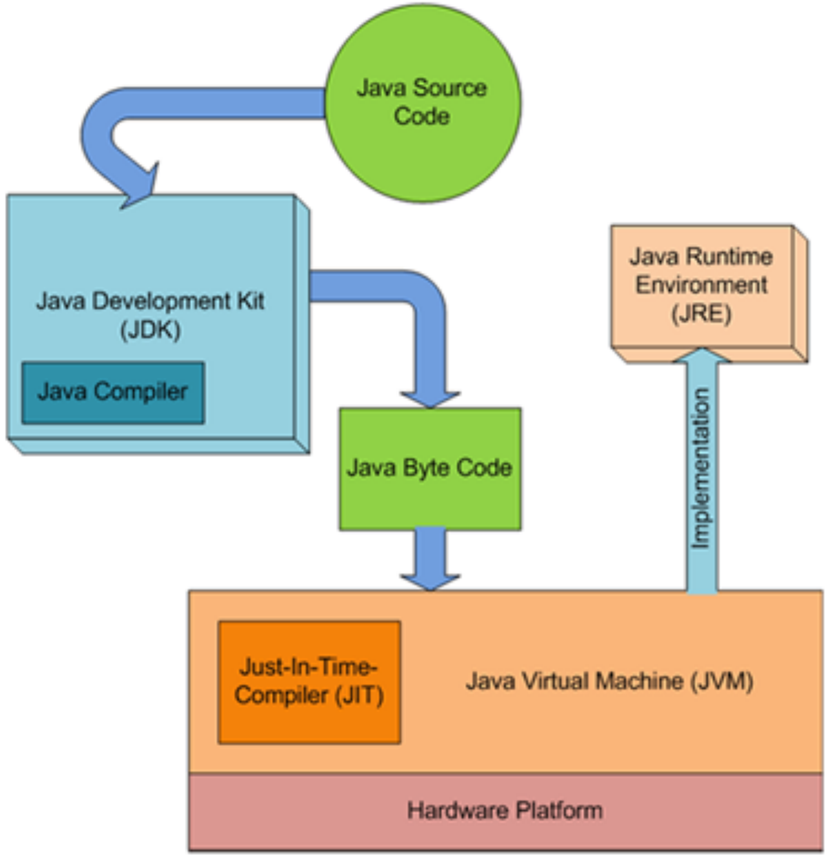
Google Trends



Source: https://trends.google.com

Compiled & Interpreted





Source: https://javapapers.com/core-java/differentiate-jvm-jre-jdk-jit/

Java Development Kit (JDK)

All you need to develop Java

Java Virtual Machine (JVM) implementation for your hardware

Java Runtime Environment (JRE)

For running Java.

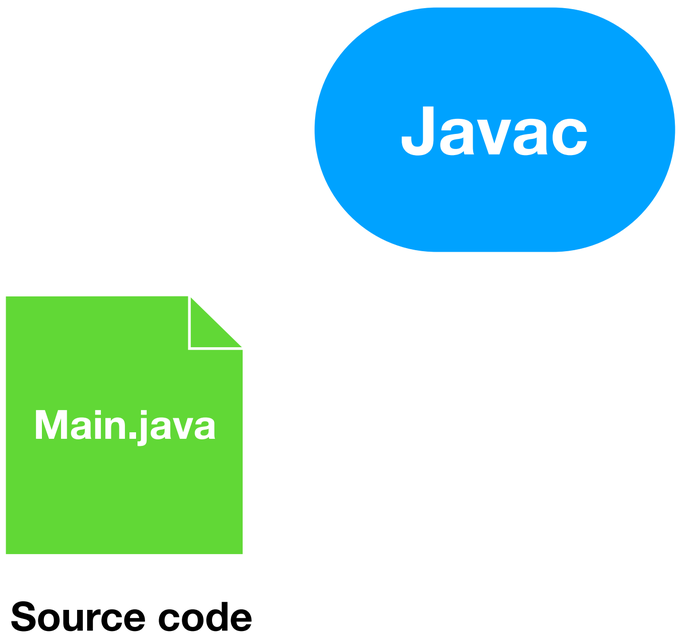
Java Virtual Machine (JVM)

The program that runs your compiled Java code.

