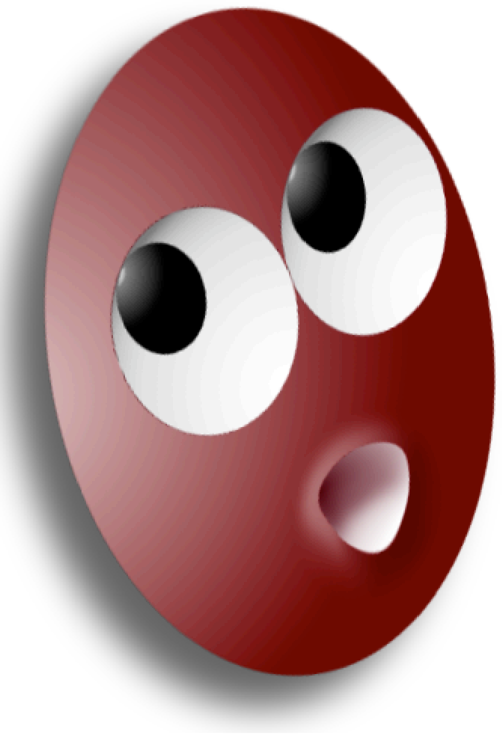
BeanShell

2.0



Why Scripting?

- The right tool for the right job.
- Productivity
- Accessibility
- Dynamic Applications

What is BeanShell

- Java Interpreter
- Scripting Language
- Small
- Embeddable / Extensible
- A *natural* scripting language for Java

What is it good for?

Development (People)

- Experimentation
- Prototyping
- Debugging
- Unit Testing
- Teaching

What is it good for?

Application Use

- Embedded Evaluation
  - Configuration, startup files
  - *Every configuration file eventally becomes a programming language...* -- James Gosling
  - Extension Language
  - Dynamic Evaluation of expressions

BeanShell About

- JDE for Emacs
- NetBeans / Forte
- Weblogic Application Server
- Various Unix distros (FreeBSD?)
- Lots of apps:
  - jEdit editor
  - jAlbum
  - Supported by Ant

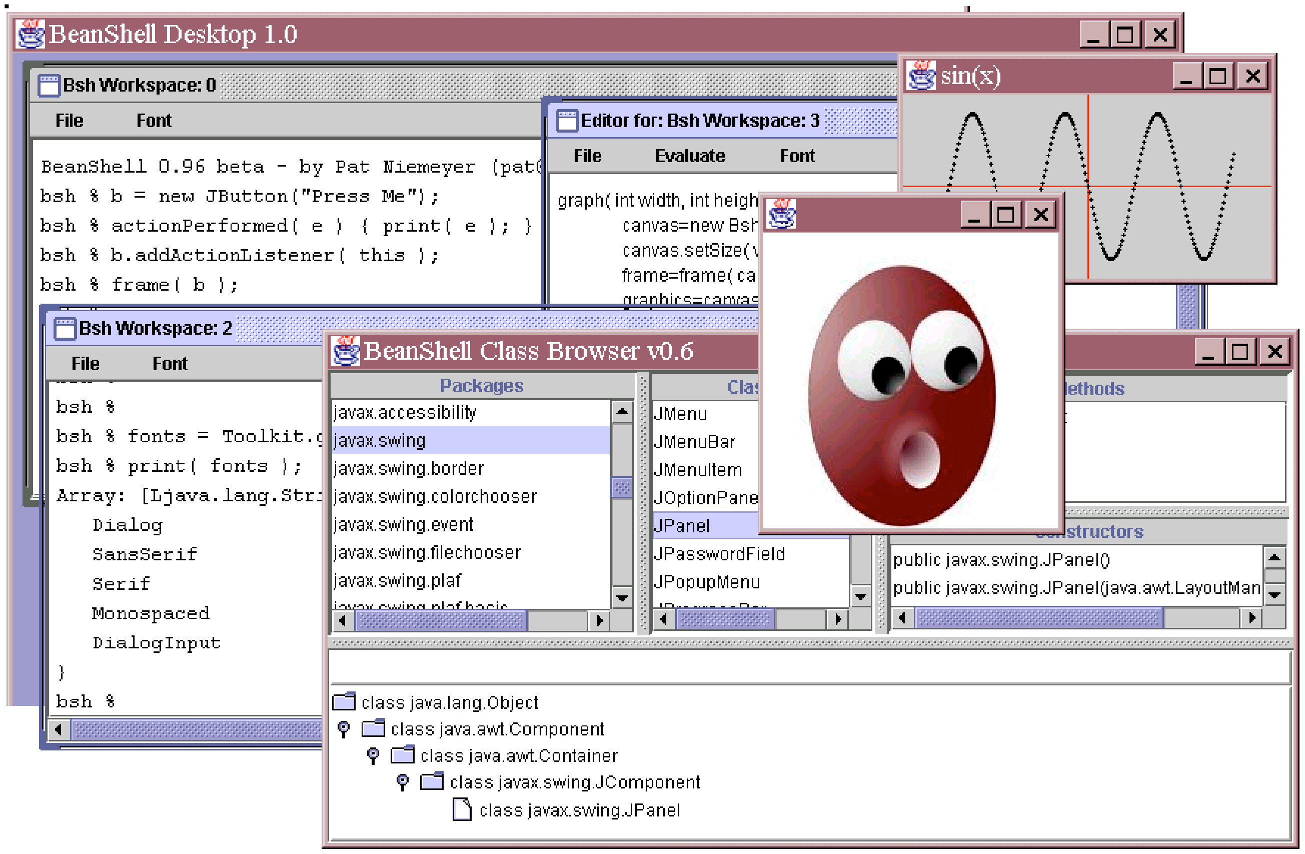
BeanShell Overview

- Getting Started
- Java Syntax
- Scripting Syntax
- Commands
- Modes of Operation
- Special Features

Getting Started

- Grab bsh-2.0b1.jar from www.beanshell.org
- Launch the jar to try out the desktop GUI.
- Add to classpath for regular use.

BeanShell Desktop



```
BeanShell 0.96 beta - by Pat Niemeyer (pat
bsh % b = new JButton("Press Me");
bsh % actionPerformed( e ) { print( e ); }
bsh % b.addActionListener( this );
bsh % frame( b );
```

```
graph( int width, int heigh
        canvas=new Bsh
        canvas.setSize(
        frame=frame( ca
        graphics=canvas
```

```
bsh %
bsh % fonts = Toolkit.g
bsh % print( fonts );
Array: [Ljava.lang.Str
    Dialog
    SansSerif
    Serif
    Monospaced
    DialogInput
}
bsh %
```

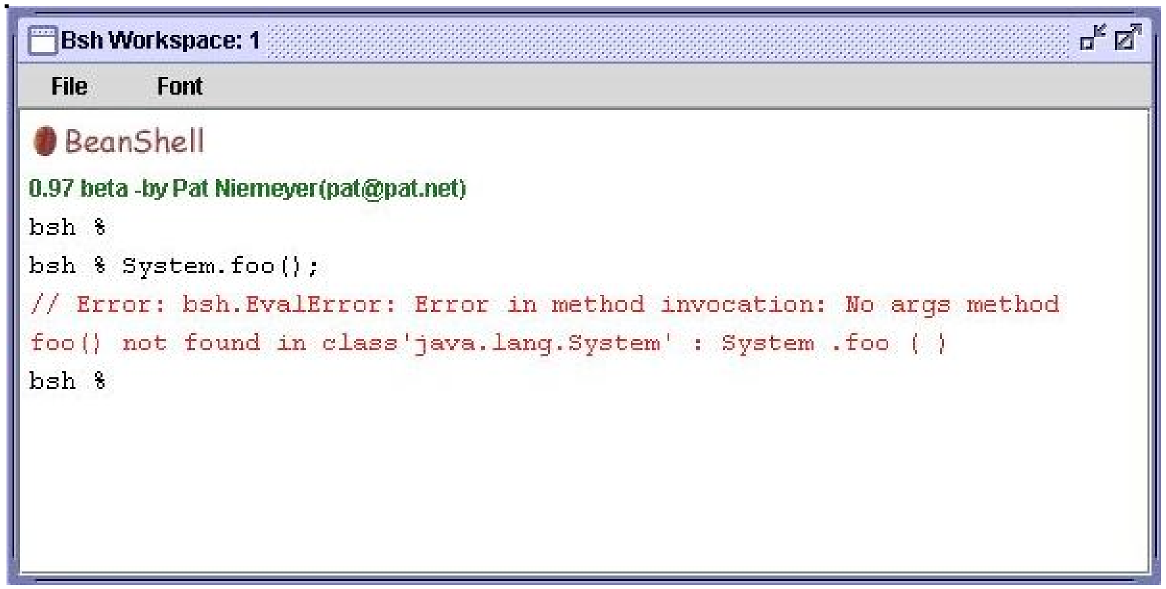
javax.accessibility
javax.swing
javax.swing.border
javax.swing.colorchooser
javax.swing.event
javax.swing.filechooser
javax.swing.plaf
javax.swing.plaf.basic

JMenu
JMenuBar
JMenuItem
JOptionPane
JPanel
JPasswordField
JPopupMenu

public javax.swing.JPanel()
public javax.swing.JPanel(java.awt.LayoutMan

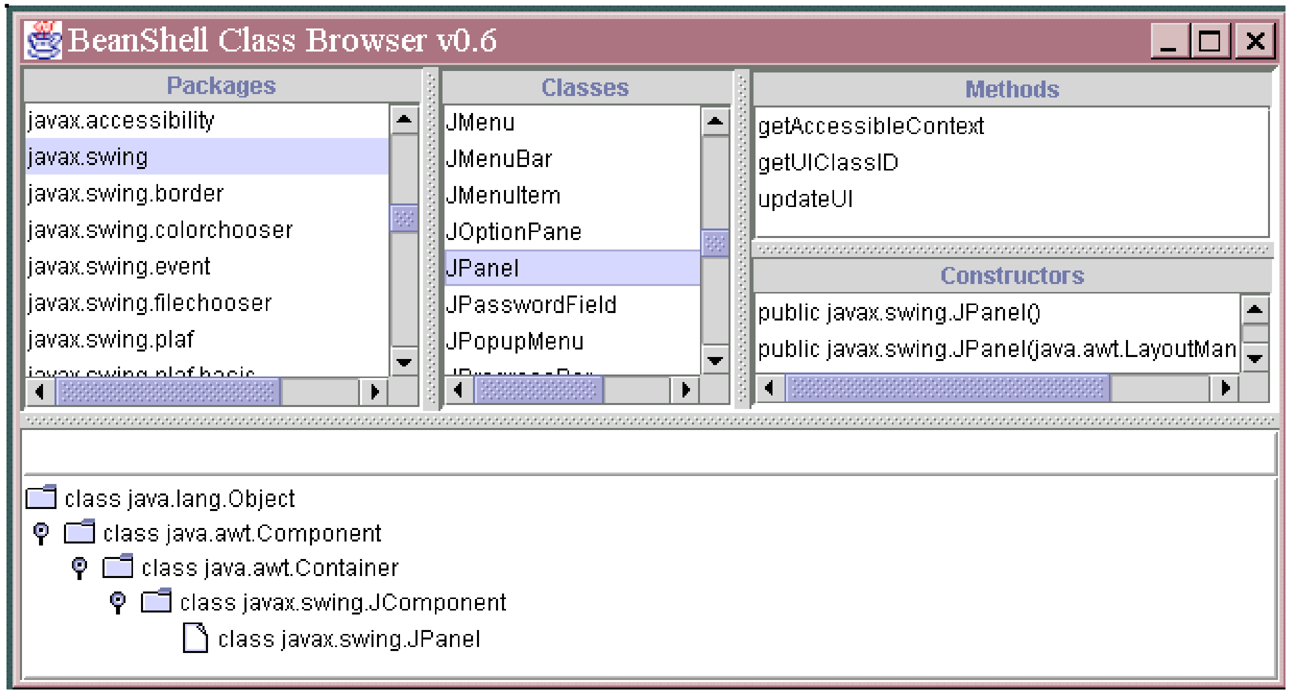
class java.lang.Object
class java.awt.Component
class java.awt.Container
class javax.swing.JComponent
class javax.swing.JPanel

JConsole



```
0.97 beta -by Pat Niemeyer(pat@pat.net)
bsh %
bsh % System.foo();
// Error: bsh.EvalError: Error in method invocation: No args method
foo() not found in class'java.lang.System' : System .foo ( )
bsh %
```

BeanShell Desktop Class Browser



Packages

javax.accessibility
javax.swing
javax.swing.border
javax.swing.colorchooser
javax.swing.event
javax.swing.filechooser
javax.swing.plaf

Classes

JMenu
JMenuBar
JMenuItem
JOptionPane
JPanel
JPasswordField
JPopupMenu

Methods

getAccessibleContext
getUIClassID
updateUI

Constructors

public javax.swing.JPanel()
public javax.swing.JPanel(java.awt.LayoutMan

```
class java.lang.Object
  class java.awt.Component
    class java.awt.Container
      class javax.swing.JComponent
        class javax.swing.JPanel
```

Running the Interpreter

```
// Run the GUI desktop
java bsh.Console

// Run as text-only on the command line
java bsh.Interpreter

// Run script file from command line
java bsh.Interpreter filename [ args ]

// Run script in remote server
java bsh.Remote URL filename
```

Basic Syntax

- Plain Java
- Loose Java
- Scripted Intefaces and Objects
- Convenience Syntax

```
// Use a hashtable
Hashtable hashtable = new Hashtable();
Date date = new Date();
hashtable.put( "today", date );

// Print the current clock value
print( System.currentTimeMillis() );

// Loop
for (int i=0; i<5; i++)
    print(i);

// Pop up a frame with a button in it
JButton button = new JButton( "My Button" );
JFrame frame = new JFrame( "My Frame" );
frame.getContentPane().add( button, "Center" );
frame.pack();
frame.setVisible(true);
```

```
// Use a hashtable
hashtable = new Hashtable();
date = new Date();
hashtable.put( "today", date );

// Print the current clock value
print( System.currentTimeMillis() );

// Loop
for (i=0; i<5; i++)
    print(i);

// Pop up a frame with a button in it
button = new JButton( "My Button" );
frame = new JFrame( "My Frame" );
frame.getContentPane().add( button, "Center" );
frame.pack();
frame.setVisible(true);
```

Scripted Methods

```
int addTwoNumbers( int a, int b ) {
    return a + b;
}
sum = addTwoNumbers( 5, 7 );

add( a, b ) {
    return a + b;
}

foo = add(1, 2);
print( foo ); // 3

foo = add("Oh", " baby");
print( foo ); // Oh baby
```

Simple Scripted Objects (closures)

```
foo() {
    int bar = 42;
    return this;
}

fooObject = foo();
print( fooObject.bar ); // prints 42!
```

Implementing Interfaces Anon. Inner Class Style

```
buttonHandler = new ActionListener() {
    actionPerformed( event ) {
        print(event);
    }
};

button = new JButton();
button.addActionListener( buttonHandler );
frame(button);
```

Implementing Interfaces via ‘this’ references

```
actionPerformed( event ) {
    print( event );
}

button = new JButton("Foo!");
button.addActionListener( this );
frame( button );
```

The `invoke()` meta-method.

```
mouseHandler = new MouseListener() {
    mousePressed( event ) {
        print("mouse button pressed");
    }

    invoke( method, args ) {
        print("Undefined method of MouseListener:"
            + name +", with args: "+args
        );
    }
};
```

Interface Example

```
import javax.xml.parsers.*;
import org.xml.sax.InputSource;

factory = SAXParserFactory.newInstance();
saxParser = factory.newSAXParser();
parser = saxParser.getXMLReader();
parser.setContentHandler( this );

invoke( name, args ) {
    print( name );
}

parser.parse( new InputSource(bsh.args[0]) );
```

Convenience Syntax
Property Access

```
button = new java.awt.Button();
// Equivalent to: b.setLabel("my button");
button.label = "my button";
// Equivalent to print( b.getLabel() );
print( button.label );

Float f = new Float(42f);
// Equivalent to print( f.isInfinite() );
print( f.infinite );
```

Convenience Syntax
Property and Map Access

```
b = new java.awt.Button();
// Equivalent to: b.setLabel("my button");
b{"label"} = "my button";

h = new Hashtable();
// Equivalent to: h.put("foo", "bar");
h{"foo"} = "bar";
```

Auto-Boxing and Un-Boxing

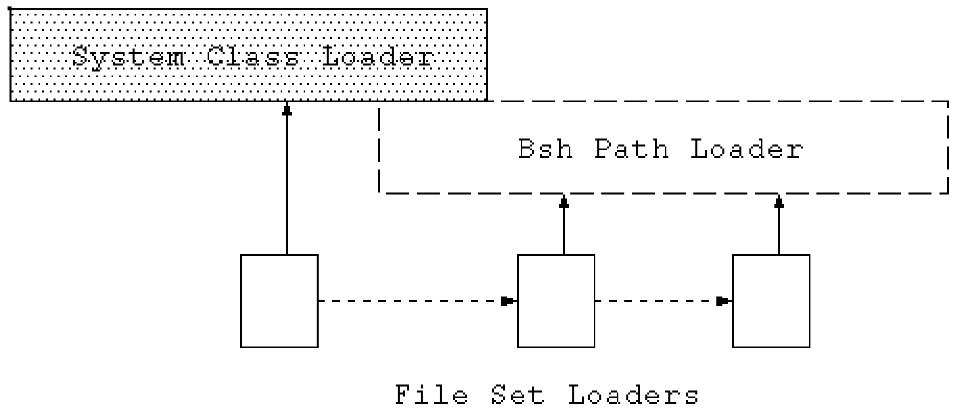
```
int i=5;
Integer iw = new Integer(5);
print( i * iw );  // 25

Vector v = new Vector();
v.put(1);
int x = v.getFirstElement();
```

BeanShell Commands

- **`source()`,`run()`** - Read a bsh script into the interpreter, or run it in a new interpreter.
- **`frame()`** - Display a GUI component in a Frame or JFrame.
- **`load()`,`save()`** - Load or save serializable objects tofile.
- **`cd()`,`cat()`,`dir()`,`pwd()`**, etc. - Unix-like shell commands
- **`exec()`** - Run a native application
- **`javap()`** - Print the methods and fields of an object, similar to the output of the Java javap command.
- **`setAccessibility()`** - Turn on unrestricted access to private and protected components.

ClassPath Management

- ClassPath Modification
- Reloading Classes



ClassPath Modification

```
// Add to ClassPath
addClassPath( "/home/pat/java/classes" );
addClassPath( "/home/pat/java/mystuff.jar" );

// URLs work too...
addClassPath( new URL("http://myserver/beans.jar"));

// Change classpath
setClassPath( URL [] );
```

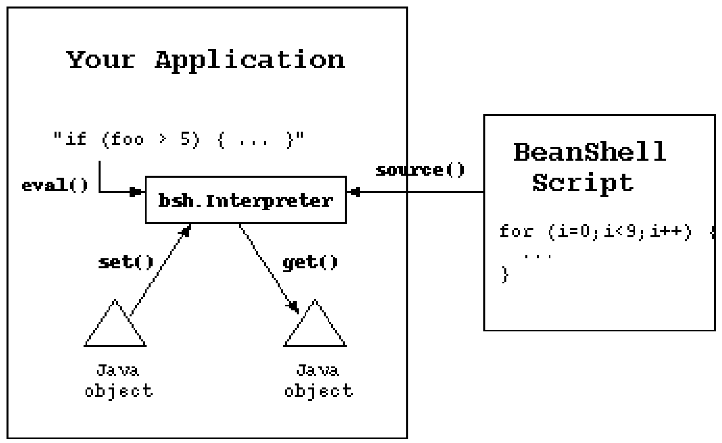
Reloading Classes

```
// Reload all user classes
reloadClasses();

// Reload a package
reloadClasses("mypackage.*");

// Reload an individual class
reloadClasses(”mypackage.MyClass”);
```

Modes of Operation

- Standalone scripts
- Embedded in application
- Remote server mode
- Servlet mode
- Applet mode

