

# UCB CS61B

---

## Lecture 1

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

- All code in java must be part of a **class**.
- Delimit the beginning and ends of code segments with { and }
- All statements must end with a semicolon ;
- For code to run, we put it inside of a **public static void main(String[] args)**.

```
public class HelloNumber {  
    public static void main(String[] args) {  
        int x = 0;  
        while (x < 10) {  
            System.out.println(x);  
            x = x + 1;  
        }  
    }  
}
```

- java requires that variables be **declared** before they are used. like **int x = 0;**
- java variables have a specific type, that type can never change, **static type**.
- Types are verified before the code runs, **Static type checking**.

## Functions

```
public class LargerDemo {  
    public static int larger(int x, int y) {  
        if (x > y) {  
            return x;  
        }  
        return y;  
    }  
  
    public static void main(String[] args) {  
        System.out.println(larger(5, 10));  
    }  
}
```

- Functions must be part of a class. A function that is part of a class is called a **method**. Therefore, all functions in Java are methods.
- To define a function, we use (for today) the **public static**.
- All parameters of a function must have a **declared type**, and the function must **return a specific type**.
- functions in java only return one value.

### why make a class file at all?

class file has been type checked. Distributed code is safer.

class file are 'simpler' for machines to execute. Distributed code is faster.

### Static Type

#### *The Goods*

Always what things are - easier to reason about code.

User facing application, less opportunity for users input to cause errors.

Many fewer bugs possible.

Early error detection. You can fix it right when you write it.

#### *The Bads*

Less flexibility.

Makes the language more verbose.

Tricky syntax. The syntax gets very weird for methods that need to deal with very specific types with certain properties.