Source code

Readable "human" code

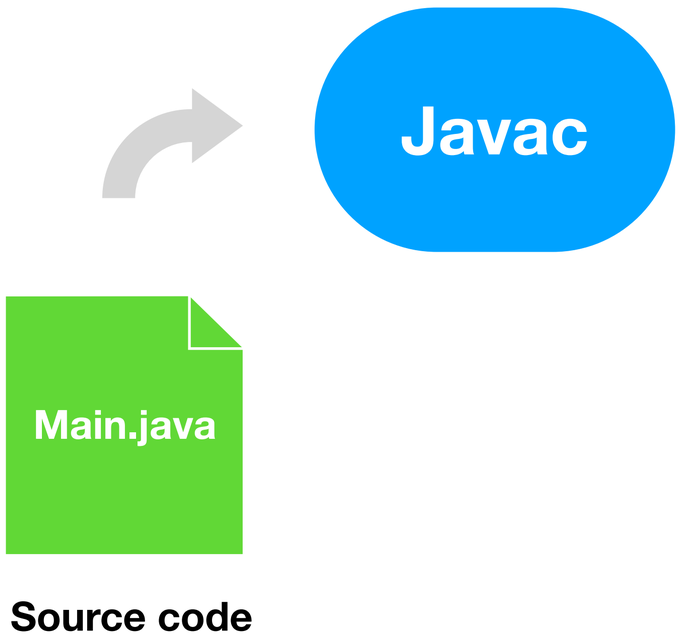
Source code

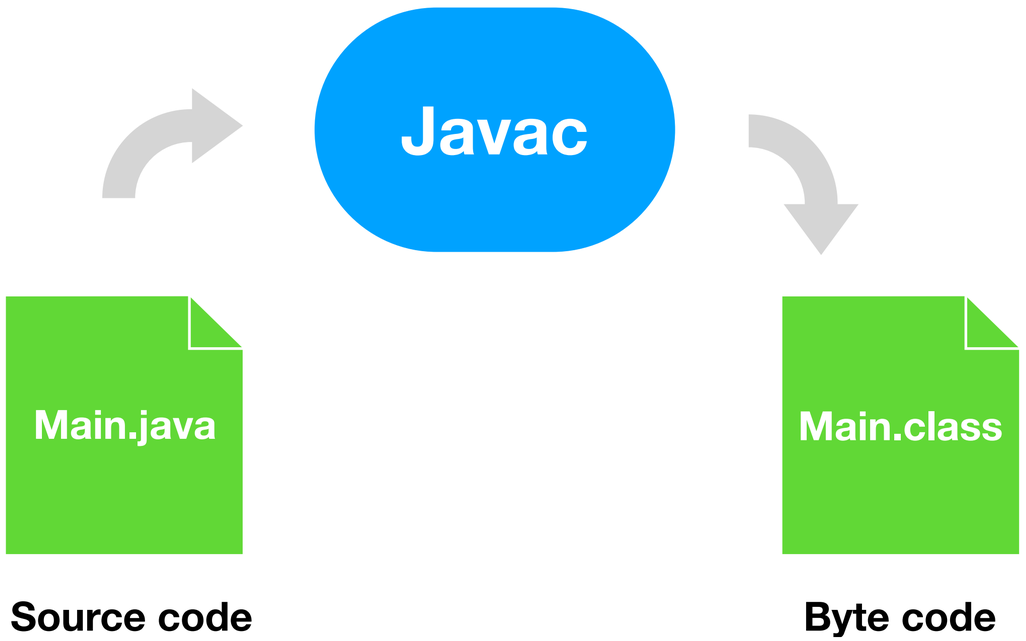


Javac

A compiler, part of JDK.

```
