Annotations in Java

When you create a new JUnit test-case class in Eclipse, you put JUnit 4 on the build path. JUnit 4 introduces some annotations that you can use in the JUnit test-case class. They are described below. You may have to import @Before, @BeforeClass, and others from package org.junit.

@Test Place this annotation before any procedure you wish to have executed when the JUnit testing class is run. For example, consider the code shown below. Procedure testB() will be called but testC() will not be called.

```
@Test
public void testB() { … }

public void testC() { … }
```

@Test (expected = *exception*.class) Use this annotation to test whether a computation throws an *exception*. Put Throwable or any subclass of Throwable you want in place of *exception*. For example, suppose a call on m.c(null) is supposed to throw a null-pointer exception. Write the testing procedure as shown below. The call on testEx succeeds only if a NullPointerException is thrown. If no exception is thrown or another exception is thrown, the call fails.

```
@Test(expected = NullPointerException.class)
public void testEx() {
    m.c(null);
}
```

Unfortunately, when one has assert statements, there may be lots of exception-throwing of to test. Using this method, one needs a different testing procedure for each one. It's not the best solution.

@Test(timeout=100) A call on a testing procedure with this annotation will halt with an error if it takes more than 100 milliseconds. Change the number 100 to anything you want.

@Before and @BeforeClass. Place @Before before a procedure, and that procedure is called before each call of an @Test-annotated procedure. Place @BeforeClass before a static procedure —it must be static— and that procedure is called before anything else.

To make this clear, we ran the class that appears in the box to the right. Each of the procedures prints a line, indicating what procedure it is. The output of running this is:

```
testBeforeClass
testBefore
test1
testBefore
test2
```

```
public class ATester {
   @Before
   public void testBefore()
      {System.out .println("testBefore");}

   @BeforeClass
   public static void testBeforeC()
      {System.out .println("testBeforeClass");}

   @Test
   public void test1()
      {System.out .println("test1");}

   @Test
   public void test2()
      {System.out .println("test2");}
}
```

Thus, if you have some static variables you want initialized before any @Test-annotated procedure is called, do it in a static @BeforeClass-annotated procedure. And, if you have some fields that should be initialized before each @Test-annotated procedure, do it in an @Before-annotated procedure.

Note that you don't know the order in which @Test-annotated procedures will be called. Don't rely on a particular ordering.

@After, @AfterClass . These two annotations are used on procedures to have them executed after each @Test-annotated procedure is called and after everything has been called, respectively. They may be needed if resources have been allocated and need to be deallocated. This is beyond the scope of this JavaHyperText.

@Ignore Put this annotation on a procedure and it will not be called, even if it also has an @Test annotation. You really don't need this; you could just comment out the @Test annotation.