IT Bootcamp

Testing

Code Coverage

DigitalHouse>

1 | Code Coverage

Code Coverage (Cobertura de código)

La cobertura de código es una métrica utilizada para determinar el porcentaje de código cubierto por los tests automatizados.

Es decir, chequea que partes del código son ejecutadas durante los tests y cuáles NO.

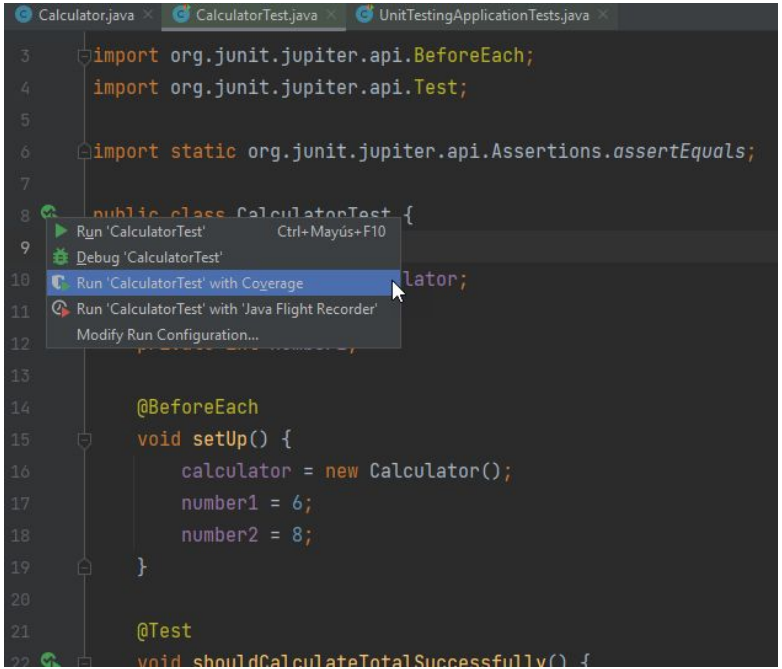
La importancia de una buena cobertura

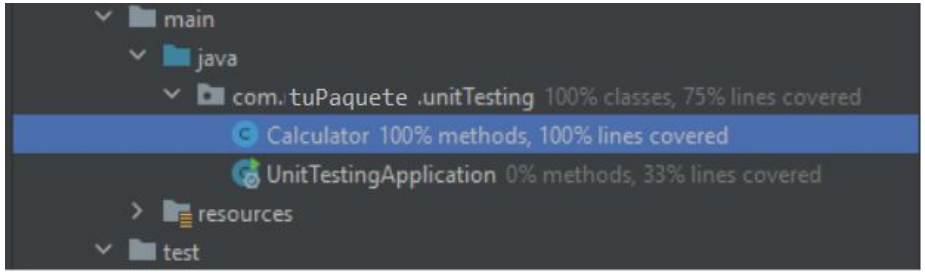
Tomar un enfoque de testeo basado en obtener una buena cobertura desde el principio del proyecto elimina posibles BUGS en un estadio inicial del ciclo de desarrollo. Es importante remarcar que lograr un 100% de cobertura no necesariamente refleja que el testing sea efectivo, ya que sólo indica la cantidad de código que ha sido cubierto por los test.

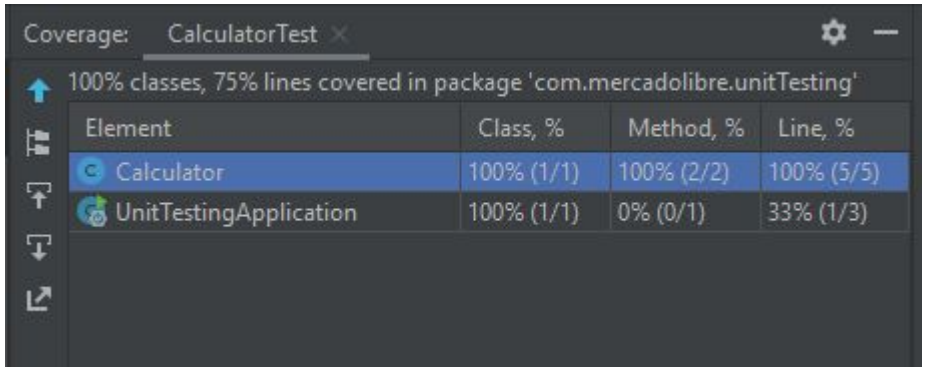
Generalmente, una buena cobertura de código debe superar el 80%

Cove Coverage

Se puede verificar la cobertura de código desde IntelliJ mediante el coverage runner que provee este IDE.







JaCoCo

Es un generador de reportes de cobertura de código para proyectos Java. Para generar reportes con JaCoCo, necesitamos declarar el siguiente plugin en el archivo pom.xml

```
<plugin>
  <groupId>org.jacoco</groupId>
  <artifactId>jacoco-maven-plugin</artifactId>
  <version>0.8.7</version>
  <executions>
     <execution>
        <goals>
           <goal>prepare-agent</goal>
        </goals>
     </execution>
     <execution>
        <id>report</id>
        <phase>prepare-package</phase>
        <goals>
           <goal>report</goal>
        </goals>
     </execution>
  </executions>
</plugin>
```

Reportes de cobertura de código

Al ejecutar los test implementados con JUnit, se pondrá en marcha automáticamente el agente de JaCoCo, el mismo, creará un reporte de cobertura de código en formato binario dentro del directorio target del proyecto - *target/jacoco.exec*.
Otras tools o plugins pueden interpretar estos resultados, como por ejemplo SonarQube.

Si deseamos revisar este reporte localmente podemos acceder al archivo generado en el directorio *target/site/jacoco/index.html* abriendo dicha página en cualquier navegador para visualizar los resultados.

Conclusiones

Como hemos podido ver, existen diversas herramientas en el ecosistema de Java que nos ayudan a controlar y por tanto conseguir una buena cobertura de código, es recomendable aprovechar estas herramientas a la hora de crear tests, en pos de seguir trabajando para entregar un producto de calidad a nuestros clientes.

DigitalHouse>