javac Main.java
```



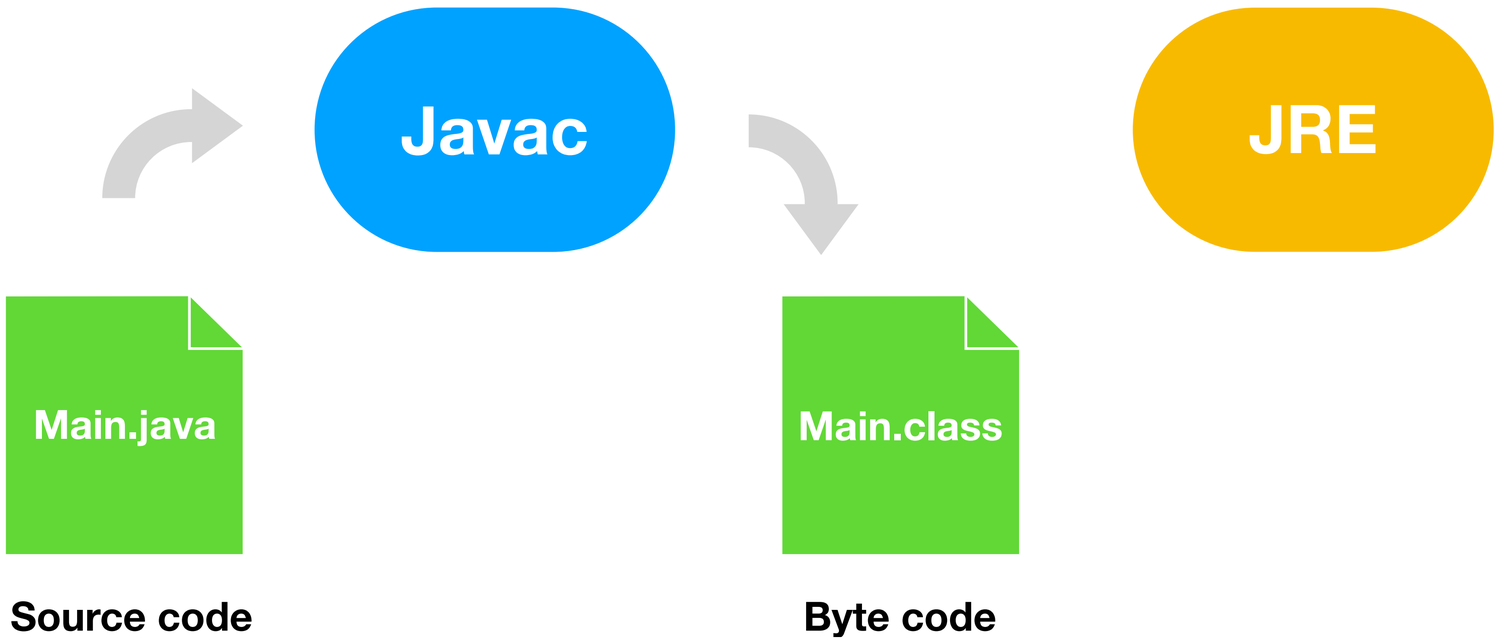


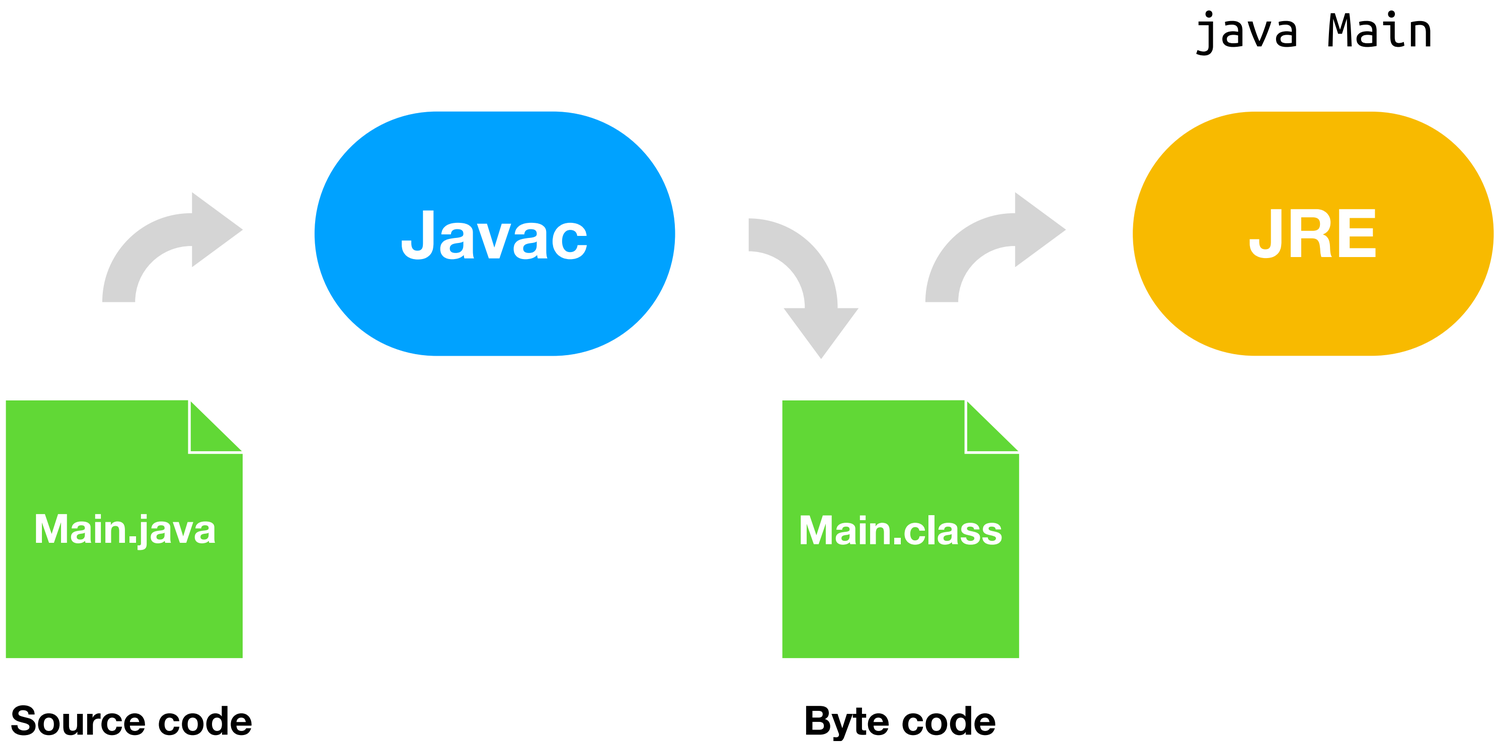
Byte code

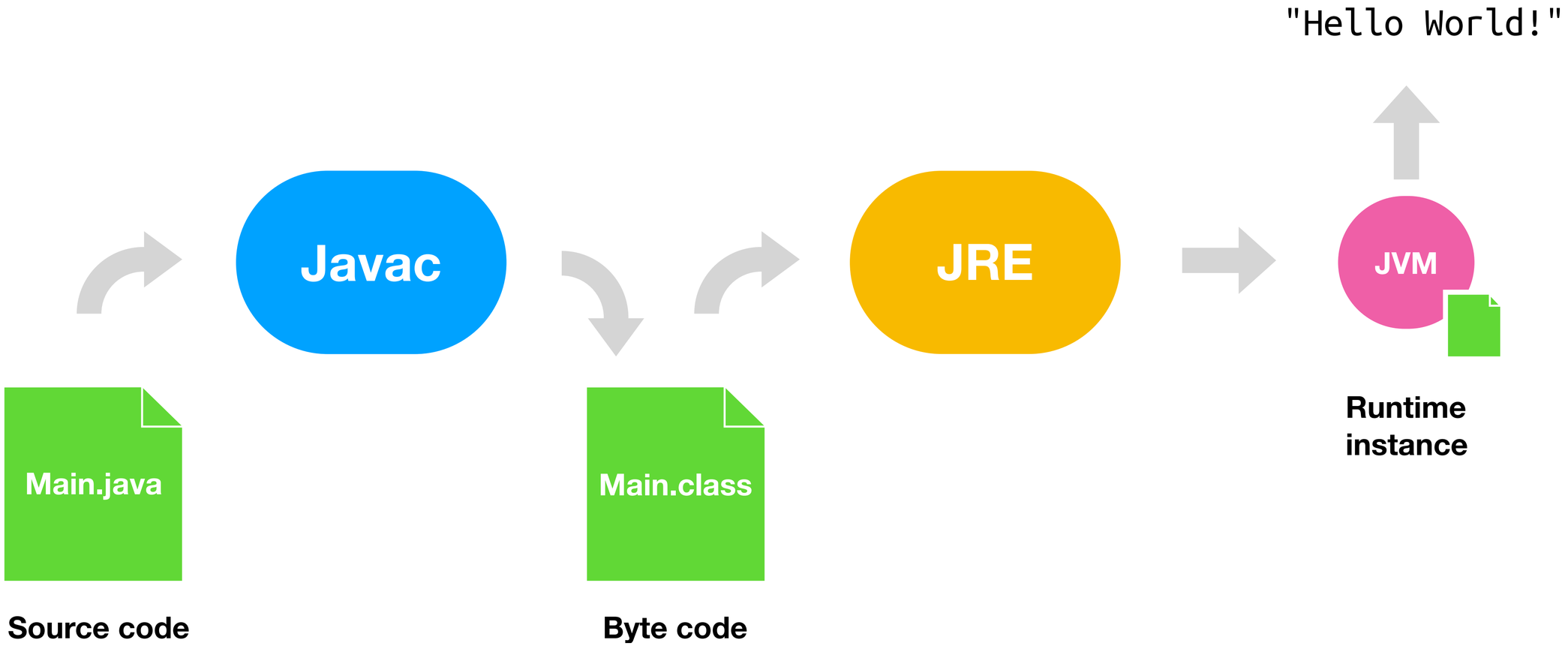
Optimised for interpretation

Java Runtime Environment (JRE)

Package for running Java byte code




java Main
Javac
JRE
Main.java
Main.class
Source code
Byte code


"Hello World!"
Javac
JRE
JVM
Main.java
Main.class
Source code
Byte code
Runtime instance

Let's try just that!

```java
public class Main {

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

}
```

