code
URJC

Ampliación de Ingeniería del Software

Colección de ejercicios

Universidad Rey Juan Carlos

Micael Gallego
Correo: micael.gallego@urjc.es
Twitter: @micael_gallego

Francisco Gortázar
Correo: francisco.gortazar@urjc.es
Twitter: @fgortazar

Michel Maes
michel.maes@urjc.es

Óscar Soto
oscar.soto@urjc.es

code
URJC

Micael Gallego, Francisco Gortázar, Michel Maes, Óscar Soto

Ampliación de Ingeniería del Software

code URJC

Indice

- La asignatura se compone de 3 temas
  - Tema 1. Pruebas y calidad del software
  - Tema 2. Mantenimiento y evolución del software
  - Tema 3. Gestión de la configuración del software

01-2023

code
URJC

Ampliación de Ingeniería del Software

Tema 1

Pruebas y Calidad del Software

Universidad Rey Juan Carlos

Micael Gallego
Correo: micael.gallego@urjc.es
Twitter: @micael_gallego

Francisco Gortázar
Correo: francisco.gortazar@urjc.es
Twitter: @fgortazar

Michel Maes
michel.maes@urjc.es
Óscar Soto
oscar.soto@urjc.es

01-2023

code URJC

Ampliación de Ingeniería del Software

Tema 1 – Pruebas y Calidad del Software

Tema 1.2 – Pruebas Unitarias

Universidad Rey Juan Carlos

Micael Gallego
Correo: micael.gallego@urjc.es
Twitter: @micael_gallego

Francisco Gortázar
Correo: francisco.gortazar@urjc.es
Twitter: @fgortazar

Michel Maes
michel.maes@urjc.es
Óscar Soto
oscar.soto@urjc.es

01-2023

code
URJC

Micael Gallego, Francisco Gortázar, Michel Maes, Óscar Soto

Universidad
Rey Juan Carlos

Índice de Ejercicios

Casos de Test

- **Ejercicio 1**
  - Implementa varios tests de la clase `Complex`
  - Comprueba que el complejo `Complex(0,0)` tiene la parte real y la parte imaginaria con valor 0
  - Comprueba que `Complex(0,0)` es el valor neutro de la operación suma:

```
Complex(0,0) + Complex(1,1) == Complex(1,1)
Complex(1,1) + Complex(0,0) == Complex(1,1)
```

Casos de Test

- **Ejercicio 2**
  - Transforma el Ejercicio 1 para usar Test fixtures
  - Define un atributo zero que se inicializa en un método setUp anotado como `@BeforeEach`
  - Ese atributo se usará siempre que se necesite el número complejo 0+0i

```
zero + Complex(1,1) == Complex(1,1)

Complex(1,1) + zero == Complex(1,1)
```

Casos de Test

- **Ejercicio 3**
  - Implementa un test que verifique que el valor absoluto de un número complejo (**método abs()**) se calcula correctamente
  - Define **varios número complejos** de ejemplo y verifica el cálculo para ellos
  - Usa tests parametrizados **`@ParameterizedTest`**

Casos de Test

- **Ejercicio 4**
  - Mejora el ejercicio 3 para que en cada test se muestre el número complejo su valor absoluto

Aserciones

- **Ejercicio 5**
  - Implementar un test que verifique que el recíproco de Zero Complex(0,0) eleva una excepción ArithmeticException con el mensaje "division by zero"
  - Como ahora mismo la clase Complex no está implementada de esa forma, el test debería fallar
  - Modifica la clase Complex para que realmente eleve una excepción cuando se intente calcular el recíproco de Complex(0,0)
  - Después de modificar el SUT, el test debería pasar

Matchers

- **Ejercicio 6**
  - Cambia las aserciones del Ejercicio 5 para que usen los matchers adecuados

AssertJ

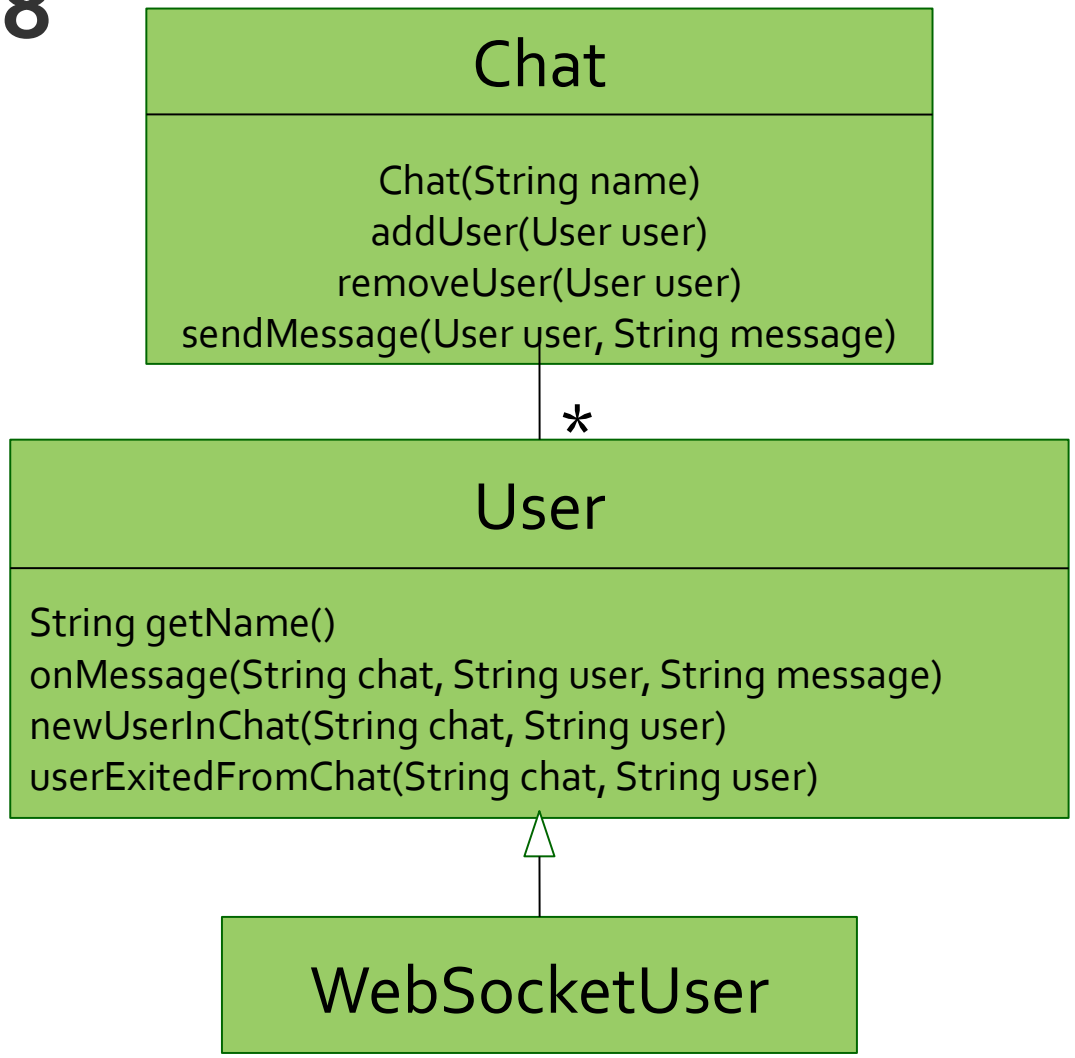
- **Ejercicio 7**
  - Convierte el Ejercicio 6 para que use la librería AssertJ
  - Cambia los matchers de Hamcrest por aserciones de AssertJ
  - Verifica la excepción con AssertJ

Dobles

- **Ejercicio 8**
  - Se quiere implementar una **aplicación web de chat** con websockets en Java
  - Se pide implementar la **clase chat** que contiene los usuarios incluidos en el chat
  - Cada vez que un usuario se añade o deja el chat, se envía una **notificación** al resto de usuarios.
  - Cada vez que un usuario envía un mensaje al chat, se **reenvía a todos los demás usuarios**

Dobles

- Ejercicio 8



Dobles

- **Ejercicio 8**
  - Se pide implementar un **test unitario de la clase Chat**
  - Hay que implementar un **mock de User** y **verificar** que sus métodos son invocados con los parámetros adecuados cuando se realizan las acciones en la clase Chat:
    - Añadir un usuario
    - Eliminar un usuario
    - Enviar un mensaje

Dobles

- **Ejercicio 9**
  - Se pide ampliar la clase chat para que use un **MediaServer**
  - El **MediaServer** se pasará como dependencia en el constructor
  - Hay que implementar los tests correspondientes que verifiquen que el **MediaServer** es usado como debería y que su comportamiento afecta al comportamiento del chat

Dobles

- **Ejercicio 9**
  - El MediaServer tiene el siguiente interfaz:
    - boolean allowMoreUsers()
    - addUser()
    - removeUser()
  - Cuando se vaya a añadir un usuario a un chat, se deberá preguntar al **MediaServer si tiene capacidad.**
  - En caso de que no tenga capacidad, el método addUser deberá devolver una **excepción NotEnoughResources**

01-2023

code URJC

Ampliación de Ingeniería del Software

Tema 1 – Pruebas y Calidad del Software

Tema 1.4 – Pruebas de sistema: Web

Universidad Rey Juan Carlos

Micael Gallego
Correo: micael.gallego@urjc.es
Twitter: @micael_gallego

Francisco Gortázar
Correo: francisco.gortazar@urjc.es
Twitter: @fgortazar

Michel Maes
michel.maes@urjc.es
Óscar Soto
oscar.soto@urjc.es

code
URJC

Índice de Ejercicios

Selenium WebDriver: WebDriverManager

code URJC

- **Ejercicio 1: Tests de Web de Anuncios**
  - Implementa tests de selenium de la aplicación de Gestión de Anuncios
  - Crear y eliminar anuncios
  - Comprobar que el nombre del usuario aparece automáticamente en el segundo mensaje creado

01-2023

code URJC

Ampliación de Ingeniería del Software

Tema 1 – Pruebas y Calidad del Software

Tema 1.6 - Testing de Sistema: APIs REST

Universidad Rey Juan Carlos

Micael Gallego
Correo: micael.gallego@urjc.es
Twitter: @micael_gallego

Francisco Gortázar
Correo: francisco.gortazar@urjc.es
Twitter: @fgortazar

Michel Maes
michel.maes@urjc.es

Óscar Soto
oscar.soto@urjc.es

code
URJC

Índice de Ejercicios

- **Ejercicio 1: Tests de API REST de Items**
  - Implementa los tests restantes con REST Assured de la API de items descrita previamente
    - Comprobar que al crear un item lo podemos recuperar
    - Comprobar que al borrar un item, no lo podemos recuperar (hay que crearlo en el mismo test)

01-2023

code URJC

Ampliación de Ingeniería del Software

Tema 1 – Pruebas y Calidad del Software

Tema 1.8 – Análisis estático de código

Universidad Rey Juan Carlos

Micael Gallego
Correo: micael.gallego@urjc.es
Twitter: @micael_gallego

Francisco Gortázar
Correo: francisco.gortazar@urjc.es
Twitter: @fgortazar

Michel Maes
michel.maes@urjc.es
Óscar Soto
oscar.soto@urjc.es

code
URJC

Índice de Ejercicios

Ejercicio 1

- **API REST con Spring**
  - Requisitos: Java 17
  - Clonar el proyecto de git con el comando:

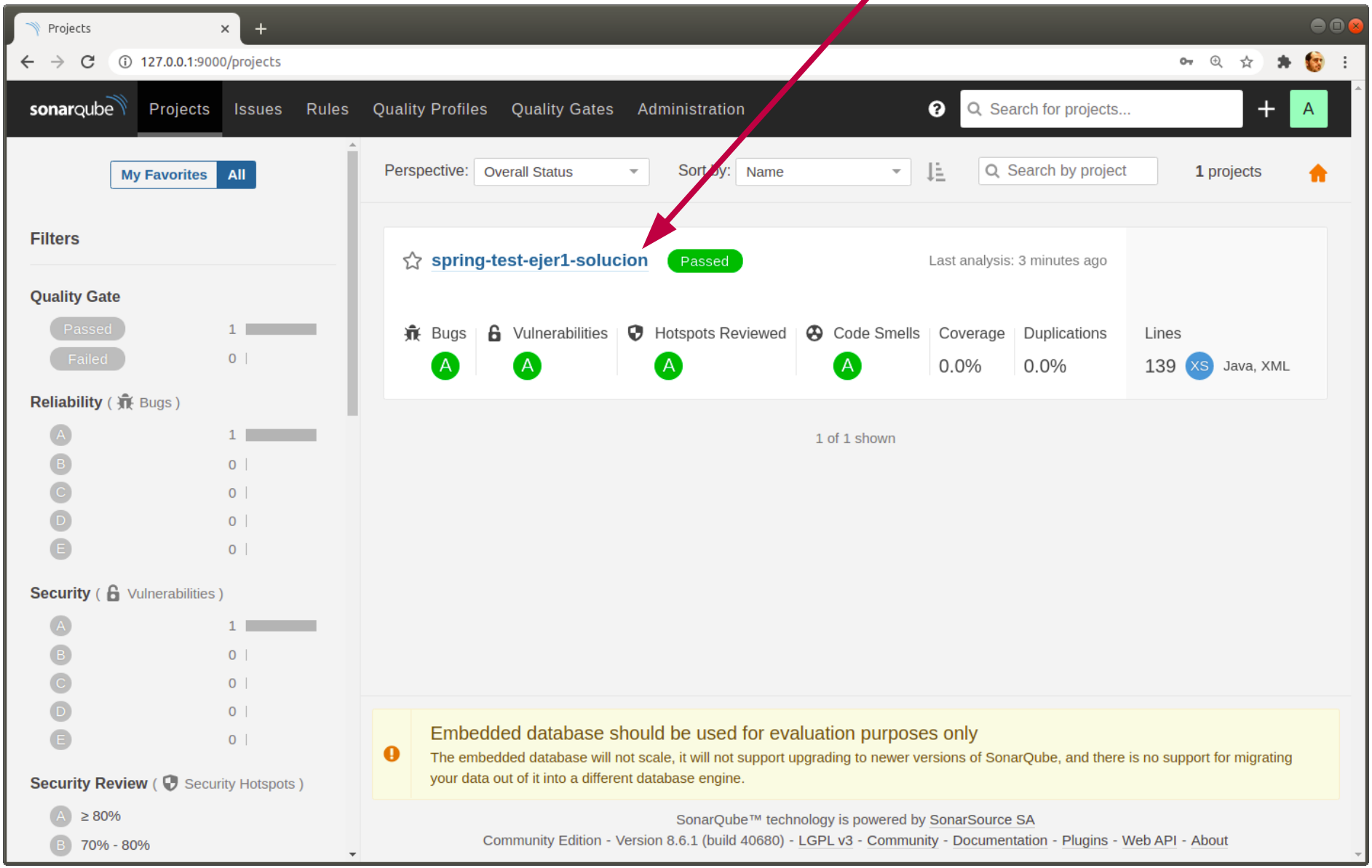
    ```
    $ git clone https://github.com/MasterCloudApps/2.4.Pruebas-de-servicios-web.git
    ```

  - Compilar el proyecto:

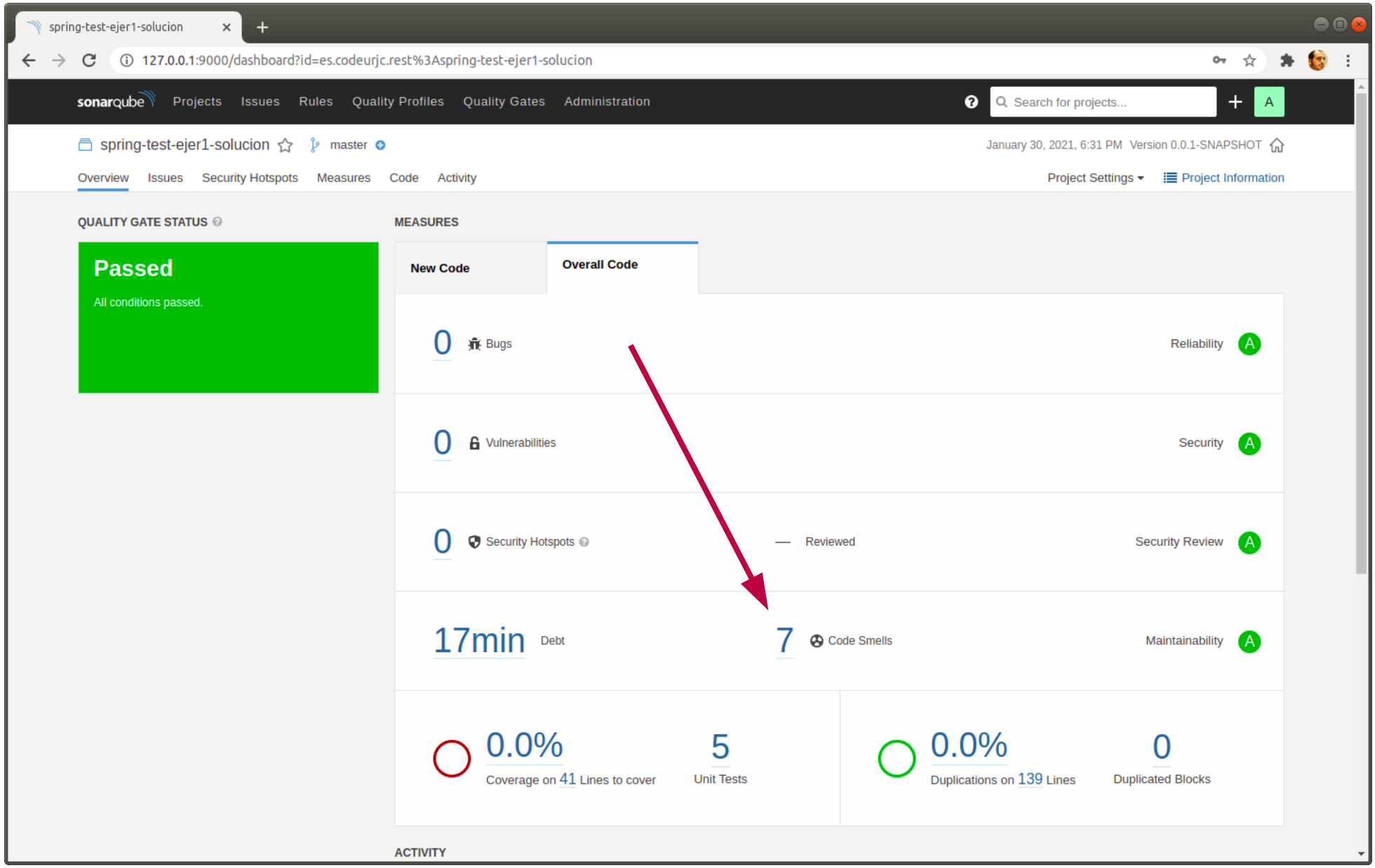
    ```
    $ cd tema2/spring-test-ejer1_solucion
    $ mvn clean install
    ```

  - Analizar con sonar:
    - `$ mvn sonar:sonar`
  - Abrir el navegador en la URL http://localhost:9000

