James Robertson

Why Seaside?

http://www.jarober.com

James Robertson

Seaside is...

- HTML Generation
- Full Support for CSS
- Full Support for Javascript
  - Scriptaculous
  - JQuery
- An Application Server that Supports...
  - Component Based Assembly

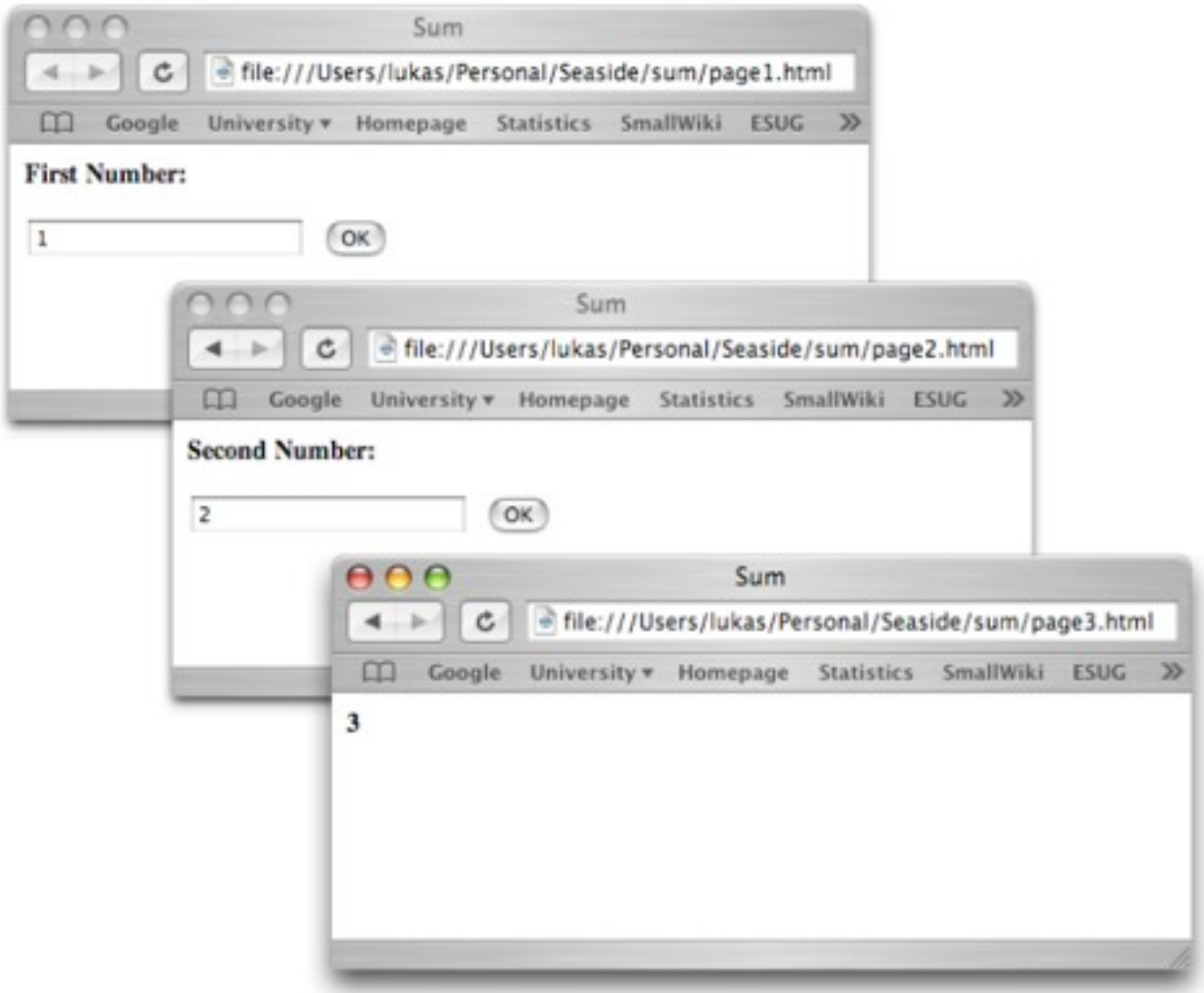
What is Seaside?

