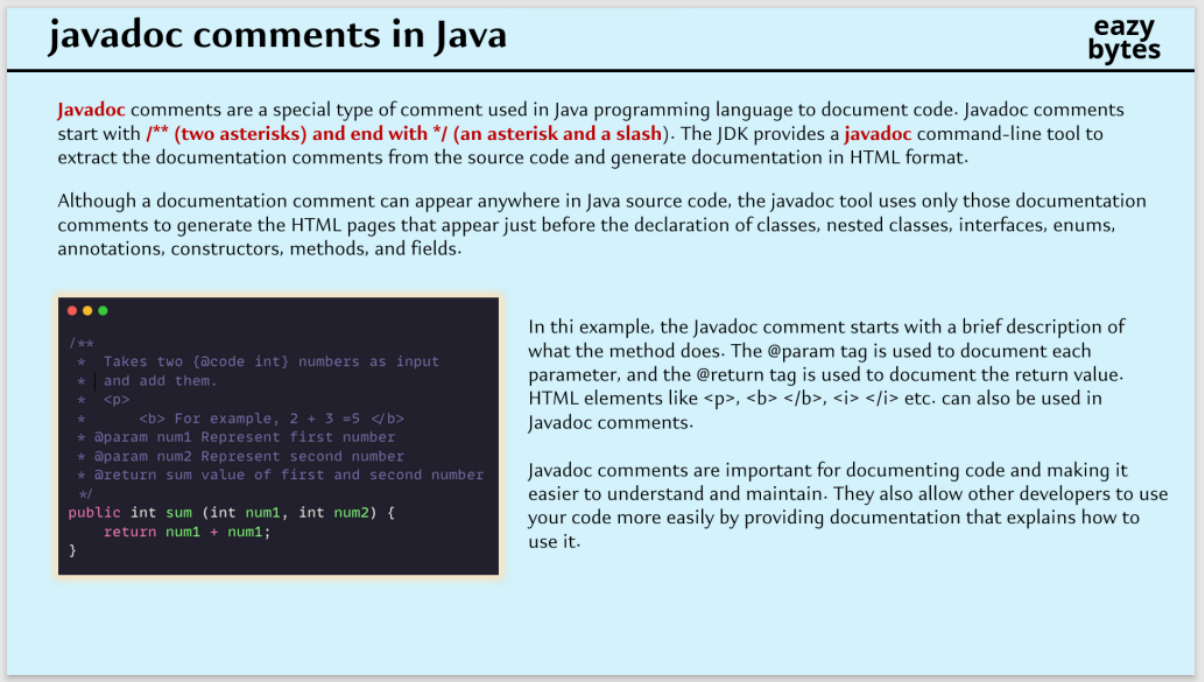
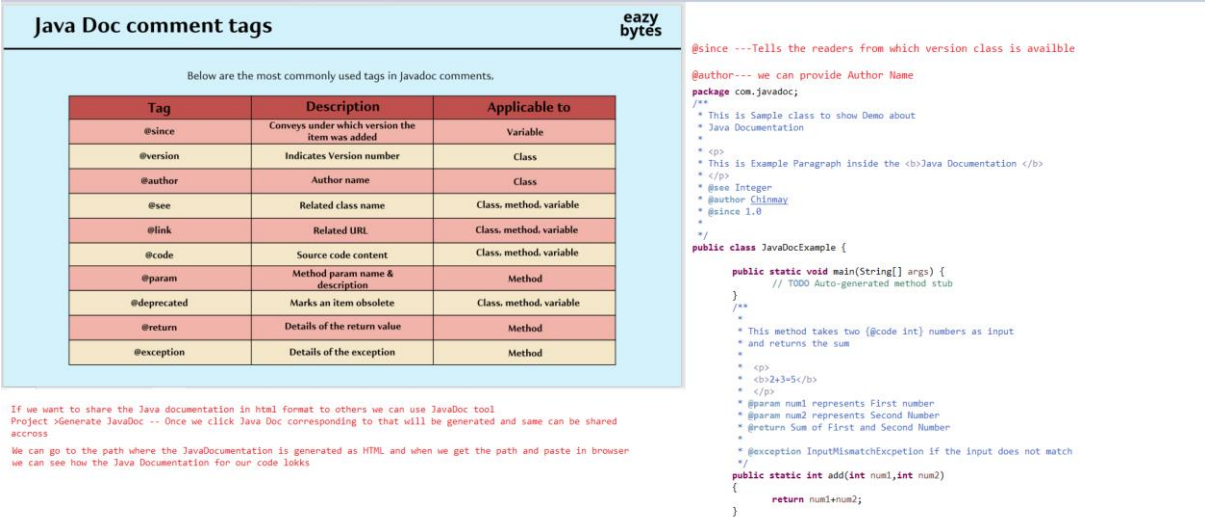




ANNOTATIONS AND JAVA DOC JAVA DOC

➢ There is a tool called java doc that will help to prepare the documentation for the classes.

