Plugin Beanshell

Mikaël Marche

26 avril 2006

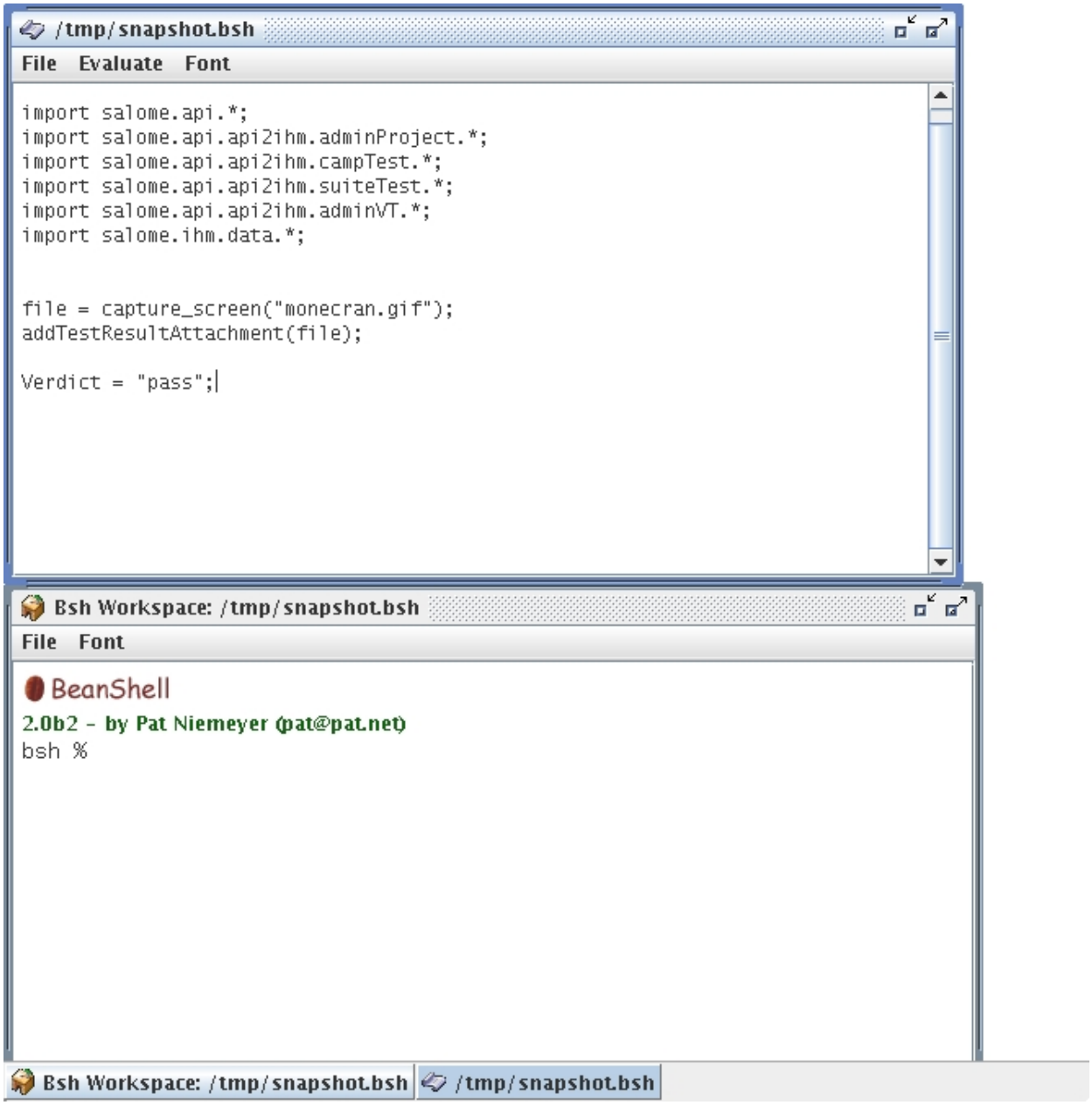
Beanshell, est un langage de script totalement intégré dans Java, qui a comme principale caractéristique la possibilité d'exécuter des objets Java compilés ou écrits en BeanShell. De plus, comme tout plugin de test et de script, les programmes BeanShell peuvent utiliser les références des objets manipulés dans Salomé.