James Robertson

- A Component based Web Framework
  - Works much like a standard UI framework
  - Developers can work with familiar metaphors
  - Workflow is “normal”

James Robertson

A first example



James Robertson

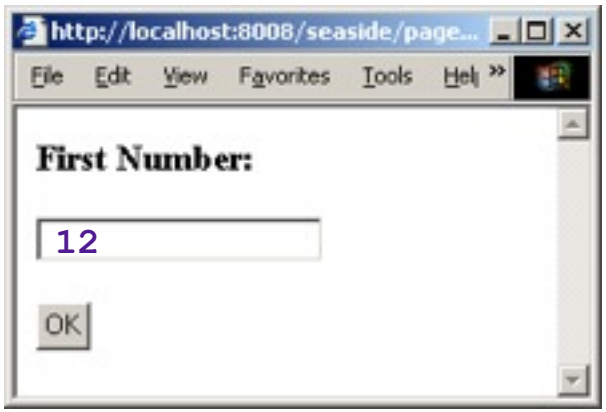
The common approach : Server Pages

```
<html><body>
          <form method="get" action="page2.ssp">
                    <h3>First Number:</h3>
                    <p><input type="text" name="val1" /></p>
                    <p><input type="submit" value="OK" /></p>
          </form>
</body> </html>
```

James Robertson

The common approach : Server Pages

```
<html><body>
        <form method="get" action="page2.ssp">
                <h3>First Number:</h3>
                <p><input type="text" name="val1" /></p>
                <p><input type="submit" value="OK" /></p>
        </form>
</body> </html>
```



James Robertson

The common approach : Server Pages

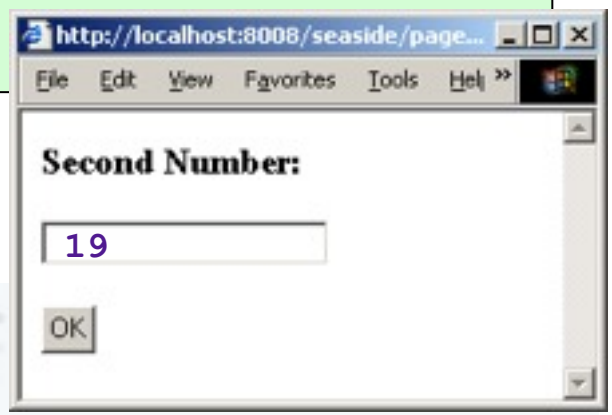
```
<html><body>
          <form method="get" action="page2.ssp">
                    <h3>First Number:</h3>
                    <p><input type="text" name="val1" /></p>
                    <p><input type="submit" value="OK" /></p>
          </form>
</body> </html>
```

James Robertson

The common approach : Server Pages

```
<html><body>
        <form method="get" action="page2.ssp">
                <h3>First Number:</h3>
                <p><input type="text" name="val1" /></p>
                <p><input type="submit" value="OK" /></p>
        </form>
</body> </html>
```

```
<html><body>
<%      val1 := request anyParameterValueAt: 'val1'.                    %>
        <form method="get" action="page3.ssp">
                <p><input type="hidden" name="val1" value="<%= val1 %>" /></p>
                <h3>Second Number:</h3>
                <p><input type="text" name="val2" /></p>
                <p><input type="submit" value="OK" /></p>
        </form>
</body> </html>
```