➢ Java provides some tags for java documentation those are known as java doc tool.

➢ The tags for class or a package are:

• @author: Adds the author name of the class.

• @version: Adds a version subheading with speci>ied version text to generated docs

• @since: To mention when was the version written or how long it may be valid.

• @see: Adds a see also heading with a link for example to see the references to the given link.

➢ The tags for methods are:

• @param: To mention the parameters taken by particular product.

• @return: To mention the value returned by the method.

• @throws/@exception: To know the exception thrown by the method.

• @deprecated: To mention whether the method is deprecated or not that is the method may not be used longer.

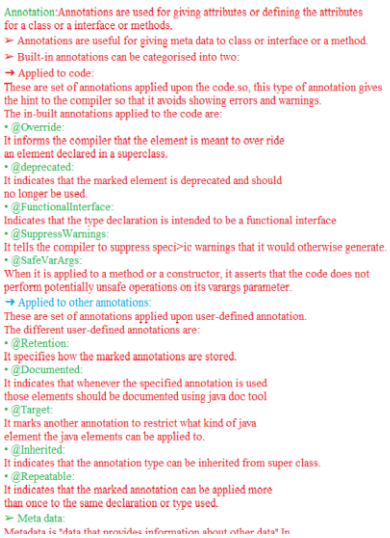
• @code: Displays text in code font without interpreting the text as HTML mark-up or nested java doc tags.

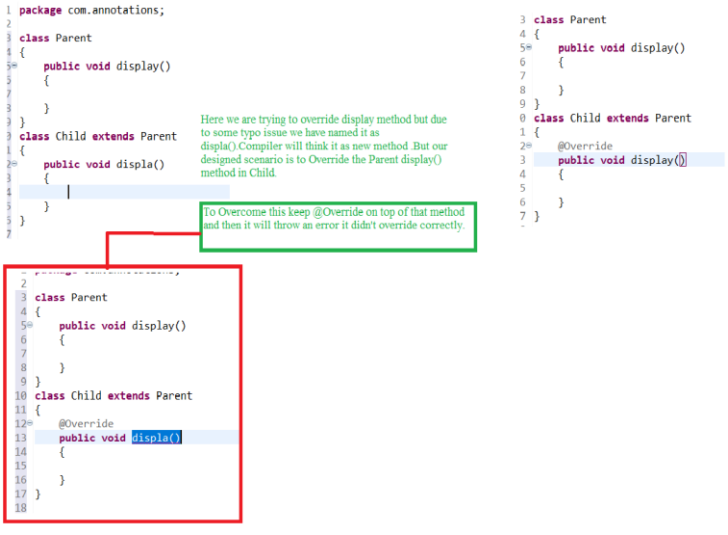
➢ The other tags available are:

• @link: To provide the link for particular resource.

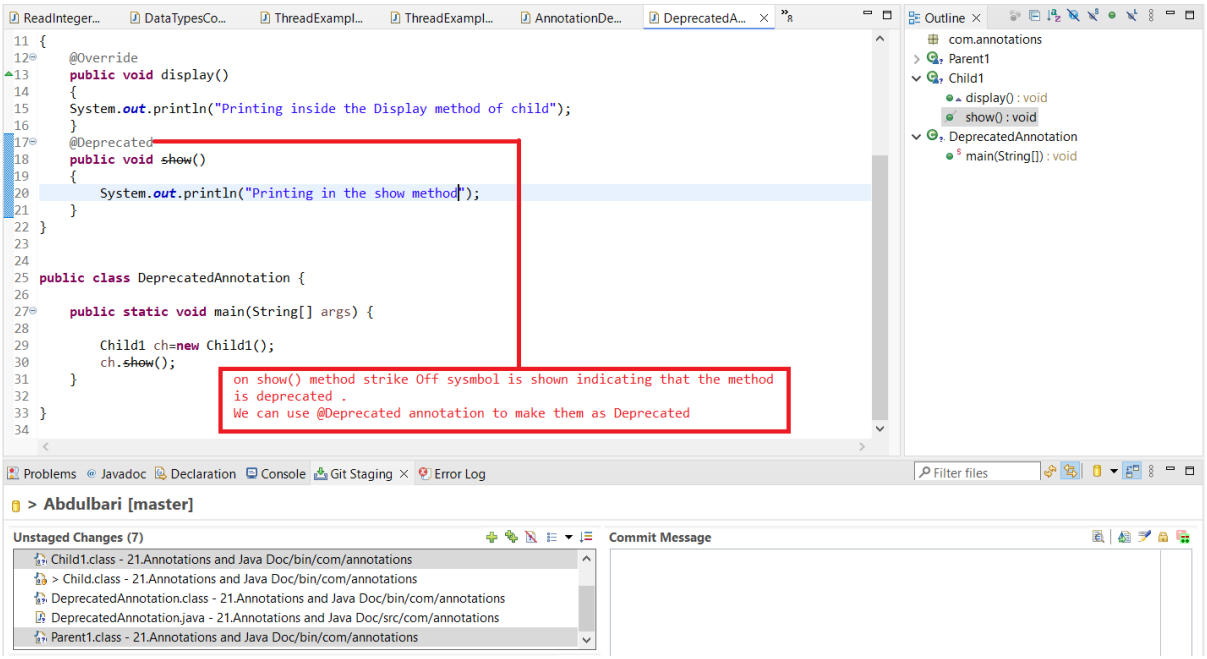
• @value: To provide value for any static variable or a member.

