Annotations in Java are a powerful feature that allows you to add metadata to your Java code. They provide additional information about the program without changing the logic of the code. Annotations can be applied to classes, methods, fields, parameters, and other elements. They are widely used in frameworks, libraries, and tools to enable various functionalities, such as dependency injection, data validation, and custom behaviours.

 @SuppressWarnings("unused")

        Test t = new Test(); //The warning goes away.

The @Retention annotation in Java defines how long an annotation is kept. There are three retention policies:

1. **SOURCE**: The annotation is kept only in the source code and discarded during compilation.
2. **CLASS**: The annotation is kept in the compiled class file but not available at runtime. This is the default.
3. **RUNTIME**: The annotation is kept in the class file and is available at runtime through reflection.

**The @Target annotation in Java is used to specify where an annotation can be applied (e.g., methods, fields).**

**@Target(ElementType.TYPE) //@Target({Element.Type, Element.Method}), If it's not present means it can be applied to any thing**

A screen shot of a computer program

Description automatically generated

Task: Create an annotation called “RunImmediately” . Appy this on one method. Create an object of a class. And the methods which are marked with “RunImmediately” annotation.

A screen shot of a computer program

AI-generated content may be incorrect.