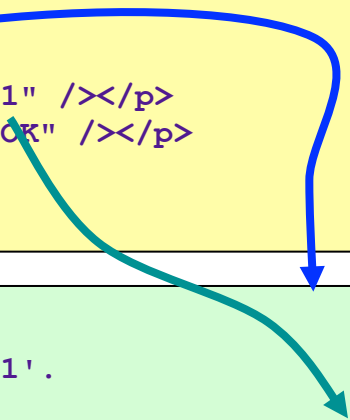


James Robertson

The common approach : Server Pages

```
<html><body>
        <form method="get" action="page2.ssp">
                <h3>First Number:</h3>
                <p><input type="text" name="val1" /></p>
                <p><input type="submit" value="OK" /></p>
        </form>
</body> </html>
```

```
<html><body>
<%      val1 := request anyParameterValueAt: 'val1'.                    %>
        <form method="get" action="page3.ssp">
                <p><input type="hidden" name="val1" value="<%= val1 %>" /></p>
                <h3>Second Number:</h3>
                <p><input type="text" name="val2" /></p>
                <p><input type="submit" value="OK" /></p>
        </form>
</body> </html>
```



James Robertson

The common approach : Server Pages

```
<html><body>
        <form method="get" action="page2.ssp">
                <h3>First Number:</h3>
                <p><input type="text" name="val1" /></p>
                <p><input type="submit" value="OK" /></p>
        </form>
</body> </html>
```


```
<html><body>
<%      val1 := request anyPar                    l1'.                    %>
        <form method="get" act
                <p><input type="hidden" name="val1" value="<%= val1 %>" /></p>
                <h3>Second Number:</h3>
                <p><input type="text" name="val2" /></p>
                <p><input type="submit" value="OK" /></p>
        </form>
</body> </html>
```

```
<html><body>
<%      val1 := request anyParameterValueAt: 'val1'.
        val2 := request anyParameterValueAt: 'val2'.                    %>
        <p><h3><%= val1 asNumber + val2 asNumber %></h3></p>
</body> </html>
```

James Robertson

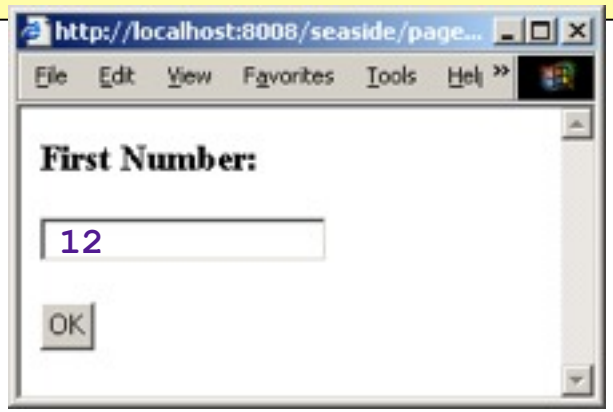
The common approach : Servlet

```
doGet: request response: response
          response write:
'<html><body>
          <form method="post" action="">
                    <h3>First Number:</h3>
                    <p><input type="text" name="val1" /></p>
                    <p><input type="submit" value="OK" /></p>
          </form>
</body></html>'
```

James Robertson

The common approach : Servlet

```
doGet: request response: response
          response write:
'<html><body>
          <form method="post" action="">
                    <h3>First Number:</h3>
                    <p><input type="text" name="val1" /></p>
                    <p><input type="submit" value="OK" /></p>
          </form>
</body></html>'
```



James Robertson

The common approach : Servlet

```
doGet: request response: response
          response write:
'<html><body>
          <form method="post" action="">
                    <h3>First Number:</h3>
                    <p><input type="text" name="val1" /></p>
                    <p><input type="submit" value="OK" /></p>
          </form>
</body></html>'
```

James Robertson

The common approach : Servlet

```
doGet: request response: response
          response write:
'<html><body>
          <form method="post" action="">
                    <h3>First Number:</h3>
                    <p><input type="text" name="val1" /></p>
                    <p><input type="submit" value="OK" /></p>
          </form>
</body></html>'
```

```
doPost: request response: response
          | val2 |
          val2 := request anyParameterValueAt: 'val2'.
          val2 isNil ifTrue: [^ self writePage2On: response forRequest: request].
          self writePage3On: response forRequest: request.
```