Java looks for a method exactly like this to start the program

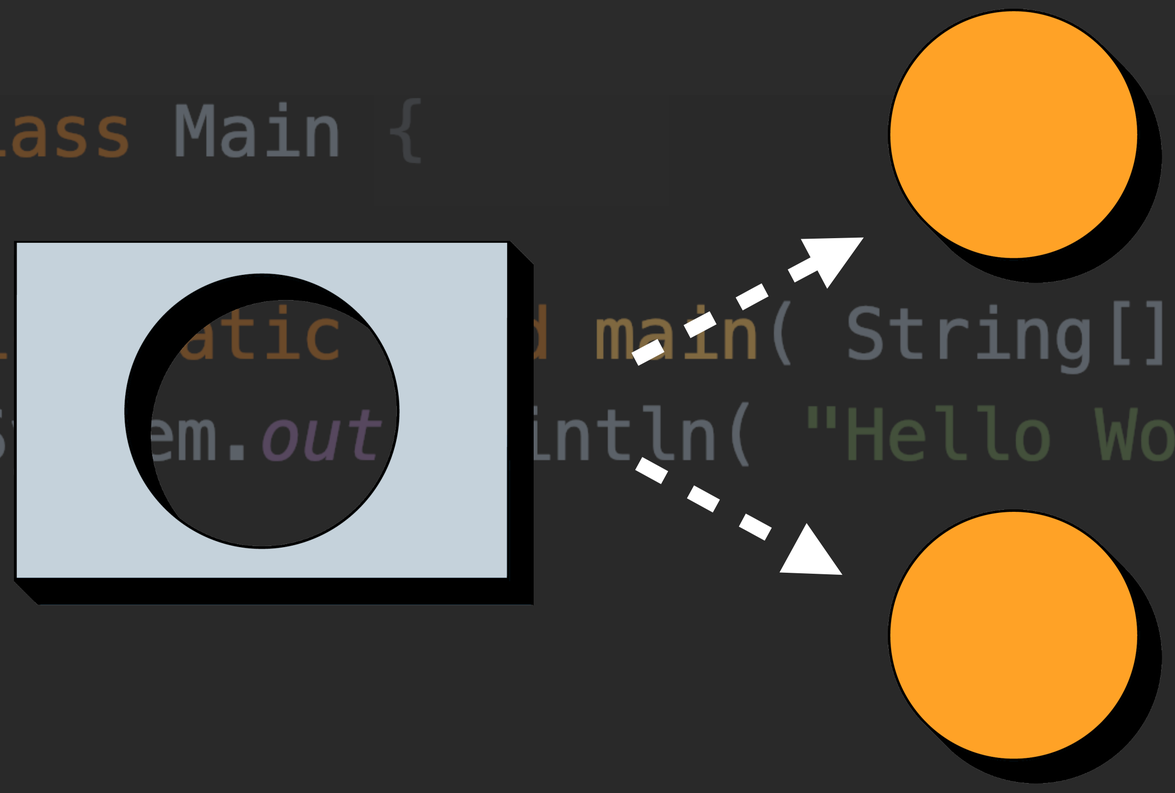
```
public class Main {

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

}
```

Class definition: A class named "Main"

```
public class Main {

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

}
```

A class is a template for creating objects

```
public class Main {

    publi    atic    d main( String[] args ){
        S    em.out    intln( "Hello World" );
    }

}
```



Access modifier: private, protected or public

```
public class Main {

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

}
```

Class name always starts with upper-case letter

```
public class Main {

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

}
```

Class scope: Encapsulates things that belong to the class

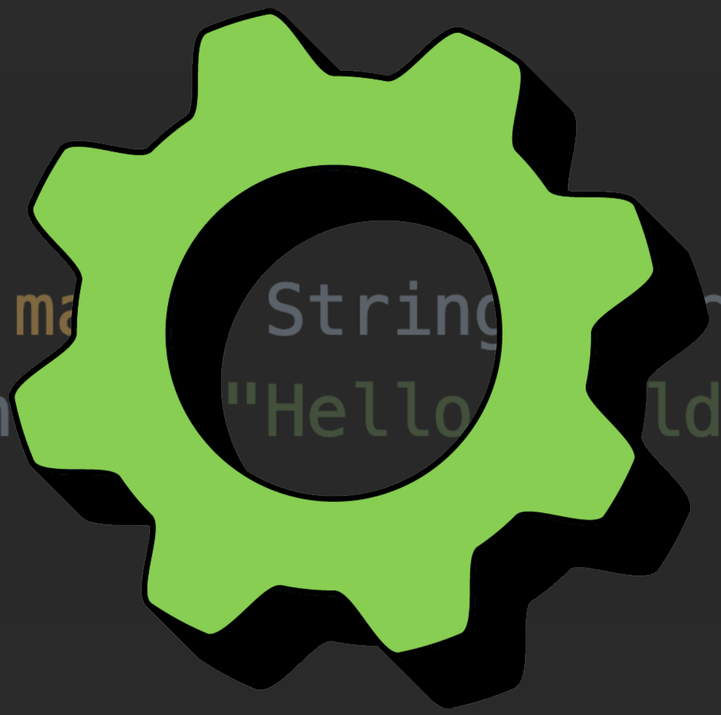
```
public class Main {

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

}
```

Method definition

```
public class Main {

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

}
```

A **function** does a job

```
public class Main {

    public static void ma      String      rgs ){
        System.out.prin      "Hello      ld" );
    }

}
```



Access modifier publicly accessible from outside this class.

