PRINCIPLES OF SOFTWARE DEVELOPMENT

2019-2020

Universidad Carlos III de Madrid

Final Practice 2019/2020

PRINCIPLES OF SOFTWARE DEVELOPMENT

Final Practice

Universidad Carlos III de Madrid. Escuela Politécnica Superior

Javier García Guzmán

SECTION 1

Objectives

The goal of this final practice is evaluating the learning level that students have acquired within the techniques studied in this course for specifying, designing, programming, and testing a simple software component.

User Requirements

Transport4Future is a company that is specialized in the development of technologies for autonomous vehicles.

To ensure that only authorized and registered sensors and actuators can access the autonomous vehicle network, Transport4Future has decided to evolve the token management software component.

The requested component will be developed in Java, J2EE platform. It will be delivered under the form of a JAR file, which provides access to the interface programming methods that will allow its integration in the information services implemented in the vehicles.

The new functional requirements for this component that must be considered in the scope of this final practice are the following:

- First of all, the method VerifyToken must decode the base64url encoded string corresponding to the token to verify. Therefore, the Token Value field must be removed from the Token class, so that this value cannot be directly searched in the Token Store.

- Before its expiration date, It will be possible to deactivate the license corresponding to a specific machine by specifying the following data in the JSON input file:

```
{
  "Token Value": "<Base64url encoded String>",
  "Type of revocation":"<Temporal | Final >",
   "Reason":"<String having 100 characters max >"
}
```

The method will return an e-mail address (Notification e-mail) corresponding to the token that has been deactivated.

By means of an exception, information about error messages will be provided. T he error messages to consider are:

- The input file has any problem related to its format or to its access.
- The token received does not exist.
- The token received has expired.
- The token was previously revoked by this method.

The interface method defined to provide the functionality is as follows:

```
String RevokeToken (String FilePath) throws TokenManagementException;
// String represents the notification e-mail to be returned
// String FilePath represents the path to the file including the input required for the functionality
// TokenManagementException represents the possible error situations
```

NOTE: The inclusion of this functionality may mean updating previously developed functionalities.

- The management component will allow executing operations on the connected components. To do it, the management component will receive a JSON file with the following format:

```
{
  "Token Value": "<Base64url encoded String >",
  "Type of operation":"<Send Information from Sensor| Send Request to Actuator | Check State >"
}
```

The component must verify that the Token received is valid.

The component will return:

- A Boolean value indicating whether or not the operation can be executed. If the operation is related to a Sensor, it only can be executed if the token received corresponds to a Sensor. In the same way, if the operation is related to an Actuator, it only can be executed if the token received corresponds to an Actuator. The operation "Check State" if authorized for both sensors and actuators.
- By means of an exception, information about error messages will be provided. The error messages to consider are:
    - The input file has any problem related to its format or to its access.
    - The token received does not exist or is not valid.
    - The device represented by the token cannot execute the requested operation.

The interface method defined to provide the functionality is as follows:

```
boolean ExecuteAction (String FilePath) throws TokenManagementException;
// Boolean represents the success after performing the task
// String FilePath represents the path to the file including the input required for the functionality
// TokenManagementException represents represents the possible error situations
```

Section 2

Rules and Procedures

The final practice will be completed in pairs.

Documentation to provide

Each team must push the solution to the project specified by the professor in their Git repository.

The solution should contain the following information.

A. Each team must apply the Test-Driven Development techniques, defining and automatizing the tests for each public function of the new version of the component. You must combine the test techniques that you consider adequate for each method. This decision will be taken into account in the evaluation of the practice. The information to provide is:
   - A PDF file showing the equivalence classes, the boundary limits, grammars, derivation trees and/or control flow graphs used to identify the different test cases for each of the public interface methods defined for the component
   - The source code that allows automating the test cases is defined in the previous document, including proper comments. The comments should follow this structure:

```
/* Test Case: <Id - Name>
* Equivalence class, boundary limit OR related derivation tree node: <Value>
* Testing method: <Equivalence class| Boundary limit| Syntax analysis | Structural <Basic paths – Loop tests>>
* Expected result: <Description>
*/
```

   - A JUnit record including the results for each test case included in the test set.

B. The source code corresponding to the implemented functionalities. This code must be refactored properly, satisfying the Singleton pattern for the classes TokenManager y TokenStore y TokenRequestStore, and the Strategy pattern for the ones that receive inputs from JSON and the ones that generate a hash code from different algorithm types.

Each team must push the solution to the corresponding project in its GitLab repository.

The deadline to submit the exercise solution is set on 28/05/2020 before 23:55.

Following the subject evaluation rules, if a student does not send the exercise solution before the fixed deadline, the student will be scored with 0 points.

Evaluation Criteria

Evaluation criteria will be the same criteria set for each of the requested parts as specified in the corresponding guided exercise.

Suggestions

Each student should keep a copy of the solution delivered until the end of the subject.