Comparing to Hamcrest

FEST Assertions Module has moved to Github !

Check this page :https://github.com/alexruiz/fest-assert-2.x/wiki

The documentation below is for Fest 1.x which is no more maintained, we are focusing our effort to the 2.x version !

Hamcrest is a powerful library for writing custom matchers. Although FEST-Assert and Hamcrest have similar goals, their implementation and APIs are quite different. Here is an example using Hamcrest (borrowed from Martin Gilday's example posted at the TestNG mailing list):

```
import static org.hamcrest.MatcherAssert.assertThat;
import static org.hamcrest.Matchers.containsString;
import static org.hamcrest.Matchers.equalTo;
import static org.testng.Assert.assertTrue;

import org.testng.annotations.Test;

public class SomeTest {

  @Test public void withHamcrest() {
    assertTrue(true);
    int result = 2 + 3;
    assertThat("calculation", 5, equalTo(result));
    assertThat("Hello World", containsString("World"));
  }
}
```

This is the same example, using FEST's assertions:

```
import static org.fest.assertions.Assertions.assertThat;

import org.testng.annotations.Test;

public class SomeTest {

  @Test public void testApp() {
    assertThat(true).isTrue();
    int result = 2 + 3;
    assertThat(result).as("calculation").isEqualTo(5);
    assertThat("Hello World").contains("World");
  }
}
```

We like FEST-Assert's approach because:

1. Only one static import is needed (`org.fest.assertions.Assertions.assertThat`)
2. Allows us to use our IDE's "auto-complete" feature: we only type "." and we get the possible assertion methods for the value passed to `assertThat`
3. We can chain related assertion methods:

```
assertThat(yoda).isInstanceOf(Jedi.class)
                .isEqualTo(foundJedi)
                .isNotEqualTo(possibleSith);
```

Which one to use? Hamcrest or FEST-Assert? It is up to you...it depends on the needs of your project and your coding style!