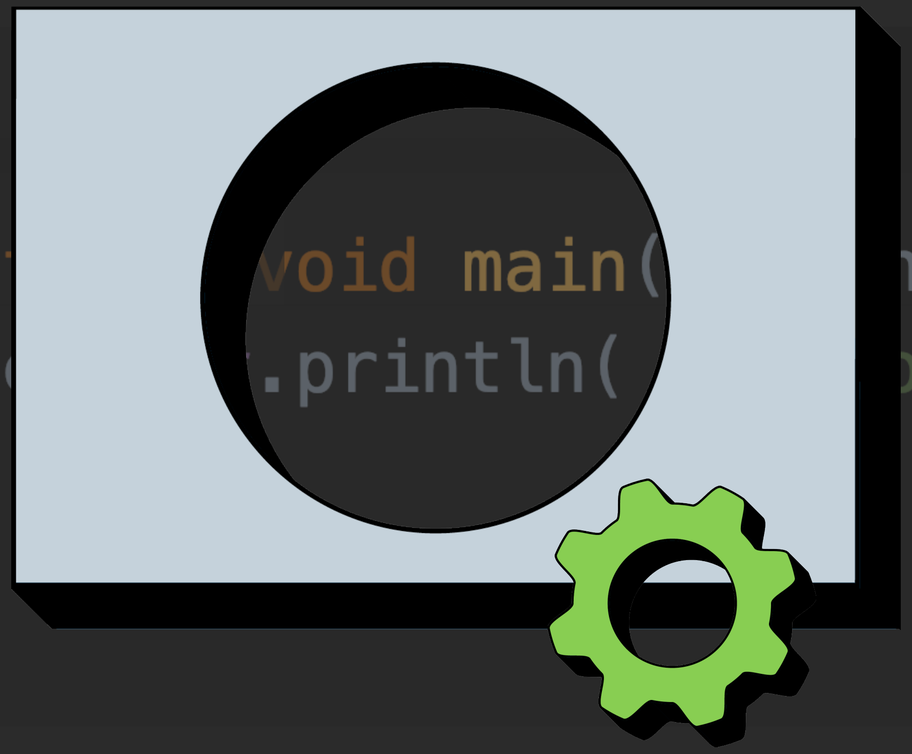
```
public class Main {

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

}
```

Static modifier This method belong to the class, not the object.

```
public class Main {

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

}
```

Static modifier This method belong to the class, not the object.

```
public class

    public st void main( ng[] args ){
        Syst .println( o World" );
    }

}
```



Return type "void" means this method returns nothing

```
public class Main {

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

}
```

Method name always starts with lower-case letter

```
public class Main {

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

}
```

Syntax: all methods have parentheses after name

```
public class Main {

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

}
```

Syntax: they encapsulates function arguments (input variables)

```
public class Main {

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

}
```

This methods has **one argument** (input variable)

```
public class Main {

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

}
```

Datatype string stores text

```
public class Main {

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

}
```

Hard brackets signify an **array** (so, an array of strings)

```
public class Main {

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

}
```

Argument name: This argument variables is called *args*

```
public class Main {

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

}
```

Whenever you see a datatype follow by a name: it's a **variable**

```
public class Main {

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

}
```

Function scope: Encapsulates things that belong to the function

```
public class Main {

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

}
```

Function code: In this case, just a single line.

```
public class Main {

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

}
```

Access the **System** class

```
public class Main {

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

}
```

Inside the **System** class, access ...

```
public class Main {

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

}
```

... the **out** object.

```
public class Main {

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

}
```

Inside the **out** object, access ...

```
public class Main {

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

}
```

... the **println** method. It prints a line of text to the console.

```
public class Main {

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

}
```

A method call

```
public class Main {

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

}
```

Quotes "" signify a **literal string** (text)

```
public class Main {

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

}
```

We send the string "hello world" as **an argument** to println

```
public class Main {

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

}
```

Hello World