Annotations in Java

https://docs.oracle.com/javase/tutorial/java/annotations/index.html

Annotations are metadata that can be attached to Java source code, but are not part of the logic of the programs themselves. They are more like compiler directives. Classes, methods, fields and packages can be annotated. Java provides a set of built in annotations (such as @SuppressWarnings, @Deprecated, @Override etc.) and you can write your own annotations.

A built-in annotation:

```
@SuppressWarnings("unchecked")
void myMethod() { ... }
```

User-defined annotation:

```
@interface Bookinfo {
  String author();
  String date();
  int currentRevision() default 1;
  String lastModified() default "N/A";
  String lastModifiedBy() default "N/A";
  String[] reviewers();
}

@Bookinfo (
        author="John Doe",
        date="2/2/2015",
        currentRevision=3,
        lastModified = "2/2/2014",
        lastModifiedBy = "Jane Doe",
        reviewers = {"Alice", "Bob", "Cindy"}
        )

class Book{
//Stuff
}
```

Persistence of annotations

Optionally annotations can be embedded in a class file and retrieved via reflection:

```
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;

@Retention(RetentionPolicy.SOURCE)
//The marked annotation is retained only in the source level and is ignored by the compiler.

@Retention(RetentionPolicy.CLASS)
//The marked annotation is retained by the compiler at compile time, but is ignored by the Java Virtual Machine (JVM).

@Retention(RetentionPolicy.RUNTIME)
//The marked annotation is retained by the JVM so it can be used by the runtime environment.
```