James Robertson

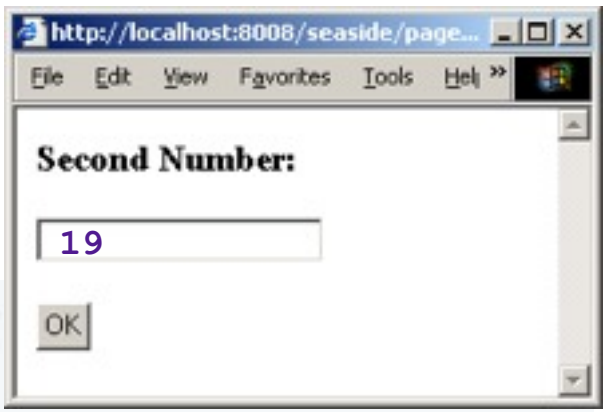
The common approach : Servlet

```
doGet: request response: response
          response write:
'<html><body>
          <form method="post" action="">
                    <h3>First Number:</h3>
                    <p><input type="text" name="val1" /></p>
                    <p><input type="submit" value="OK" /></p>
          </form>
</body></html>'
```

```
doPost: request response: response
          | val2 |
          val2 := request anyParameterValueAt: 'val2'.
          val2 isNil ifTrue: [^ self writePage2On: response forRequest: request].
          self writePage3On: response forRequest: request.
```

```
writePage2On: response forRequest: request

| val1 |
val1 := request anyParameterValueAt: 'val1'.
response write:
'<html><body>
<form method="post" action="">
   <p><input type="hidden" name="val1" value="' ,
             val1 , '" /></p>
   <h3>Second Number:</h3>
   <p><input type="text" name="val2" /></p>
   <p><input type="submit" value="OK" /></p>
</form>
</body></html>'
```



James Robertson

The common approach : Servlet

```
doGet: request response: response
          response write:
'<html><body>
          <form method="post" action="">
                    <h3>First Number:</h3>
                    <p><input type="text" name="val1" /></p>
                    <p><input type="submit" value="OK" /></p>
          </form>
</body></html>'
```

```
doPost: request response: response
          | val2 |
          val2 := request anyParameterValueAt: 'val2'.
          val2 isNil ifTrue: [^ self writePage2On: response forRequest: request].
          self writePage3On: response forRequest: request.
```

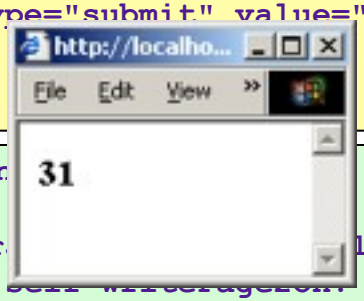
```
writePage2On: response forRequest: request

| val1 |
val1 := request anyParameterValueAt: 'val1'.
response write:
'<html><body>
<form method="post" action="">
   <p><input type="hidden" name="val1" value="' ,
             val1 , '" /></p>
   <h3>Second Number:</h3>
   <p><input type="text" name="val2" /></p>
   <p><input type="submit" value="OK" /></p>
</form>
</body></html>'
```

James Robertson

The common approach : Servlet

```
doGet: request response: response
          response write:
'<html><body>
          <form method="post" action="">
                    <h3>First Number:</h3>
                    <p><input type="text" name="val1" /></p>
                    <p><input type="submit" value="OK" /></p>
          </form>
</body></html>'
```




```
doPost: request response: respon
          | val2 |
          val2 := request anyPar                    l2'.
          val2 isNil ifTrue: [^ self writePage3On: response forRequest: request].
          self writePage3On: response forRequest: request.
```

```
writePage2On: response forRequest: request

| val1 |
val1 := request anyParameterValueAt: 'val1'.
response write:
'<html><body>
<form method="post" action="">
   <p><input type="hidden" name="val1" value="' ,
            val1 , '" /></p>
   <h3>Second Number:</h3>
   <p><input type="text" name="val2" /></p>
   <p><input type="submit" value="OK" /></p>
</form>
</body></html>'
```

```
writePage3On: response forRequest: request


| val1 val2 |
val1 := request anyParameterValueAt: 'val1'.
val2 := request anyParameterValueAt: 'val2'.
response write:
'<html><body>
            <p><h3>' ,
            (val1 asNumber + val2 asNumber) printString
             , '</h3></p>
</body></html>'
```

