chem-bla-ics

Multiple unit test inheritance with JExample

Egon Willighagen

Published October 26, 2010

Citation
Willighagen, E. (n.d.). In *chem-bla-ics*. chem-bla-ics. https://doi.org/10.59350/b3fxm-rg432

Keywords
Java, Junit

Copyright
Copyright © Egon Willighagen 2010. Distributed under the terms of the Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

chem-bla-ics

Two months ago I wrote about JExample (see Specifying unit test dependencies with JExample). At the time, my examples did not include multiple unit test inheritance, but was informed later by @jexample that is possible. I just got time to try it in the Oscar project:

```
@Test public Oscar testConstructor()
throws URISyntaxException
{
  Oscar oscar = new Oscar();
  Assert.assertNotNull(oscar);
  return oscar;
}

@Given("#testConstructor")
public String testNormalize(Oscar oscar)
throws URISyntaxException
{
  String input = oscar.normalize(
    "This is a simple input string with benzene."
  );
  Assert.assertNotNull(input);
  return input;
}

@Given("#testConstructor,#testNormalize")
public List testTokenize(
  Oscar oscar, String input) throws Exception
{
  List tokens = oscar.tokenize(input);
  Assert.assertNotNull(tokens);
  Assert.assertNotSame(0, tokens.size());
  return tokens;
}

@Given("#testConstructor,#testTokenize")
public List testRecognizeNamedEntities(
  Oscar oscar, List tokens)
throws Exception
{
  List entities =
    oscar.recognizeNamedEntities(tokens);
  Assert.assertNotNull(entities);
  Assert.assertEquals(1, entities.size());
```

```
  return entities;
}
```

The pattern here is that each test method returns one variable, so that any method depending on two other unit test will have two parameters. The order is defined by the order in which they are given by the *@Given* clause.