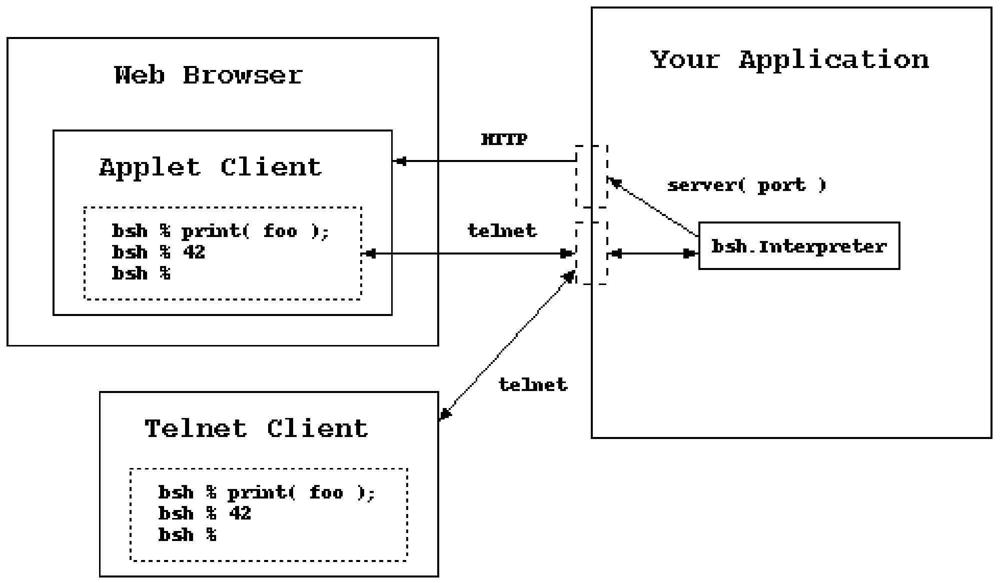
Embedded Mode



Calling BeanShell from Java

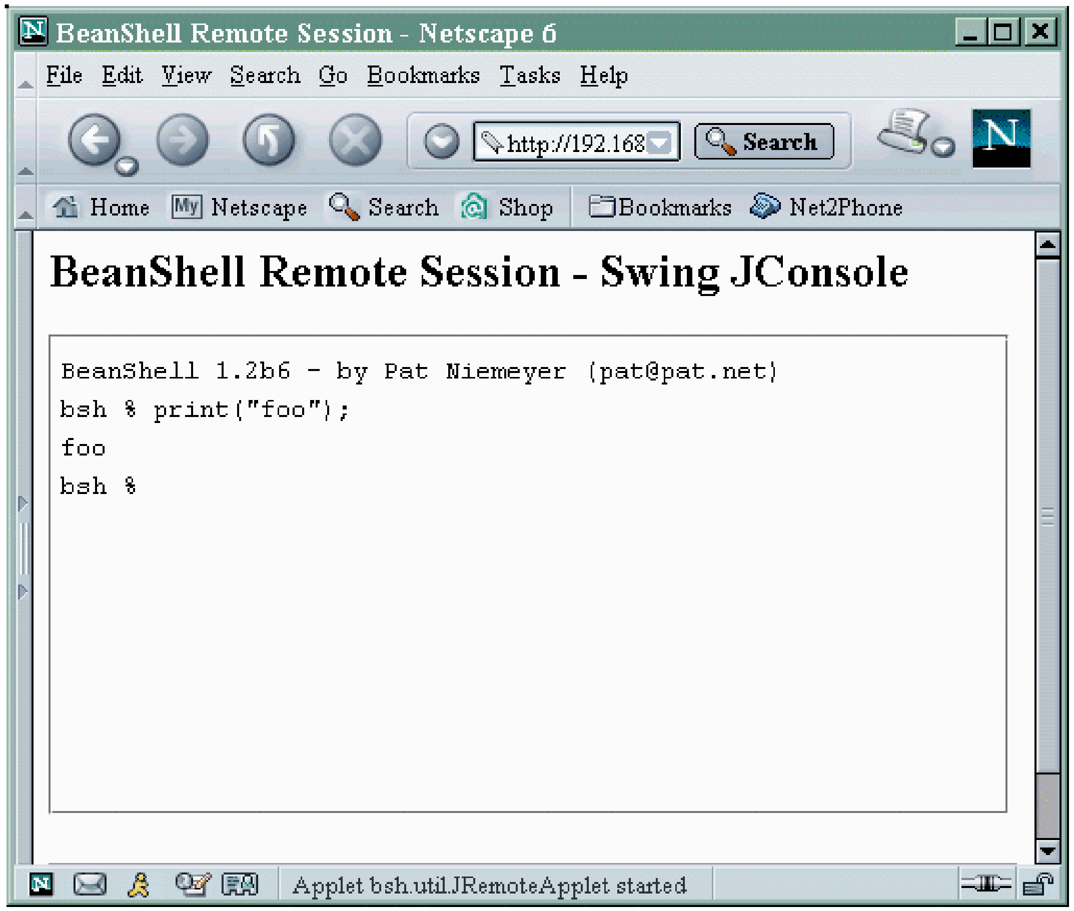
```
// Construct an interpreter
Interpreter i = new Interpreter();
// Set variables
i.set("foo", 5);
i.set("date", new Date() );

// retrieve a variable
Date date = (Date)i.get("date");

// Eval a statement and get the result
i.eval("bar = foo*10");
System.out.println( i.get("bar") );

// Source an external script file
i.source("somefile.bsh");
```

Remove Server Mode



```
bsh % print( foo );
bsh % 42
bsh %
```

```
server(1234);
// Httpd started on port: 1234
// Sessiond started on port: 1235
```

