Outline
class Main {
    public static void main (String[] args) {
        System.out.println("Hello world!");
    }
}
public final class Main {
    public static void main (final String... args) {
        System.out.println("Hello world!");
    }
}
public final class Main {
    private Main(){}
    public static void main (final String... args) {
        System.out.println ("Hello world!");
    }
}
public final class Main {
    public static void main (final String ... args) {
        System.out.println ("Hello world!");
    }
}
Consider each word.
What happens if you leave it out?
What happens if you spell it differently?
What happens if you replace it its “opposite” (if there is such a thing)?
access modifier for things with unrestricted access; one public, top-level Java class per file

modifier for classes that are not to be sub-classed

keyword introducing a Java class

name of class; capitalized by convention; should be same as file name

access modifier for methods with unrestricted access; method main must be public if it is to be the starting point of the program by the Java virtual machine

method modifier indication a non-instance method

return type of void means the method does not return a value; it is a subprocedure not a function
name of method; must be called "main" if it is to be the starting method

type of the one parameter to the method; must be an array of strings (or equivalently varargs), if the method is to be the starting method

name of the one parameter to the method

java.lang.System is the name of the class in package java.lang containing standard I/O objects

Field of java.lang.System with type java.io.PrintWriter containing the object with the reference the program’s standard output stream.

Name of the overloaded method that puts strings on to output stream (prints or displays the text on the screen).
Definitions.

- access modifier

Rules of thumb.

- Declare your (outer) classes public.
- Declare your classes final.
- Declare your formal arguments final.