Jakarta Cactus–what's new with the project?

Petar Tahchiev
(ptahchiev@apache.org)

Dedicated to my sister Hristina:

"… because you were always there for me …"

Who am I actually?

- Petar Tahchiev, 23
- Technical Consultant at HP
- Apache Jakarta Commiter
- Leader of the Bulgarian Java User Group
- Part-time assistant at Sofia University
- Following my masters degree at Sofia University
- Lead the Bulgarian translators team of NetBeans
- Blogger :-) (http://weblogs.java.net/blg/paranoiabla/)

How important is testing?(1)

Testing is the most simple method of ensuring the correctness of your code. The most important features of the testing philosophy are:

- Writing test cases is relatively easy
- Eliminates typical, and commonly-occurred errors
- Keeps the code clean (free of typo errors)

How important is testing?(2)

- Tends to be neglected from some developers, since it is time-consuming

– Goal: Accelerate programming and increase the quality of code

– Write tests sooner and then implement the code

– Test early and test often

Different types of testing...

- Unit testing
- Functional (GUI) testing
- Regression
- Distributed
- Distributed Testing
- ...

In-container testing: tools, methodologies ...

- Where Cactus fits?
- Cactus is an extension of JUnit
- JUnit is a regression testing framework written by Erich Gamma & Kent Beck
- Cactus is an in-container testing framework

In-container testing...

- What is in-container testing?
- Means that the test-cases are executed directly on the server.
- Allows usage of HTTP components (like request, response, cookies,...)
- Allows usage of container native libraries.
- Cactus is the most famous project for in-container testing.

A bit of Jakarta Cactus history

How Cactus really started?

- Created by Vincent Massol while at work in OCTO technologies
- Initially was named J2EEUnit and resided in the Sourceforge's repositories
- J2EEUnit was renamed to Cactus because J2EE was infringing on the Sun trademark
- Soon after Cactus was moved to Apache Jakarta project
- “Cactus found his family”

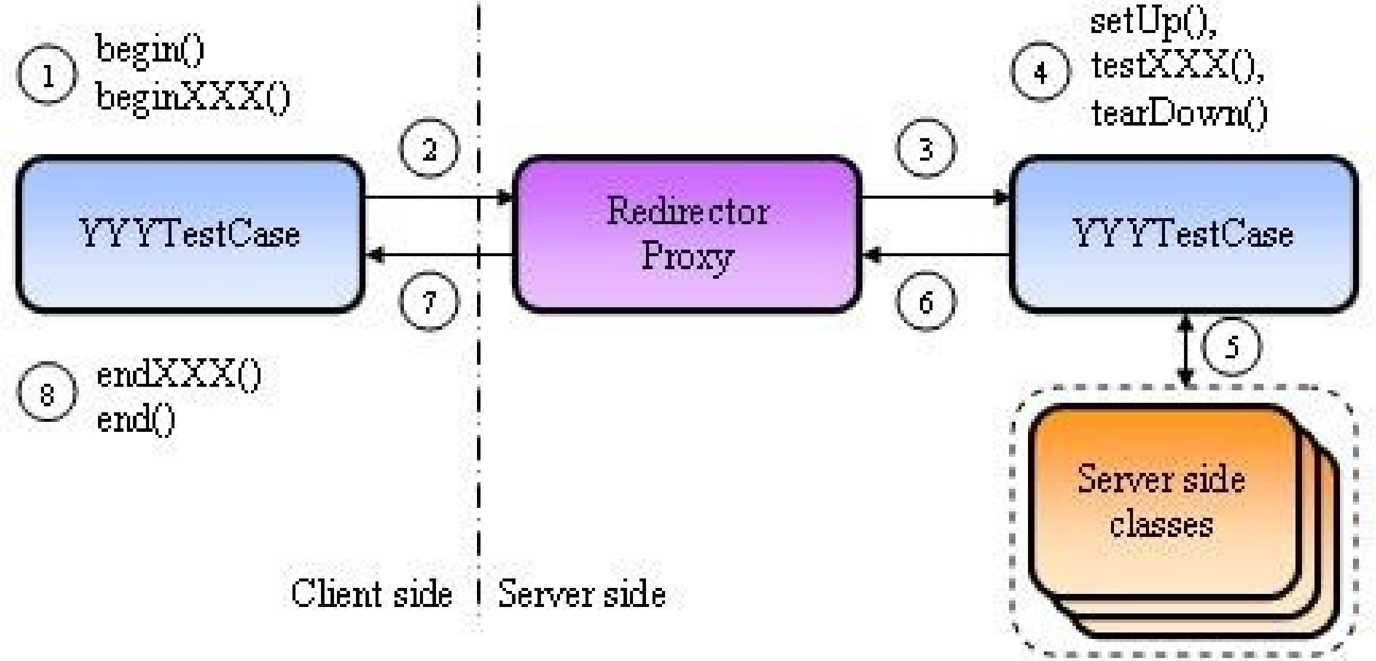
How Cactus really works?