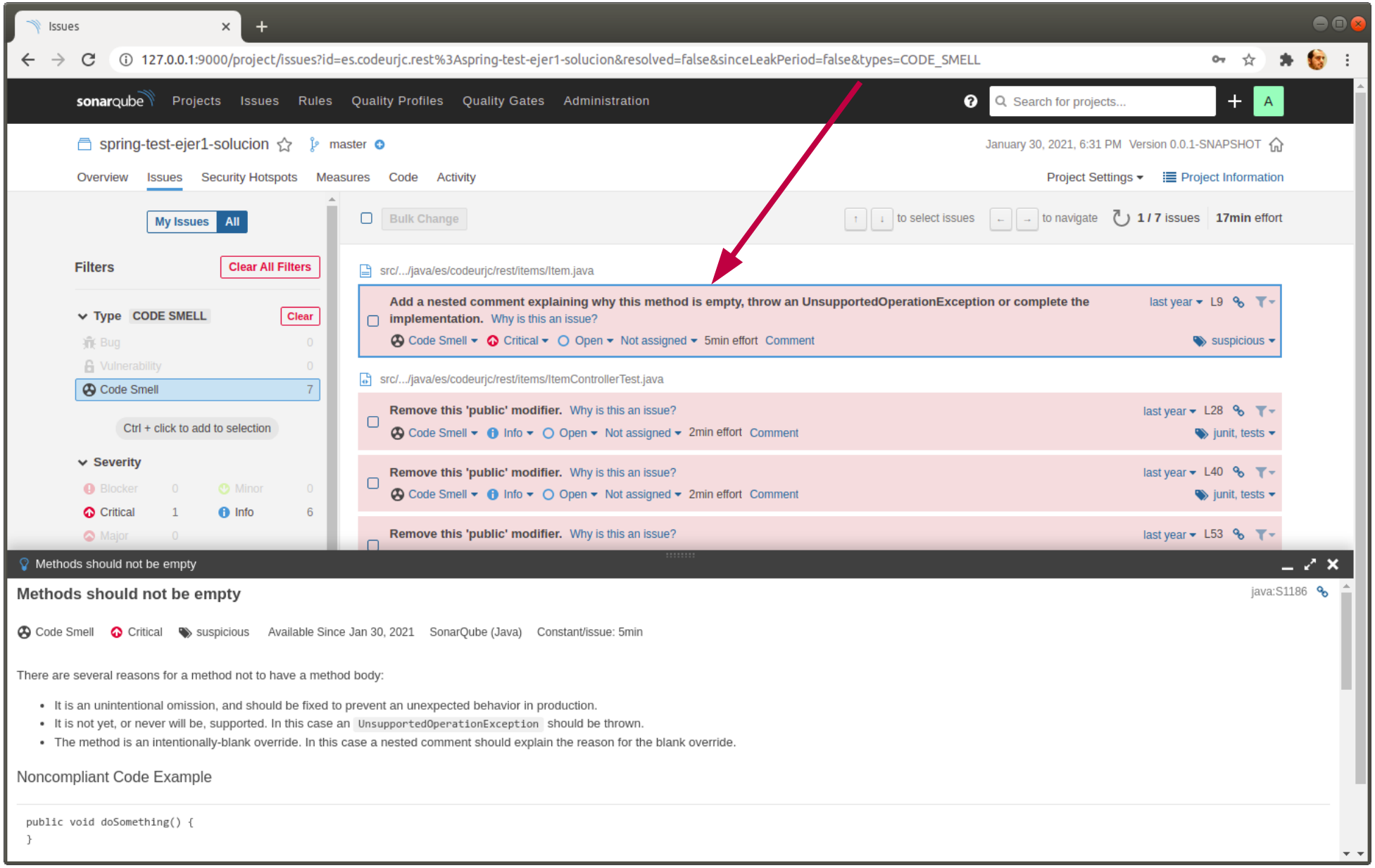
Ejercicio


Projects
127.0.0.1:9000/projects
sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
Search for projects...
A
My Favorites
All
Perspective:
Overall Status
Sort by:
Name
Search by project
1 projects
Filters
Quality Gate
Passed 1
Failed 0
Reliability (Bugs)
A 1
B 0
C 0
D 0
E 0
Security (Vulnerabilities)
A 1
B 0
C 0
D 0
E 0
Security Review (Security Hotspots)
A ≥ 80%
B 70% - 80%
spring-test-ejer1-solucion
Passed
Last analysis: 3 minutes ago
Bugs
Vulnerabilities
Hotspots Reviewed
Code Smells
Coverage
Duplications
Lines
A
A
A
A
0.0%
0.0%
139
XS
Java, XML
1 of 1 shown
Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.
SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 8.6.1 (build 40680) - LGPL v3 - Community - Documentation - Plugins - Web API - About

Ejercicio


spring-test-ejer1-solucion
127.0.0.1:9000/dashboard?id=es.codeurjc.rest%3Aspring-test-ejer1-solucion
sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
Search for projects...
spring-test-ejer1-solucion
master
January 30, 2021, 6:31 PM Version 0.0.1-SNAPSHOT
Overview
Issues
Security Hotspots
Measures
Code
Activity
Project Settings
Project Information
QUALITY GATE STATUS
Passed
All conditions passed.
MEASURES
New Code
Overall Code
0 Bugs
Reliability A
0 Vulnerabilities
Security A
0 Security Hotspots
Reviewed
Security Review A
17min Debt
7 Code Smells
Maintainability A
0.0%
Coverage on 41 Lines to cover
5
Unit Tests
0.0%
Duplications on 139 Lines
0
Duplicated Blocks
ACTIVITY