• @serial: To mention the serial id for serialization.

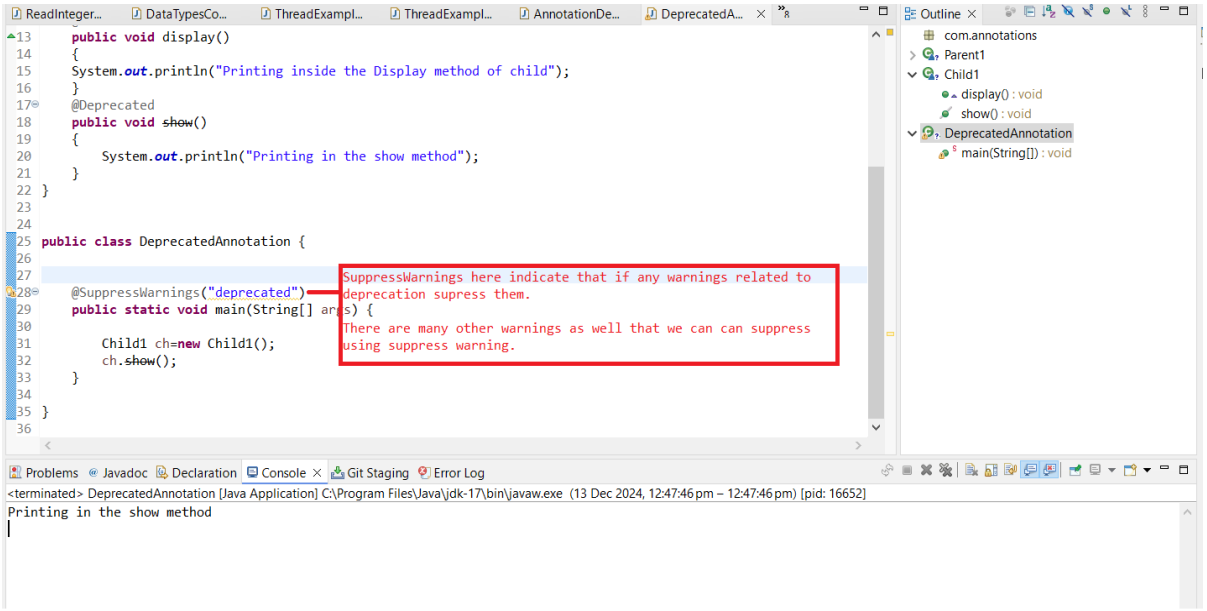




When ever we are overriding it is good practice to use @Override Annotation.



@Deprecated Annotation



@Supress Warnings

```java
package com.annotations;

class My<T>
{
    @SafeVarargs   Will be used for variable arguments and that method should be private or final
    private void show(T...arg)
    {                          We can use @Supresswarnings as well to supress those
        for(T x:arg)           warnings
        {
            System.out.println(x);
        }
    }
}

public class SafeVar {

    public static void main(String[] args) {


    }

}
```

Problems | Javadoc | Declaration | Console | Git Staging | Error Log

@Safevargs

208.User-Defined Annotation

ReadInteger... | AnnotationDe... | DeprecatedA... | SafeVar.java | FunctionalI... | UserDefinedA... | 11

```java
package com.annotations;

@interface MyAnno
{

}
@MyAnno
public class UserDefinedAnnotation {

    @MyAnno
    int val;
    @MyAnno
    public static void main(@MyAnno String[] args) {

        @MyAnno
        int a=10;
    }
}
```

whenever you define annotation that will be automatically inheriting from annotation class that is inside annotation package which is inside java.lang package.

we can give annotation at class or method level or parameter level or a local variable level Or an instance variable.

Annotation definition will be similar to interface but it will be starting with @.

```
//User Defined Annotation
@interfcae MyAnno
{

}
```

User-Defined Annotatons.txt

209: Built in Annotations: