Samples of the forward and reverse engineering

Performing the forward engineering

The example shows how to perform a simple java code generation.

Step #1. Creating *CodeEngineeringSet*

```
    Project project = Application.getInstance().getProject();
    String name = "sample CE project";
    String workingDir = OPENAPI_DATA_DIRECTORY_PATH;

    // create a working package
    ElementsFactory ef = project.getElementsFactory();
    Package workingPackage = ef.createPackageInstance();
    workingPackage.setName("my working package");
    workingPackage.setOwner(project.getModel());

    // creating a code engineering set
    CodeEngineeringSet javaGenerationSet = CodeEngineeringManager.
createCodeEngineeringSet(
                               CodeEngineeringConstants.Languages.JAVA,
null, name, project, workingPackage, workingDir);
```

Step #2. Adding model elements to *CodeEngineeringSet*

```
        Project project = Application.getInstance().getProject();

         // create a new element
         ElementsFactory ef = project.getElementsFactory();
         Class classA = ef.createClassInstance();
         classA.setName("ClassA");
        classA.setOwner(project.getModel());
        List<BaseElement> modelsForSample = new ArrayList<BaseElement>();
        modelsForSample.add(classA);
        javaGenerationSet.addElementsToCodeEngineeringSet(modelsForSample);
```

Step #3. Performing the *CodeEngineeringSet* generation

```
    CodeEngineeringManager.generate(javaGenerationSet);
```

Performing the reverse engineering

The example shows how to perform a simple java code reverse.

Step #1. Creating *CodeEngineeringSet*

```
    Project project = Application.getInstance().getProject();
    String name = "sample CE project";
    String workingDir = OPENAPI_DATA_DIRECTORY_PATH;// e.g C:
\myworkingPackage

    // create a working package
    ElementsFactory ef = project.getElementsFactory();
    Package workingPackage = ef.createPackageInstance();
    workingPackage.setName("my working package");
    workingPackage.setOwner(project.getModel());

    // creating a code engineering set
    CodeEngineeringSet ces = CodeEngineeringManager.
createCodeEngineeringSet(
                               CodeEngineeringConstants.Languages.JAVA,
null, name, project, workingPackage, workingDir);
```

Here the null dialect is used for the java language, because java doesn't have any dialect.

Step #2. Adding the source code to *CodeEngineeringSet*

```
         ces.addAllFilesRecursivelyToCES(new File(workingDir + File.
separator + "test directory")); // starting from C:\myworkingPackage\test
directory\
```

This sets the given instance of the code engineering set working directory and adds all files from that directory. Set java classpaths for the project:

```
    String[] claspath = new String[] { path1, path2, path3, path4 };
    JavaCodeEngineeringManager.setJavaClasspath(project, claspath);
```

Step #3. Performing the reverse of *CodeEngineeringSet*

```
     CodeEngineeringManager.reverse(ces, false);
```