Table des matières

1 Créer un test ou un script BeanShell

Pour créer un script BeanShell dans les environnements ou les exécutions, il faut sélectionner un fichier ń .bsh ż dans la fenêtre de sélection de fichier. Pour créer un test BeanShell, après avoir créé un test automatique de type BeanShellTester, il faut ajouter un script *.bsh* à partir du panel *Script* du test.

Chaque script BeanShell est un programme exécutable qui peut être édité dans le menu « Script » des tests, environnements, et exécutions.



2 Utiliser l'API de Salomé dans BeanShell

Par défaut, les scripts possèdent des références sur les variables suivantes (*=org.objectweb.salome_tmf.data) :

Nom	Classe	Description	Disponible
date	*Java.lang.Date*	Date de l'exécution	T, Env, Ex
time	*Java.lang.Time*	Heure de l'exécution	T, Env, Ex
salome_projectName	*Java.lang.String*	Nom du projet courant	T, Env, Ex
salome_projectObject	**.Project*	Référence de l'objet projet de Salomé	T, Env, Ex
salome_debug	*boolean*	Faut lors de l'évaluation du script dans l'éditeur	T, Env, Ex
salome_ExecResultObject	**.ExecutionResult*	Référence sus les résultats d'exécution courant	T, Ex
salome_ExecTestResultObject	**.ExecutionTestResult*	Référence sur le résultat d'exécution du test courant	T
salome_CampagneName	*Java.lang.String*	Nom de la campagne courante	T, Env, Ex
salome_CampagneObject	**.Campaign*	Référence de la campagne courante	T, Env, Ex
salome_environmentName	*Java.lang.String*	Nom de l'environnement courant	T, Env, Ex
salome_environmentObject	**.Environment*	Référence sur l'environnement courant	T, Env, Ex
salome_ExecName	*Java.lang.String*	Nom de l'éxécution courante	T, Env, Ex
salome_ExecObject	**.Execution*	Référence sur l'exécution courante	T, Env, Ex
salome_TestName	*Java.lang.String*	Nom du test courant	T
salome_TestObject	**.Test*	Référence sur le test courant	T
salome_SuiteTestName	*Java.lang.String*	Nom de la suite de tests courante	T
salome_SuiteTestObject	**.TestList*	Référence sur la suite de tests courante	T
salome_FamilyName	*Java.lang.String*	Nom de la famille courante	T
salome_FamilyObject	**.Family*	Référence sur la famille courante	T
testLog	*Java.lang.String*	Log de tests à ajouter comme attachement à l'éxécution	T
Verdict	*Java.lang.String*	Verdict du test (pass, fail, unconclusif)	T
salome_classloader	*Java.net.URLClassLoader*	Classe loader de BeanShell	T, Env, Ex