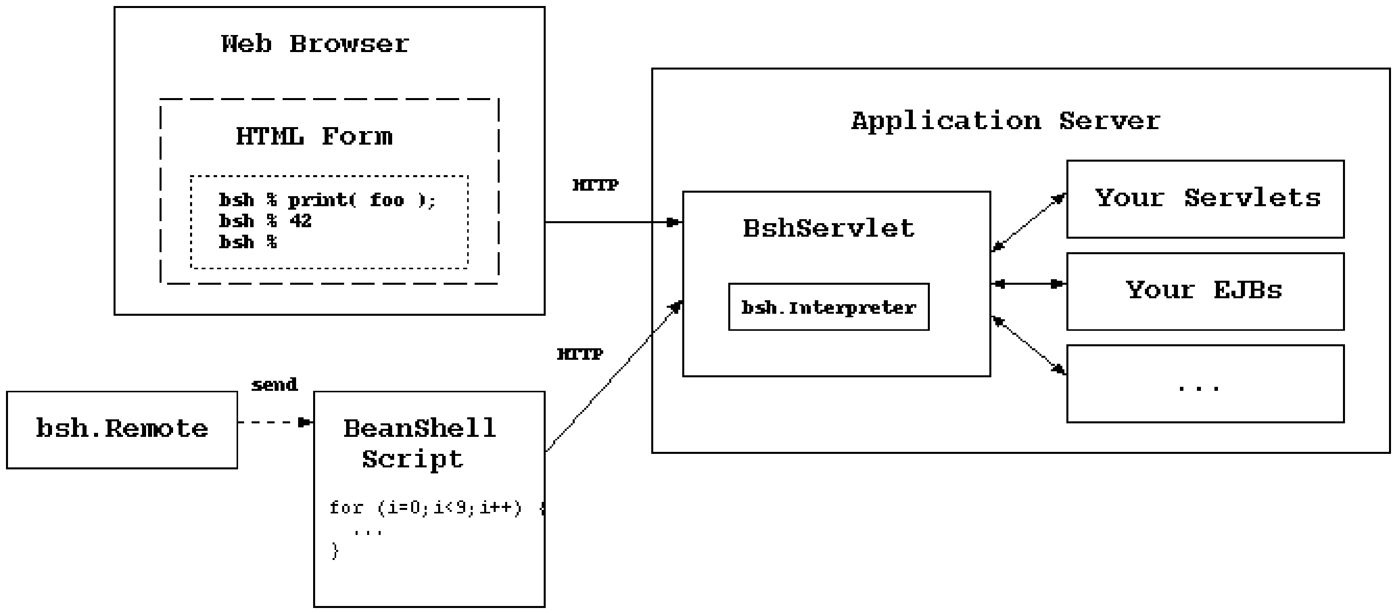

BeanShell Remote Session - Netscape 6
File Edit View Search Go Bookmarks Tasks Help
http://192.168
Search
Home
My Netscape
Search
Shop
Bookmarks
Net2Phone

BeanShell Remote Session - Swing JConsole

```
BeanShell 1.2b6 - by Pat Niemeyer (pat@pat.net)
bsh % print("foo");
foo
bsh %
```

Applet bsh.util.JRemoteApplet started

Servlet Mode



Servlet Mode web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE web-app
    PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
    "http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app>
    <servlet>
        <servlet-name>bshservlet</servlet-name>
        <servlet-class>bsh.servlet.BshServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>bshservlet</servlet-name>
        <url-pattern>/eval</url-pattern>
    </servlet-mapping>

</web-app>
```

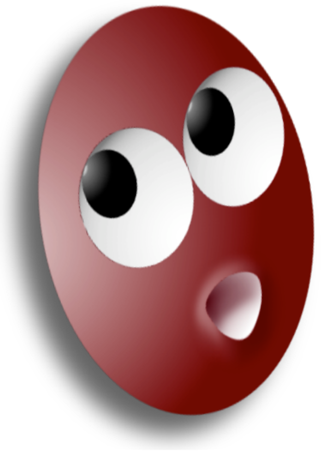
Servlet Mode

```
http://localhost/bshservlet/eval

java bsh.Remote
   http://localhost/bshservlet/eval test1.bsh
```

Wrap up Overview

- Questions?
- On to 2.0

New Features for 2.0



- Performance
- Error Reporting
- Full Java Compatability
- Extensibility / Convenience Syntax
- New Language Features
-

Performance Improvements

- JavaCC 3.0 based Parser faster and 30% smaller. Many grammar optimizations.
- Caching of method resolution for performance. (50% speed improvement in some cases).

Better Error Reporting

- All script error messages should now include line numbers and invocation text.
- Error messages now include a script stack trace (e.g. method a() called method b(), etc.)

Java Compatability

- Java style scoping everywhere

```
x=42;

foo() {
   x=43;
}

incrementX(){
   x=x+1;
}
```

Java Compatability

- All Java modifiers supported (e.g. **`public`**, **`private`**, **`static`**, **`abstract`**, etc.)
- The **`synchronized`** modifier is now implemented for methods and synchronized blocks.
- The **`throws`** clause on methods is now supported.

Java Compatability

- Java 1.5 style enhanced for loop.
  - JDK 1.1+ (no collections): Enumeration, arrays, Vector, String, StringBuffer
  - JDK 1.2+ (w/collections): Collections, Iterator

```
array = new int [] { 1, 2, 3 };

for( i : array )
  print( i );
```

Static Imports

- Java 1.5 style static imports

```
static import java.lang.Math.*;
sqrt(4.0);
```

Mix-ins

- Instance Object imports (Mix-ins) with **`importObject()`**

```
Map map = new HashMap();
importObject( map );
put("foo", "bar");
print( get("foo") ); // "bar"
```

User Command Path

- User Command Path - Scripts and compiled commands may be imported into namespace with **`importCommands()`**.

```
// equivalent
importCommands("/bsh/commands")
importCommands("bsh.commands")

// Classpath modifications obeyed
addClassPath("mycommands.jar");
importCommands("/mypackage/commands");
```

Convenience Syntax

-
- **`switch`** statements now work with all types.
- Maps can now be used with the hashtable style accessor syntax.

```
        map{"foo"} = "bar":
```

-
- Field style object property access now supports isFoo() style getters

Properties Style Auto-Allocation of Variables

```
// foo is initially undefined
foo.bar.gee = 42;

print( foo.bar.gee ); // 42

print( foo.bar ); //'this' reference (XThis)
to Bsh object: auto: bar

print( foo ); // 'this' reference (XThis) to
Bsh object: auto: foo
```

Props Example 1

```
// Home directory
myApp.homeDir="/pkg/myApp";

// User Info
myApp.user.defaultUser="Bob";
myApp.color.fgcolor = "Aqua";

/* Complex properties */
myApp.color.bgcolor = Color.BLUE;  // Real enumerations for free!
myApp.color.colorset =
        new Color [] { Color.RED, Color.GREEN, Color.BLUE };

// Script application behavior
onStartup() {
    print( "Hello User: " + USERNAME );
}

// Include more config and scripts via source(), eval(), etc.
source( System.getProperty("user.home") + "/" + ".myAppConfig" );

// Configure with real objects, with real arguments
SocketFactory.setDefaultSocketFactory( new MySocketFactory(42) );
```

```
// Create interpreter and read myprops.bsh
Interpreter config = new Interpreter();
config.source("myprops1.bsh");

// Read simple string properties
String homeDir = (String)config.get("myApp.homeDir");
String defaultUser = (String)config.get("myApp.user.defaultUser");

// Read true object properties
Color fgcolor = (Color)config.get("myApp.color.bgcolor");

// True nested properties (not yet as pretty as it could be)
NameSpace myAppColor = ((This)config.get("myApp.color")).getNameSpace();
// Iterate over myApp.color nested properties
String [] varNames = myAppColor.getVariableNames();
//for( String name : varNames ) { }

// Set the USERNAME variable for the script's use
config.set("USERNAME", "Pat");

// Execute user's scripted onStartup() method, if it exists
config.eval("onStartup()");
```

Properties Example 2

```
public class MyApp2
{
    // JavaBean accessor methods
    public void setHomeDir( String homeDir ) { ... }
    public User getUser() { return new User(); }

    void readProps() throws IOException, EvalError
    {
        // Create interpreter and read myprops.bsh
        Interpreter config = new Interpreter();
        // Set this object as "myApp"
        config.set("myApp", this);
        config.source("myprops2.bsh");

        // No property fetching code necessary!
        // Execute behavior...
    }
```

Full Java Syntax Support

- Interpreted classes with real Java types
- Extend / Implement arbitrary Java classes
- Load classes from .java source files

Scripted Classes

- Expose all typed methods and variables of the class.
- Bound in the namespace in which they are declared.
- May freely mix loose / script syntax with full Java class syntax.

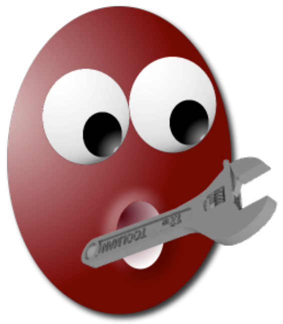
Scripted Class Example

```
// MyScript.bsh
count = 5;

class HelloWorld extends Thread {
   public void run() {
      for(i=0; i<count; i++)
         print("Hello World!");
   }
}

new HelloWorld().start();
```

Scripted Class Implementation

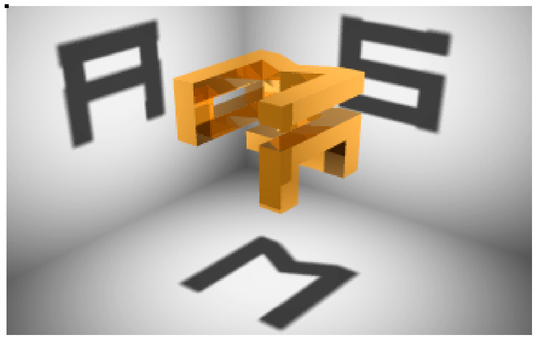
- Light weight bytecode generator
- Stub classes delegate method and constructor calls to interpreter
- Generated accessor methods for superclass visibility

Current Limitations



- Reflective Access Permissions
- ClassLoader

ASM Bytecode Library



- www.objectweb.org
- Visitor pattern for reading and writing classes
- Very light weight, easy to use
- Only 22k!
- DumpClassVisitor writes code for you!

Wrapup

- Script Documentation
- License
- The Open Source Experience
-

BshDoc JavaDoc Style Documentation

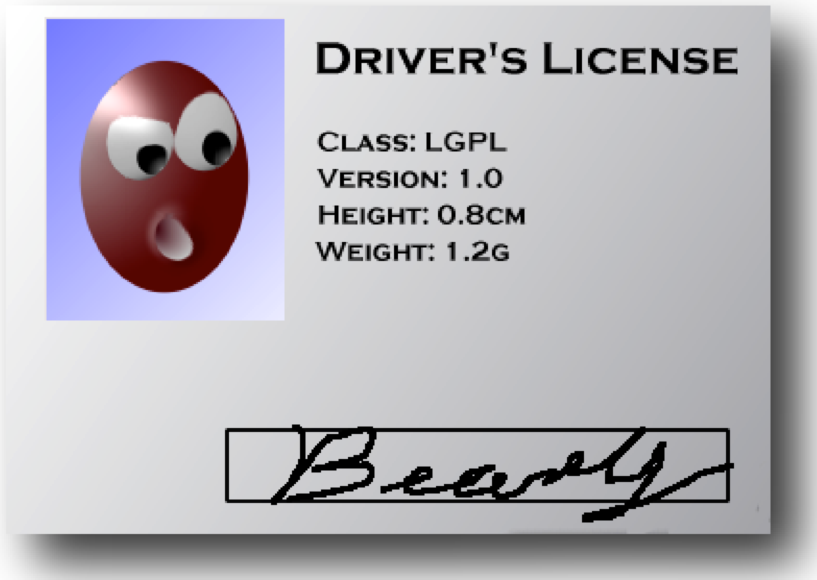
```
/**
    BshDoc for the foo() command.
    Explicitly supply the signature to be displayed for the foo()
    method.