Cactus is a really simple testing framework based on Junit.

- Used for testing server-side Java code(JSPs, Servlets, Filters, EJBs, ...)
- We write test-cases that extend some of cactus's test cases
- Deploy your test-cases on the server side, along with your code

How Cactus really works?

- You start the test-cases on the client side and they invoke the ones that are on the server-side



A sample example(1)

```
public class MyTest extends ServletTestCase
{
    public void testXXX()
    {
        [...]
        MyServletToTest servlet = new MyServletToTest();
        servlet.init(config);
        servlet.methodToTest();
    }
}
```

A sample Example(2)

```
<cactus warfile="${test.dir}/test.war" printsummary="yes"
              failureproperty="tests.failed">
    <classpath>
        [...]
     </classpath>
     <containerset>
        [...]
     </containerset>
  </cactus>
```

What is the current status of the project?

- The current version of the project is **1.7.2** from **March 26th, 2006**
- No single line committed in the repository since then
- Vincent Massol withdrew as a main developer
- Several bugs opened in the bugzilla
- Committers: me, Kazuhito Siguri(active on the mailing lists), Felipe Leme (lots of ideas and support)

My part of the game

What are my responsibilities and why am I here?

- First heard about Cactus a year ago on the Google Summer of Code programme
- Participated with Felipe in a project for migrating Cactus's build system from **Ant** to **Maven**
- This project didn't pass the initial vote at the ASF
- Since then I am trying to make it on my own

What's in the trunk anyway?

- For half an year I have been working in the Sourceforge repositories (I didn't have commit rights at the ASF)
- After that we migrated it in the ASF repositories
- My work resides not in the trunk of the project, but rather in a separate branch named CACTUS_TRUNK_MAMOUTH

Much ado about ...

So what are the changes I have made to the project?

- Migrated build system from Ant to Maven
- Cactus-cargo integration
- Cactus-maven plugin (in progress)
- New Ant tasks
- Bugs fixed
- New web-site
- New web-site skin

Cactus Ant build system

Cactus build system used to be made with Ant.

- Was really complicated: numerous of build scripts that invoke each other, downloads jars (maven-like style), variables implemented, external libraries used (ant-contrib), loops implemented, ...

Cactus Maven build system

- Cactus folder structure was refactored so that the new build system could be implemented
- Cactus provides different implementations for the different JVMs
- That way we had to produce multiple artifacts that shared the same base source
- This was solved by using the assembly plugin
- Demonstration

Cargo: what is this?

- Most of the complaints in the Cactus mail lists come from the fact that currently only a few containers are supported
- This defect is corrected with the integration between Cactus and Cargo
- Codehaus Cargo is thin wrapper around existing J2EE containers
- This project was started by Vincent Massol, again, and supplies a Java API to start/stop/configure existing java containers

Cactus-Cargo integration

- Integration between Cactus & Cargo affects only the Cactus Ant tasks.
- Cargo is used as an external library in the classpath of the Cactus project
- Cargo is used for the whole manipulation of the servers

New Ant tasks syntaxes

- Demonstration

Cactus-Maven2 plugin

- The ways to execute cactus tests is either through the Ant tasks or via Maven(1x, 2x)
- Felipe Leme wrote the Cactus Maven1 plugin that was included in the Cactus 1.7.2
- I started the Cactus-Maven2 plugin and have been working over it for the past months
- Only the cactifywar, cactifyear, webxmlmerge ready
- The actual test is still to come
- Problems I have encountered regard the unability to extend Maven MOJOs

Cactus-maven plugin example

- Demonstration

New site for Cactus

- After the build system moved to Maven the documentation had to be build from Maven
- RDF differences occured
- New site generation
- New skin for the site
- Demonstration

The future of Cactus

Two paths for the Jakarta Cactus Project

Pesimistic:

- Endangered to become one of the stalled projects at the ASF
- Lack of contributors and active develops

The future of Cactus

Optimistic:

- Finish the Cactus-Maven2 plugin
- Write thorough document
- Add support for Junit4
- Implement J2EE1.6 (annotations)
- Have a release of Cactus 2.0
- Move Cactus+JMeter to a TLP (testing.apache.org)

QUESTIONS?

Resources:

http://jakarta.apache.org/cactus

http://java-bg.net/svn/apachecon2007