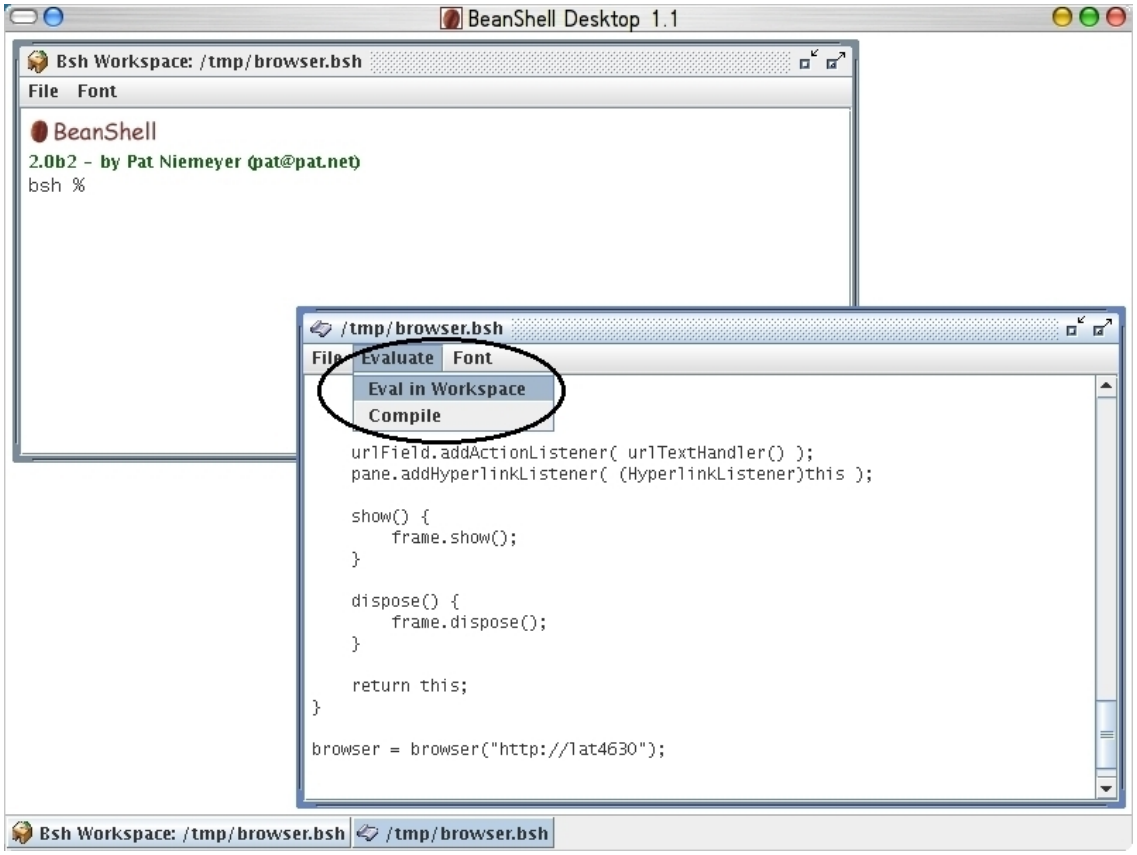
La colonne « disponible » décrit où peuvent être utilisées ces variables (T pour les tests, Env pour les Environnements, et Ex pour les Exécutions).

En plus de ces variables, les variables (avec leur valeur) définies dans un jeu de données ou un environnement sont accessibles dans les scripts, directement à partir de leur nom et sont typées *Java.lang.String*.

On note aussi qu'une Hashtable nommée « testargs » contenant l'ensemble des variables prédéfinies et les variables de test est accessible dans les scripts de test.

3 Compiler et évaluer un script

En Mode modification d'un script il est possible d'évaluer et de compiler la syntaxe d'un script.



Lors de l'évaluation de script les variables directement liées à l'exécution d'une campagne ne sont pas valuées (valeurs des jeux de données), et la variable *salome_debug* a pour valeur *true*.