James Robertson

Stepping back : BASIC

```
10 input " Enter first number: ", val1
   input " Enter second number: ", val2
   print " Result: ", val1 + val2
   input " More ? ", reply$
   if reply$ = "yes"goto 10
```

The Seaside Way

James Robertson

- Iterative Development
  - "Normal" Web Development isn't
  - Seaside Brings it back
  - Uses all the Smalltalk tools
    - Debugger, etc

James Robertson

The Seaside Way

```
| val1 val2 |

val1 := self request: 'Enter first number:'.

val2 := self request: 'Enter second number:'.

self inform: val1 asNumber + val2 asNumber.
```

James Robertson

Seaside Features

- Sessions as continuous piece of code
- Handles HTTP complexities
- XHTML/CSS building
- Callback based event-model
- Composition and Reuse
- Decorations
- Development tools
- Interactive debugging
- Javascript Integration

James Robertson

Seaside User Interface Design

XHTML by Seaside and Developers

CSS2 by Graphic designers

Seaside Development

James Robertson

- Brings Smalltalk Productivity to the party
  - No gluing of disparate tools
  - Everything in one place
  - Leverage presentation/domain separation
- Build Web Apps like you build regular apps
  - Component Based Assembly

James Robertson

XHTML generation

```
html div id: #title; with: 'Title'.
html div id: #list; with:
        [html span class: #item; with: 'Item 1'.
        html span class: #item; with: 'Item 2'].
```

James Robertson

XHTML generation

```
html div id: #title; with: 'Title'.
html div id: #list; with:
        [html span class: #item; with: 'Item 1'.
        html span class: #item; with: 'Item 2'].
```

```
<div id="title">Title</div>
<div id="list">
        <span class="item">Item 1</span>
        <span class="item">Item 2</span>
</div>
```

James Robertson

Callback event model - 1

```
renderContentOn: html
   html form:
      [html submitButton
               callback: [self inform: 'Hello'];
               text: 'Say Hello']
```

James Robertson

Callback event model - 1

```
renderContentOn: html
   html form:
      [html submitButton
               callback: [self inform: 'Hello'];
               text: 'Say Hello']
```

```
<form method="post" action="HelloExample">
    <input name="_s" value="WwaXKYdGNmGxCoiy" type="hidden" />
    <input name="_k" value="EUCelKJU" type="hidden" />
    <input value="Say Hello" name="1" type="submit" />
</form>
```

James Robertson

Callback event model - 2

```
renderContentOn: html
    html form:
        [html text: 'Username:'.
        html textInput
                callback: [:value | username := value].
        html break.
        html submitButton
                callback: [self inform: 'Hi ' , username];
                text: 'Say Hello']
```

James Robertson

Callback event model - 2

```
renderContentOn: html
    html form:
        [html text: 'Username:'.
        html textInput
                callback: [:value | username := value].
        html break.
        html submitButton
                callback: [self inform: 'Hi ' , username];
                text: 'Say Hello']
```

```
<form method="post" action="HelloExample">
    <input name="_s" value="WwaXKYdGNmGxCoiy" type="hidden" />
    <input name="_k" value="ZUojGfTw" type="hidden" />
    Username:
    <input name="1" type="text" />
    <br />
    <input value="Say Hello" name="2" type="submit" />
</form>
```

Demo

James Robertson

- Simple Form Based Application
  - Input a Value
  - Add it to an existing value

What About Flow Control?

