@renggli question your talk tomorrow: if i already use Seaside heavily, is it worth getting up at 4:30 am to watch or is it..

about 12 hours ago via Echofon
Web Applications
Cmsbox – systematic content management

Watch the cmsbox video (3 min)

It just works.

Cmsbox, the beautifully designed, super-easy, yet powerful and flexible Content Management System (CMS) offers everything you need to create, edit and enhance the content of your web site. All elements and contents can be arranged and customized directly on your own web site.

» view more

Simply professional.

Cmsbox makes sure your site will always look great. With centrally defined format style sheets and a flexible column layout, all types of content will automatically adapt to the individual website design.

» become an associate

Subscribe to our Newsletter
Control Flow
Built in Security
Agile Manifesto

- Individuals and Interactions
- Working Software
- Customer Collaboration
- Responding to Change

http://agilemanifesto.org/
Agenda

⭐ Individuals and Interactions
⭐ Working Software
⭐ Customer Collaboration
⭐ Responding to Change

http://agilemanifesto.org/
Individuals and Interactions
over processes and tools
Sprints & Camps
Seaside Sprint
Amsterdam 2008
Motivated Individuals
Presentations & Demonstrations

Photos: Adriaan van Os
Blogs & Tweets
Dynamic Web Development with seaside

Stéphane Ducasse, Lukas Renggli, David C. Shaffer, Rick Zaccone with Michael Davies

Books & Articles

Dynamically Typed Languages

www.computer.org/software

An Introduction to Seaside
Developing Web Applications with Squeak and Smalltalk
Individuals and interactions over processes and tools

and we have mandatory processes and tools to help how those individuals interact.

http://halfarsedagilemanifesto.org/
Mailing-Lists
<table>
<thead>
<tr>
<th>Subject</th>
<th>From</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Seaside] return character magic?</td>
<td>Johan ... Stéphane, Reza</td>
<td>20:28 30/08/2010</td>
</tr>
<tr>
<td>[Seaside] JQueryWidgetBox -&gt; JQWidgetBox-MbMenu</td>
<td>- That error is saying that</td>
<td>08:19</td>
</tr>
<tr>
<td>[Seaside] freeform placement of elements in seaside...</td>
<td>- On 8/30/10 7:27 PM, Bob</td>
<td>06:25</td>
</tr>
<tr>
<td>[Seaside] markup - how to</td>
<td>laurent, me</td>
<td>08:42</td>
</tr>
<tr>
<td>[Seaside] JQuery UI slider updates textfield and textfield updates slider: how</td>
<td>radoslav</td>
<td>23:27</td>
</tr>
<tr>
<td>[Seaside] JQuery options - There are various</td>
<td>radoslav</td>
<td>30:08</td>
</tr>
<tr>
<td>[Seaside] Seaside testing and WebTester w/ Selenium</td>
<td>- On Sat, Aug 28, 2010 at 1:</td>
<td>28:30</td>
</tr>
<tr>
<td>[Seaside] jQuery lightbox in Seaside 3.0</td>
<td>me, Facundo</td>
<td>20:23</td>
</tr>
<tr>
<td>[Seaside] Problem with static css file</td>
<td>Robert, Sean, Facundo</td>
<td>20:23</td>
</tr>
<tr>
<td>[Seaside] iPhone and iUI - I understand that</td>
<td>Carl Gustaf</td>
<td>20:43</td>
</tr>
<tr>
<td>[Seaside] Trouble getting started - Casual</td>
<td>Rick, Miguel</td>
<td>20:43</td>
</tr>
<tr>
<td>[Seaside] Connection closed failure - should</td>
<td>Andy, John</td>
<td>20:43</td>
</tr>
<tr>
<td>[Seaside] Some of the Seaside.js jQuery UI</td>
<td>Philippe, me, Sven</td>
<td>20:43</td>
</tr>
<tr>
<td>[Seaside] Load Balancing Multiple Seaside Images with mod_proxy_balance</td>
<td>Robert, Sean, Facundo</td>
<td>20:43</td>
</tr>
<tr>
<td>[Seaside] Seaside builder - Where did the</td>
<td>Apostolis, TimM, me</td>
<td>20:43</td>
</tr>
<tr>
<td>[Seaside] onkeypress JQuery - If you</td>
<td>Robert Sirois</td>
<td>20:43</td>
</tr>
<tr>
<td>[Seaside] Interesting Comet failure - Should</td>
<td>TimM</td>
<td>20:43</td>
</tr>
<tr>
<td>[Seaside] Seaside environment copy strangeness in pier on Seaside 3.0</td>
<td>radoslav</td>
<td>20:43</td>
</tr>
<tr>
<td>[Seaside] Query expressions - Fair enough!</td>
<td>me, Facundo</td>
<td>20:43</td>
</tr>
</tbody>
</table>