4 Fonctions BeanShell prédéfinies

Le fichier *salome_testcmd.bsh* du répertoire *commands* du plugin BeanShell, contient un ensemble de fonctions beanshell utilisables dans l'ensemble des scripts de test, d'environnement, et d'exécution.

```
import org.objectweb.salome_tmf.api.*;
import org.objectweb.salome_tmf.api.api2ihm.adminProject.*;
import org.objectweb.salome_tmf.api.api2ihm.campTest.*;
import org.objectweb.salome_tmf.api.api2ihm.suiteTest.*;
import org.objectweb.salome_tmf.api.api2ihm.adminVT.*;
import org.objectweb.salome_tmf.data.*;
import salomeTMF_plug.beanshell.ParentClassLoader;

import java.awt.image.BufferedImage;
import java.io.File;
import java.net.URL;
```

```
/* Ajout d'un path (String ou URL) au classloader */
addJar( path ) {
        URL url;
        if ( path instanceof URL ) {
                url = path;
        } else {
            try  {
                url = new URL(path);
            } catch (e){
                url = pathToFile( path ).toURL();
            }
        }
        salome_classloder.AddJar( url );
}

/* Attente d'un temps t*/
waittime(int t){
    Thread.sleep(t);
}

/* capture une copie d'écran et l'enregistre dans out_file (File) */
File capture_screen(String out_file) {
    BufferedImage image;
    Thread.sleep(5);
    Toolkit tk = Toolkit.getDefaultToolkit();
    tk.sync();
    Rectangle ecran = new Rectangle(tk.getScreenSize());
    Robot robot = new Robot();
    robot.setAutoDelay(0);
    robot.setAutoWaitForIdle(false);
    image = robot.createScreenCapture(ecran);
    file = new File(out_file);
    javax.imageio.ImageIO.write(image, "png", file);
    return file;
}

/* Ajout d'un fichier en attachement du résultat d'exécution de test courant */
addTestResultAttachment (File f){
    if (salome_debug == false) {
        salome_ExecTestResultObject.addAttchment(f);
    }
}

/* Ajout d'un fichier en attachement à l'exécution de campagne courante */
addExecResultAttachment (File f){
    if (salome_debug == false) {
        salome_ExecResultObject.addAttchment(f);
    }
}

/* Ajout d'un fichier en attachement à l'exécution de campagne courante */
addTestAttachment(File f){
    if (salome_debug == false) {
        salome_TestObject.addAttchment(f);
    }
}
```

```
/* Ajout d'un fichier en attachement à la suite test courante */
addSuiteAttachment(File f){
    if (salome_debug == false) {
        salome_SuiteTestObject.addAttchment(f);
    }
}


addTestResultAttachmentUrl (URL u){
    if (salome_debug == false) {
        salome_ExecTestResultObject.addAttchment(u);
    }
}

addExecResultAttachmentUrl (URL u){
    if (salome_debug == false) {
        salome_ExecResultObject.addAttchment(u);
    }
}

addTestAttachmentUrl(URL u){
    if (salome_debug == false) {
        salome_TestObject.addAttchment(u);
    }
}

addSuiteAttachmentUrl(URL u){
    if (salome_debug == false) {
        salome_SuiteTestObject.addAttchment(u);
    }
}

addEnvAttachmentUrl(URL u){
    if (salome_debug == false) {
        salome_environmentObject.addAttchment(u);
    }
}

addTestResultAttachmentText (String fileName, String text){
    if (salome_debug == false) {
        salome_ExecTestResultObject.addAttchment(fileName, text);
    }
}

addExecResultAttachmentText (String fileName, String text){
    if (salome_debug == false) {
        salome_ExecResultObject.addAttchment(fileName, text);
    }
}

addTestAttachmentText(String fileName, String text){
    if (salome_debug == false) {
        salome_TestObject.addAttchment(fileName, text);
    }
}

addSuiteAttachmentText(String fileName, String text){
    if (salome_debug == false) {
```

```
        salome_SuiteTestObject.addAttchment(fileName, text);
    }
}

addEnvAttachmentText(String fileName, String text){
    if (salome_debug == false) {
        salome_environmentObject.addAttchment(fileName, text);
    }
}
```

Ce fichier peut être enrichi par l'administrateur de Salomé.

5 Exemple

L'exemple suivant présente l'utilisation du plugin BeanShell pour les scripts d'environnement, d'exécutions, et de tests. Une des particularités de cet exemple est qu'il montre que le contexte d'exécution des scripts est préservé lors de l'exécution d'une campagne. De plus, il montre comment utiliser les variables de tests ainsi que les fonctions prédéfinies.

L'exemple est constitué de quatre scripts :

- un script d'environnement déclarant une fenêtre de browser Web, associé à un environnement déclarant une variable *url_test* ;
- un script de pré-exécution affichant la fenêtre du browser déclarée dans l'environnement ;
- un script de test utilisant le browser de l'environnement et la variable *url_test* ;
- un script de post-exécution fermant la fenêtre du browser de l'environnement.