James Robertson

- Tasks
  - Very Much Like “procedural” tasks in a GUI
  - Built in Support in Seaside with WATask
  - Demo

James Robertson

Composition and reuse

```
initialize
    self children: OrderedCollection new.
    self children add: HelloExample1 new.
    self children add: HelloExample2 new.
    self children add: WACounter new.
```

```
renderContentOn: html
    self children
        do:
            [:each | html render: each]
        separatedBy: [html break]
```

James Robertson

Composition and reuse

- Tabs
- Forms
- Dialogs
- Wizards
- Reports
- Batches
- …

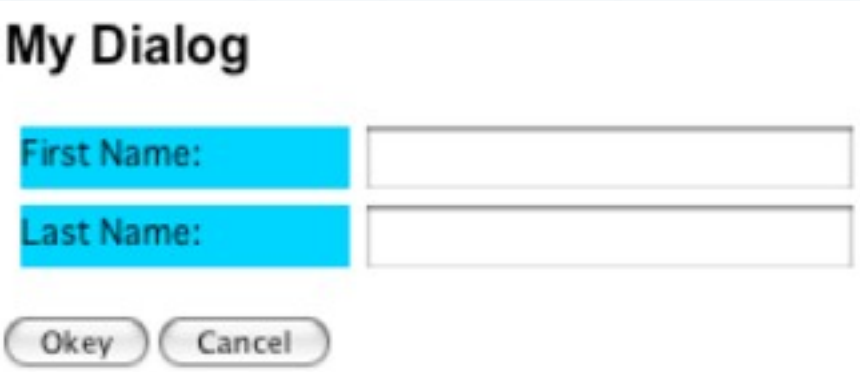
<< 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 ... 865 >>

James Robertson

Decorations : Views

- Headers
- Footers
- Borders
- Windows
- Buttons

- Alternate Look



James Robertson

Decorations : Filters

- Once
  - Prevent the use of the back-button
- Transaction
  - Disallow coming back to the component
- Authentication
  - Require authentication
- Validation
  - Validate enclosed component

James Robertson

Development tools

- Enabling Halos

- Editing Code on the Run
- Inspecting XHTML Source
- Inspecting Component Model
- Editing Stylesheet (CSS)
- Validating XHTML
- Debugging

Higher Level Tools?

James Robertson

- Sea Breeze from Georg Heeg
  - Adds a GUI Builder
  - Works much like standard UI Builders
  - http://seabreeze.heeg.de/
- WebVelocity from Cincom
  - Database driven web development
  - http://www.cincomsmalltalk.com/main/products/webvelocity/
- tODE from Gemstone (VMWare)
  - A Seaside based development environment
  - Gemstone is an OODB
  - http://gemstonesoup.wordpress.com/2011/08/12/a-peek-at-tode/

Portability

James Robertson

- Seaside is an OSS Framework
  - Maintained and Developed in Pharo
  - Portable across all major Smalltalks
  - Lots of tutorial material available

A Simple Example

James Robertson

- A Simple “tutorial” blog server
  - In VisualWorks
  - In VA Smalltalk
  - Same code could be migrated to Pharo as well

Tutorials

James Robertson

- See http://www.jarober.com/blog/st4u_seaside.ssp
- Lots of other good resources as well

James Robertson

Summary

- High-level framework
- High-level of abstraction
  - Request / Response
  - HTTP
  - XHTML
  - Javascript
  - Composition / Reuse
- Designed for development of web applications

James Robertson

Contact Info

- James Robertson
  - http://www.jarober.com
  - jarober@gmail.com
  - @jarober (Twitter)
  - http://www.facebook.com/jarober
  - http://www.youtube.com/jarober
  - Independent Misinterpretations
    - http://www.jarober.com/blog/im.ssp
  - Smalltalk 4 U
    - http://www.jarober.com/blog/st4u.ssp