Ejercicio


Issues
127.0.0.1:9000/project/issues?id=es.codeurjc.rest%3Aspring-test-ejer1-solucion&resolved=false&sinceLeakPeriod=false&types=CODE_SMELL
sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
Search for projects...
spring-test-ejer1-solucion
master
January 30, 2021, 6:31 PM Version 0.0.1-SNAPSHOT
Overview
Issues
Security Hotspots
Measures
Code
Activity
Project Settings
Project Information
My Issues
All
Bulk Change
to select issues
to navigate
1 / 7 issues
17min effort
Filters
Clear All Filters
Type CODE SMELL
Clear
Bug 0
Vulnerability 0
Code Smell 7
Ctrl + click to add to selection
Severity
Blocker 0
Minor 0
Critical 1
Info 6
Major 0
src/.../java/es/codeurjc/rest/items/Item.java
Add a nested comment explaining why this method is empty, throw an UnsupportedOperationException or complete the implementation. Why is this an issue?
last year L9
Code Smell Critical Open Not assigned 5min effort Comment
suspicious
src/.../java/es/codeurjc/rest/items/ItemControllerTest.java
Remove this 'public' modifier. Why is this an issue?
last year L28
Code Smell Info Open Not assigned 2min effort Comment
junit, tests
Remove this 'public' modifier. Why is this an issue?
last year L40
Code Smell Info Open Not assigned 2min effort Comment
junit, tests
Remove this 'public' modifier. Why is this an issue?
last year L53
Methods should not be empty
Methods should not be empty
java:S1186
Code Smell Critical suspicious Available Since Jan 30, 2021 SonarQube (Java) Constant/issue: 5min
There are several reasons for a method not to have a method body:
It is an unintentional omission, and should be fixed to prevent an unexpected behavior in production.
It is not yet, or never will be, supported. In this case an UnsupportedOperationException should be thrown.
The method is an intentionally-blank override. In this case a nested comment should explain the reason for the blank override.
Noncompliant Code Example
public void doSomething() {
}

Ejercicio 1

- Elimina los problemas de calidad (Issues) del código del ejemplo 1
- Analiza el código de nuevo
- Observa la evolución

01-2023

code URJC

Ampliación de Ingeniería del Software

Tema 2

Mantenimiento y evolución del software

Universidad Rey Juan Carlos

Micael Gallego
Correo: micael.gallego@urjc.es
Twitter: @micael_gallego

Francisco Gortázar
Correo: francisco.gortazar@urjc.es
Twitter: @fgortazar

Michel Maes
michel.maes@urjc.es

Óscar Soto
oscar.soto@urjc.es

01-2023

code
URJC

Ampliación de Ingeniería del Software

Tema 3

Gestión de la configuración del Software

Universidad Rey Juan Carlos

Micael Gallego
Correo: micael.gallego@urjc.es
Twitter: @micael_gallego

Francisco Gortázar
Correo: francisco.gortazar@urjc.es
Twitter: @fgortazar

Michel Maes
michel.maes@urjc.es
Óscar Soto
oscar.soto@urjc.es

01-2023

code URJC

Ampliación de Ingeniería del Software

Tema 3 – Gestión de la configuración del Software

Tema 3.2 – Gestión del código fuente

Universidad Rey Juan Carlos

Micael Gallego
Correo: micael.gallego@urjc.es
Twitter: @micael_gallego

Francisco Gortázar
Correo: francisco.gortazar@urjc.es
Twitter: @fgortazar

Michel Maes
michel.maes@urjc.es
Óscar Soto
oscar.soto@urjc.es

code
URJC

Índice de Ejercicios

Git

- **Ejercicio 1**
  - Crear un fichero README.md con texto
  - Añadirlo al repositorio: comitear el fichero README.md
  - Verificar que tenemos dos commits en el repo

Trabajo con el repositorio

- **Ejercicio 2**
  - Añadir un fichero LICENSE y editar el fichero README.md
  - Comitear el fichero LICENSE primero
  - Comitear el fichero README.md después

Ramas (*branches*)

- **Ejercicio 3**
  - Añadir una rama llamada “documentation”
  - En dicha rama:
    - Crear una carpeta docs
    - Movernos a la carpeta docs
    - Añadir los siguientes ficheros (no importa el contenido):
      - index.html
      - styles.css
  - Comitear

Ramas (*branches*)

- **Ejercicio 4**
  - Cambiar a master (hacer un checkout sin -b)
    - git checkout master
  - En master:
    - Modificar el fichero README.md
    - Comitear

Borrar commits

- **Ejercicio 5**
  - Crea un fichero nuevo en master
  - Modifica el contenido del README.md
  - Ejecuta: **`$ git reset --hard HEAD`**
  - Comprueba que el contenido del README.md se restaura al contenido original pero que el nuevo fichero (*untracked*) sigue intacto