    @method foo( int | Integer ) and other text...
*/
foo( arg ) { ... }
```

```
java bsh.Interpreter
    bshdoc.bsh myfile.bsh [ myfile2.bsh ] [ ... ] > output.xml
```

BshDoc XML Output

```
<!-- This file was auto-generated by the bshdoc.bsh script -->
<BshDoc>
  <File>
    <Name>foo</Name>
    <Method>
      <Name>doFoo</Name>
      <Sig>doFoo ( int x )</Sig>
      <Comment>
        <Text>&lt;![CDATA[ doFoo() method comment. ]]&gt;</Text>
        <Tags>
        </Tags>
      </Comment>
    </Method>
    <Comment>
        <Text>&lt;![CDATA[ foo file comment. ]]&gt;</Text>
        <Tags>
        </Tags>
    </Comment>
  </File>
</BshDoc>
```

LGPL / SPL License



DRIVER'S LICENSE

CLASS: LGPL
VERSION: 1.0
HEIGHT: 0.8CM
WEIGHT: 1.2G

Bearly

Open Source Project Experiences

- More eyes and hands are great, but...
- Constant refactoring.
- Interesting people.

Favorite Pair of Quotes...

```
"... it's been a long time since I've seen a
non-commercial project that is so well-documented"

  -- Robert F Schmitt.
```

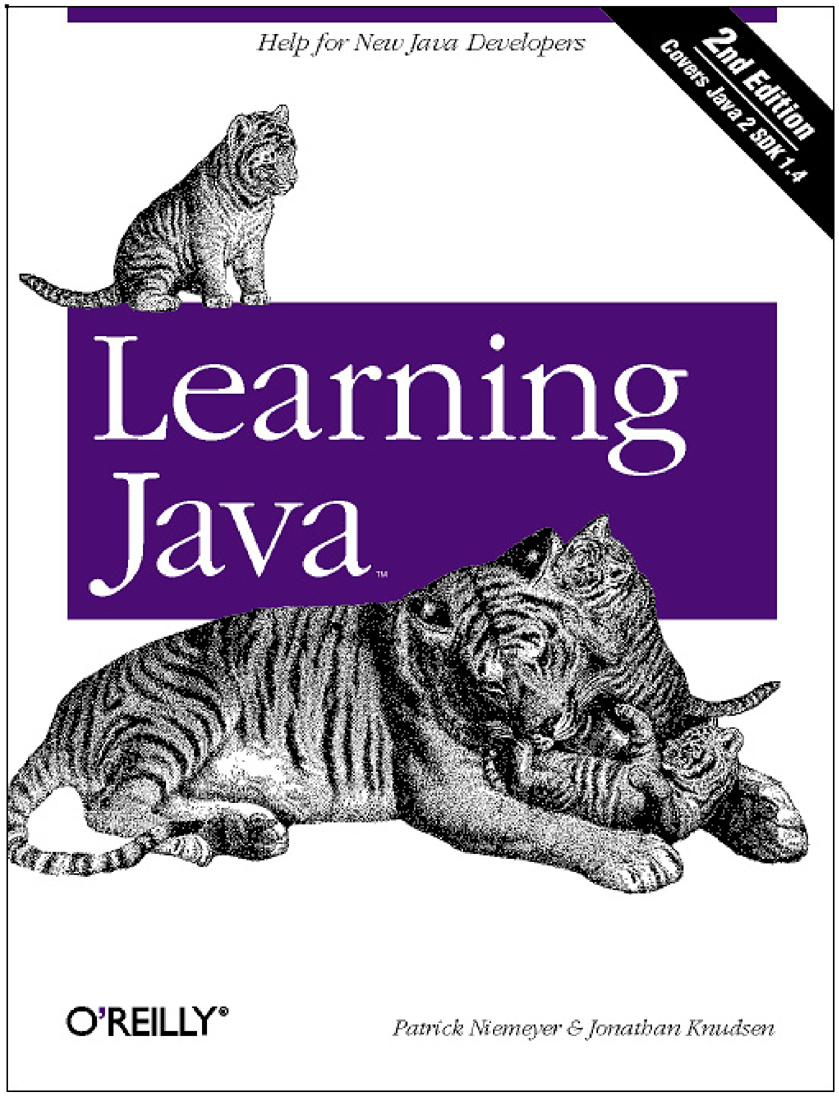
Favorite Pair of Quotes...

"... it's been a long time since I've seen a non-commercial project that is so well-documented"

-- Robert F Schmitt.

"In going through your tutorial I have found a few spelling errors and poorly constructed sentences. I am assuming English is not a first language?"

-- Bob Linden



Help for New Java Developers

2nd Edition
Covers Java 2 SDK 1.4

Learning Java™

O'REILLY®

Patrick Niemeyer & Jonathan Knudsen