SMA:
Software Modeling and Analysis

Practical Session

Week 12

Assignment 11

Discussion

A11 - Exercise 01 | Exploring projects (3 pts)

In GT, a project does not have a clear structure, but instead it is consists of multiple packages. In this exercise, we will work on an easy to use project explorer that can present all the packages or classes of a project.

a) Describe the steps necessary to add an extension method.

GT

i) Open a class whom behavior you want to extend in Calypso.
ii) Add the extension method in the class.
iii) In the category, input * and name of your package.
iv) You can also select your target package from the dropdown list.

Pharo

i) Open the class whom behavior you want to extend in Calypso.
ii) In the menu, choose “Move to package”.
iii) Select your target package.

Programmatically

```
ExistingClass compile:
'myMethod’
^ true classified: '\*MyPackage'
```

A11 - Exercise 01 | Exploring projects (3 pts)

In GT, a project does not have a clear structure, but instead it is consists of multiple packages. In this exercise, we will work on an easy to use project explorer that can present all the packages or classes of a project.

b) How many unique projects exist in the GT image?

```
allPackages :=
    RPackageOrganizer default packages
        collect:  #packageGroup.
    ^ allPackages asSet.
```

c) Collect all packages of the `Collections` implementation.

```
allPackages :=
    RPackageOrganizer default packages
        select:  [ :e | e packageGroup match:  'Collections'].
    ^ allPackages asSet.
```

A11 - Exercise 02 | Exploring hierarchies (7 pts)

In this exercise, you have to work on the GT class hierarchy to match the project structure. You are supposed to consider the `Object` class as root of all hierarchies, because all classes inherit from `Object`.

a) Calculate the depth of the `Collection` class using the class hierarchy available in GT.

The hierarchy depth is 2.
You can retrieve that number with: `Collection classDepth - 1.`

b) Does the superclass of `HashedCollection` reside in the same package?

Yes.
`HashedCollection isSuperclassInAnotherPackage.`

c) Do the subclasses of `HashedCollection` reside in the same package?

No.
`HashedCollection isSubclassesInAnotherPackage.`

A11 - Exercise 02 | Exploring hierarchies (7 pts)

In this exercise, you have to work on the GT class hierarchy to match the project structure. You are supposed to consider the `Object` class as root of all hierarchies, because all classes inherit from `Object`.

d) Calculate the depth of `Collection` in the package hierarchy.
The package hierarchy depth is 1.
`(HA ClassExplorer asExplorerClass: Collection) calculatePackageLevel.`

e) Add reasonable class comments to `ProjectAnalyzer` and `HierarchyAnalyzer`.
You can find the solution in the Github repository.

f) Implement the two methods `calculatePackageLevel` and `calculateProjectLevel` in the class `PA_ProjectExplorer`. You can find the corresponding implementations for classes (instead of projects) in `HA_ClassExplorer`.
You can find the solution in the Github repository.

Assignment 12

Preview

A12 - Exercise 01 | General questions (6 pts)

a) Explain the term fuzz testing.

b) Explain the term smart fuzzer.

c) What is the role of symbolic execution engines in white box fuzzers?

d) Explain one limitation of symbolic execution?

e) Name three concolic execution engines.

f) How does concolic execution extend symbolic execution?

A12 - Exercise 02 | AFL tool (1 pt)

Answer which of the statements below are correct with respect to the AFL fuzzer. You do not need to justify or elaborate your answer.

AFL is a grey box fuzzer.

AFL instruments the source code of the target program to measure code coverage.

AFL also supports the QEMU mode when the source code is unavailable.

AFL was released in 2016 and became an international standard in fuzz testing.

A12 - Exercise 03 | Fuzzing in practice (3 pts)

Preparation: For this exercise, we work with a Debian-based operating system, because all required tools are already available from its package management system. We already prepared a virtual machine (VM) for you with all the necessary tools and our compiled VulnerableApp that can be run with VirtualBox.

Your tasks:

a) Create a text file that only contains the term Fuzztesting. Next, parametrize the zzuf fuzzer with a seed value of 2 and a ratio of 0.01, and pass the content of your text file (e.g., test.txt) to it. What is the resulting command string, and what is the output message? (2 pts)

b) Your task is to find seed values between 20 and 30 where zzuf leads to a crash in VulnerableApp. (1 pt)

Next week:
Q & A Session

Submit your questions by email to pascal.gadient@inf.unibe.ch.

One BONUS point per question, up to three points.

Mock Exam

Zoom meeting will open at 11:45am.

See you back there!