Hanno User Guide 1.x

Introduction

Hanno is a test automation framework in Java that uses the model-based testing paradigm. It can be used to develop an automated testing tool for any web application.

Hanno implements a model-based test automation approach. To test a web application, an SCXML model is created to describe the application behavior. A Java class is created with methods for each event or state in the model. Each method calls Watij code to execute the event in Internet Explorer, or to verify that the browser is in the correct state. The Java class is run by an engine with a simple algorithm to determine which event to execute next. The order of test execution is not predetermined.

Capturing the application behavior in a model enables a complex application to be tested with a smaller amount of information that can be easier to maintain than a large test script library. The dynamic nature of the test execution can also find new bugs because new test sequences are executed on each run.

Hanno is an open source project released under GPL license. It is hosted on SourceForge.net at http://sourceforge.net/projects/hanno/.

System Requirements

Hanno runs on Windows with Internet Explorer. It was developed on Windows XP in Eclipse SDK 3.3.0. It requires the following:

- Java SE Development Kit 6 Update 4 (JDK 1.6.0_04)
- Internet Explorer 7
- Watij 3.2.1
- Apache Commons SCXML 0.6
- Junit 4.2
- Ant 1.7 (to run from build file).

Installation

- Download Hanno from the project site: http://sourceforge.net/projects/hanno/
- Unzip the Hanno_X_xx.zip file. Extract or copy the Hanno folder to your workspace.
- Import the project into your Java development environment. For Eclipse:
    - Select File -> Import...
    - Select Existing Projects into Workspace. Select Next.
    - Select Browse. Browse to the Hanno folder. Select OK. Select Finish.

Engine

The Hanno Engine consists of three Java classes:

- hanno.engine.HannoRunner Main application
- hanno.engine.HannoStateMachine Superclass for state machine classes
- hanno.engine.HannoListener Listens for events and calls event methods

Example project

Hanno includes an example project. It browses to several pages on the Wikipedia site. The source code for the example project is located under src/hava/hanno/examples/wikipedia. The project consists of two files:

- wikipedia.xml SCXML model for the Wikipedia site
- WikipediaStateMachine.java Java class with event and state methods for the model

The example project is intentionally simple and incomplete, and should be used for illustration purposes only. Hanno is not intended for testing public web sites, but for software test engineers to test their own projects.

Running the example project

There are several ways to run the example project:

- In Eclipse, select the HannoRunner class and Run As Java Application.
- In Eclipse, select the execute target of the build.xml file and Run As Ant Build.
- From command line, enter:
    - java hanno.engine.HannoRunner

To run from command line, your Classpath must be set correctly. See the build.xml file for a list of Classpath entries.

Command Line options

Hanno has the following command line options:

- -c <classname> run the specified class (default hanno.examples.wikipedia)
- -o <outputfile> output file events are logged to (default examples.wikipedia.log)
- -s <true or false> stop execution when all events have been covered (default false)
- -l <limit> stop execution after specified number of events (default 100)
- -i <inputfile> run from an event log file (default off)

If the -i option is set, execution is run from an event log file, rather than generating a new event sequence. The -o, -s. and -l options will be ignored if the -i option is set.

If the -s option is set true and -l is set, both options are used, so execution will stop after all events have been covered, or specified number of events have occurred, whichever comes first.

Output

The Commons SCXML engine that Hanno uses to run the state machine logs its output to System.err. In order for output to display properly, all other output is logged to System.err than System.out. When running in Eclipse, the last few lines of output would typically look like this:

```
INFO: Current States: [help_contents_state]
EVENT: wikipedia_community
Feb 18, 2008 9:35:52 AM org.apache.commons.scxml.SCXMLExecutor
logState
INFO: Current States: [wikipedia_community_state]
SUMMARY: Test steps 100
SUMMARY: Total events 7
SUMMARY: All events covered.
SUMMARY: End execution from state machine.
```

If execution stops prematurely, review the output as well as the log file. SCXML does not always stop executing on an Exception, so a failure at one point may not stop execution until sometime later.

Creating a new project

To create a new Hanno project, create a new folder under the src/java/hanno folder in the Hanno project folder. Copy the files from the src/java/hanno/examples/wikipedia folder and give them new names appropriate to the application you are testing.

Each project must include an XML file with an SCXML model of the application behavior you want to test, and a Java file with a subclass of HannoStateMachine that references the XML file in the same project folder. See the Example project.

Each state must have a unique name and a method with that name in the state machine class. Each transition must have an event. Each event must have an event method in the state machine class. An event may exist in more than one transition, but events may not have the same name as any state.

It is recommended to build the XML model and Java class together incrementally, starting with the model of a single behavior, debugging, fixing, then adding more behaviors to the model and class.

Origin

Hanno is named for Hanno the Navigator, a Carthaginian who explored the coast of west Africa about 450 B.C.E.

More information

Sourceforge Project web site: http://sourceforge.net/projects/hanno/

SCXML working draft: http://www.w3.org/TR/scxml/

Apache Commons SCXML: http://commons.apache.org/scxml/

Watij: http://www.watij.com

Original author's blog: http://opentesting.blogspot.com