Borrar commits

- **Ejercicio 6**
  - Modifica el contenido de README.md y README2.md
  - Retrocede master dos commits anteriores
  - Revisa que los ficheros README.md y README2.md están intactos (no han perdido información)
  - Haz un nuevo commit

Borrar commits

- **Ejercicio 7**
  - Agrupar los dos últimos commits en un único commit

Resolviendo conflictos

- **Ejercicio 8**
  - Crea una nueva rama, modifica un fichero y comitea
  - Vuelve a master, modifica el mismo fichero en la misma línea y comitea
  - Haz merge resolviendo los conflictos

Pull Requests de GitHub

- **Ejercicio 9**
  - Implementa una nueva feature en una nueva rama
  - Acepta esa rama en master con un commit que fusione los cambios

Pull Requests de GitHub

- **Ejercicio 10**
  - Por parejas (Alice y Bob)
  - Crear un repositorio con README.md en una de las cuentas
  - Dar permiso al compañero/a
  - Clonar el repositorio ambos
  - Alice arrancará una rama donde modificará el README.md
  - Bob hará lo propio en otra rama
  - Ambos comitearán, subirán la rama y abrirán un PR
  - Uno de los dos aceptará el PR primero
  - El otro no podrá
  - Debe conseguir subir sus cambios a master a través del PR

Pull Requests de GitHub

- **Ejercicio 11**
  - Por parejas (Alice y Bob)
  - Continuando con el ejercicio 10
  - Bob creará una nueva rama, la empujará a GitHub y creará un PR
  - Alice debe sacar la rama de Bob e incorporar nuevos commits empujándolos después a GitHub
  - Bob debe rebasar los cambios de Alice, incluir nuevos commits y empujarlos a GitHub
  - Bob debe aceptar el PR

Gestionando el repositorio

- **Ejercicio 12**
  - Bob anota el último PR como v1.0
  - Alice hace un checkout de este tag v1.0

Git flow

- Ejercicio 13
  - Repetir el proceso con dos features
    - Se crea la feature f1, se introducen un par de commits
    - Sin cerrar f1, se crea la feature f2
    - Se introducen un par de commits en f2 que toquen el mismo fichero que f1
    - Se finaliza f2
    - Se finaliza f1

01-2023

code URJC

Ampliación de Ingeniería del Software

Tema 3 – Gestión de la configuración del Software

Tema 3.4 – Integración Continua con GitHub Actions

Universidad Rey Juan Carlos

Micael Gallego
Correo: micael.gallego@urjc.es
Twitter: @micael_gallego

Francisco Gortázar
Correo: francisco.gortazar@urjc.es
Twitter: @fgortazar

Michel Maes
michel.maes@urjc.es

Óscar Soto
oscar.soto@urjc.es

code
URJC

Micael Gallego, Francisco Gortázar, Michel Maes, Óscar Soto

Índice de Ejercicios

Integración Continua con GitHub Actions

code URJC

Ejercicio 1

- Realiza un fork sobre el ejemplo 1
  - https://github.com/URJC-AIS/github-actions
- Parametriza el workflow para que utilice distintas versiones de Java y distintos runners
- Comprueba y analiza los resultados

01-2023

code URJC

Ampliación de Ingeniería del Software

Tema 3 – Gestión de la configuración del Software

Tema 3.6 – Entrega Continua con Docker y GitHub Actions

Universidad Rey Juan Carlos

Micael Gallego
Correo: micael.gallego@urjc.es
Twitter: @micael_gallego

Francisco Gortázar
Correo: francisco.gortazar@urjc.es
Twitter: @fgortazar

Michel Maes
michel.maes@urjc.es

Óscar Soto
oscar.soto@urjc.es

code
URJC

Micael Gallego, Francisco Gortázar, Michel Maes, Óscar Soto

Índice de Ejercicios

Entrega continua con Docker y Github Actions

code URJC

Ejercicio 1

- Realiza un fork del ejemplo

  https://github.com/URJC-AIS/docker-example

- Realiza los cambios oportunos para que el tag de la imagen Docker sea la versión del pom.xml y la fecha siguiendo el siguiente formato

  <user_name>/<image_name>:X.X.X-Timestamp

- Ejecuta el workflow y comprueba que se crea la imagen en DockerHub con el formato correcto