Le script d'environnement est le suivant :

```
import javax.swing.*;
import javax.swing.event.*;
import javax.swing.text.*;
import java.awt.event.*;
import java.awt.*;

browser( startingUrl ) {

        invoke( method, args ) {}

        windowClosing(WindowEvent we) {
                we.getWindow().setVisible(false);
        }

        hyperlinkUpdate( HyperlinkEvent he ) {
                type = he.getEventType();
                if (type == HyperlinkEvent.EventType.ENTERED) {
                        pane.setCursor(
                                Cursor.getPredefinedCursor( Cursor.HAND_CURSOR) );
                        statusBar.setText(he.getURL().toString());
                } else
                        if (type == HyperlinkEvent.EventType.EXITED) {
                                pane.setCursor( Cursor.getDefaultCursor() );
                                statusBar.setText(" ");
                        } else {
                                setPage( he.getURL() );
                                if (urlField != null)
                                        urlField.setText(he.getURL().toString());
                        }
        }
```

```
        // This is the equivalent of an inner class in bsh.
        urlTextHandler() {
                actionPerformed(ActionEvent ae) {
                setPage( ae.getActionCommand() );
                }
                return this;
        }

        setPage( url ) {
                try {
                pane.setPage( url );
                } catch(Exception e) {
                        statusBar.setText("Error opening page: "+url);
                }
        }

        show() {
                frame.show();
        }

        dispose() {
                frame.dispose();
        }

        frame = new JFrame("Browser");
        frame.setSize(800,600);
        frame.addWindowListener( this );
        urlPanel = new JPanel();
        urlPanel.setLayout(new BorderLayout());
        urlField = new JTextField(startingUrl);
        urlPanel.add(new JLabel("Site: "), BorderLayout.WEST);
        urlPanel.add(urlField, BorderLayout.CENTER);
        statusBar = new JLabel(" ");
        pane = new JEditorPane();
        pane.setEditable(false);
        setPage( startingUrl );
        jsp = new JScrollPane(pane);
        frame.getContentPane().add(jsp, BorderLayout.CENTER);
        frame.getContentPane().add(urlPanel, BorderLayout.SOUTH);
        frame.getContentPane().add(statusBar, BorderLayout.NORTH);

        urlField.addActionListener( urlTextHandler() );
        pane.addHyperlinkListener( (HyperlinkListener)this );
        return this;
        }
browser = browser("http://www.beanshell.org");
```

On note l'existence des méthodes *setPage, show* et *dispose*, qui seront utilisées respectivement dans le script de tests, et les scripts d'exécution.

Le script de pré exécution est le suivant :

```
browser.show();
```

le script de post-exécution est le suivant :

```
browser.dispose();
```

Finalement le script de test est le suivant :

```
waittime(5000);
/* Capture de l'écran courant */
file = capture_screen("before.gif");

/* Ajouter la capture en attachement du résultat d'exécution de test */
addTestResultAttachment(file);

/* Demander au browser d'afficher l'url valuée par le paramètre d'environnement "url_test" */
browser.setPage(url_test);

waittime(5000);

/* Capture de l'écran courant */
file = capture_screen("after.gif");

/* Ajouter la capture en attachement du résultat d'exécution de test */
addTestResultAttachment(file);

/* Positionner le verdict de test à PASS */
Verdict = "pass";

/* Ajouter un attachement contenant ń OK ż en du résultat d'exécution de test */
testLog = "OK";
```

Le résultat de l'exécution de test produit en attachement trois fichiers, les deux captures d'écran *before.gif* et *after.gif*, ainsi que le log d'exécution de tests *log_ exec_ Goto.txt* contenant *OK*.

Lors d'une exécution, avec *url_ test = http ://www.pagesjaunes.fr* comme valeur du paramètre d'environnement, les captures d'écran sont les suivantes :