seaside@lists.squeakfoundation.org
seaside-dev@lists.squeakfoundation.org
Seaside provides a layered set of abstractions over HTTP and XHTML that let you build highly interactive web applications quickly, reusably and maintainably. It is based on Smalltalk, a proven and robust language that is implemented by different vendors.

Seaside includes:

- **Programmatic XHTML generation.** A lot of markup is boilerplate: the same patterns of lists, links, forms and tables show up on page after page. Seaside has a rich API for generating XHTML that lets you abstract these patterns into convenient methods rather than pasting the same sequence of tags into templates every time.

- **Callback-based request handling.** Why should you have to come up with a unique name for every link and form input on your page, only to extract them from the URL and request fields later? Seaside automates this process by letting you associate blocks, not names, with inputs and methods instead of ids and strings.

- **Embedded components.** Stop thinking a whole page at a time; Seaside lets you build your UI as a tree of individual, stateful component objects, each encapsulating a small part of a page. Often, these can be used over and over again, within and between applications - nearly every application, for example, needs a way to present a batch list of search results, or a table with sortable columns, and Seaside includes components for these out the box.

- **Modal session management.** What if you could express a complex, multi-page workflow in a single method? Unlike servlet models which require a separate handler for each page or request, Seaside models an entire user session as a continuous piece of code, with natural, linear control flow. In Seaside, components can call and return to each other like subroutines; string a few of those calls together in a method, just as if you were using console I/O or opening modal dialog boxes, and you have a workflow. And yes, the back button will still work.

Seaside also has good support for CSS and javascript, excellent web-based development tools and debugging support, a rich configuration and preferences framework, and more.

If you would like to contribute, please visit [Seaside's contributors page](http://code.google.com/p/seaside/).
Seaside 3.0

**Project Description**

This is the official code repository for Seaside 3.0. All code within this project is MIT licensed. The repository is publicly readable. To gain write access, please ask in the development mailing-list. To manually load Seaside 3.0 follow the instructions on the development wiki.

If you are interested in contributing, check out Seaside's contributors page.

**Links**

- Web Site
- Issue Tracker
- Coding Conventions

**Related Repositories**

- Seaside 3.0 LGPL Code
- Seaside 3.0 Addons
- Seaside 3.0 Deprecated
- Seaside 2.8 and older versions

**Members**

- Creator: marshal
- Admin: Seaside Admins
- Developer: Seaside Developers

**Tags**

seaside, server

**License**

Code committed to this repository will be automatically under MIT license.

**Registration**

MCHttpRepository

location: `http://www.squeaksource.com/Seaside30`
+1
Working Software
over comprehensive documentation
Iterative Development
Release Cycle for Seaside 3.0
Release Cycle for Seaside 3.0
Seaside Walkback

ZeroDivide:

Debug Proceed Full Stack

Stack Trace

1. thisContext
   SmallInteger>>/
   self
   1
   aNumber
   0

2. thisContext
   WACounter>>decrease
   self
   a WACounter

http://localhost:8080/examples/counter?_s=Td05S7n2IdKh9jd&k=d2Z6i-ed...
Seaside Walkback

ZeroDivide:

Debug Proceed Full Stack

Stack Trace

1. thisContext
   SmallInteger>>/
   self
   1
   aNumber
   0

2. thisContext
   WACounter>>decrease
   self
   aWACounter

http://localhost:8080/examples/counter?_s=1

self
all inst vars
decoration
count

thisContext
stack top
all temp vars
Seaside Walkback

ZeroDivide:

Debug Proceed Full Stack

Stack Trace

1. thisContext
   SmallInteger>>/
   self
   1
   aNumber
   0

2. thisContext
   WACounter>>decrease
   self
   aWACounter
   http://localhost:8080/examples/counter?_s=
We are always using the latest seaside*
Continuous attention to technical excellence and good design.
Working software is the principal measure of progress.
Continuous Integration
Hudson
Hudson
Yanni Chiu
Lukas Renggli
Philippe Marshall
A set of scripts to build Pharo images with Hudson — Read more
http://hudson.lukas-renggli.ch

added helvetia icon

renggli (author)

history

<table>
<thead>
<tr>
<th>name</th>
<th>age</th>
<th>message</th>
</tr>
</thead>
<tbody>
<tr>
<td>.gitignore</td>
<td>May 16, 2010</td>
<td>more mac metadata to ignore [renggli]</td>
</tr>
<tr>
<td>LICENSE</td>
<td>July 20, 2010</td>
<td>fixed some typos [renggli]</td>
</tr>
<tr>
<td>README</td>
<td>July 03, 2010</td>
<td>do not link builds [renggli]</td>
</tr>
<tr>
<td>build-oneclick.sh</td>
<td>July 31, 2010</td>
<td>fixed naming of builds [renggli]</td>
</tr>
<tr>
<td>build-resize.sh</td>
<td>August 15, 2010</td>
<td>unify the way images are specified [renggli]</td>
</tr>
<tr>
<td>build.sh</td>
<td>July 19, 2010</td>
<td>fixed some basic stuff [renggli]</td>
</tr>
<tr>
<td>builds/</td>
<td>May 15, 2010</td>
<td>cleanup &amp; link sources into build dir [renggli]</td>
</tr>
<tr>
<td>cache/</td>
<td>May 15, 2010</td>
<td>cleanup &amp; link sources into build dir [renggli]</td>
</tr>
<tr>
<td>images/</td>
<td>May 15, 2010</td>
<td>cleanup &amp; link sources into build dir [renggli]</td>
</tr>
</tbody>
</table>
This build server is using the infrastructure from [github.com/renngli/builder](http://github.com/renngli/builder) to build Pharo images, run tests, calculate coverage, and run code checker. For additional information on the projects being built on this server see [www.lukas-renngli.ch](http://www.lukas-renngli.ch).

<table>
<thead>
<tr>
<th>Job</th>
<th>Last Success</th>
<th>Last Failure</th>
<th>Last Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chic</td>
<td>22 hr (#45)</td>
<td>N/A</td>
<td>5 min 27 sec</td>
</tr>
<tr>
<td>Development</td>
<td>22 hr (#178)</td>
<td>N/A</td>
<td>27 min</td>
</tr>
<tr>
<td>Filesystem</td>
<td>22 hr (#6)</td>
<td>N/A</td>
<td>7 min 20 sec</td>
</tr>
<tr>
<td>GitFS</td>
<td>22 hr (#238)</td>
<td>N/A</td>
<td>10 min</td>
</tr>
<tr>
<td>Glamour</td>
<td>1 hr 53 min (#35)</td>
<td>N/A</td>
<td>13 min</td>
</tr>
<tr>
<td>Helvetia</td>
<td>1 day 13 hr (#58)</td>
<td>N/A</td>
<td>27 min</td>
</tr>
<tr>
<td>Magritte</td>
<td>1 day 10 hr (#87)</td>
<td>N/A</td>
<td>16 min</td>
</tr>
<tr>
<td>Magritte 2</td>
<td>14 hr (#200)</td>
<td>N/A</td>
<td>6 min 26 sec</td>
</tr>
<tr>
<td>OpalCompiler</td>
<td>22 hr (#24)</td>
<td>N/A</td>
<td>10 min</td>
</tr>
<tr>
<td>PetitParser</td>
<td>21 hr (#309)</td>
<td>N/A</td>
<td>30 min</td>
</tr>
<tr>
<td>Pharo</td>
<td>3 days 11 hr (#44)</td>
<td>N/A</td>
<td>12 min</td>
</tr>
<tr>
<td>Pier</td>
<td>1 day 9 hr (#94)</td>
<td>N/A</td>
<td>35 min</td>
</tr>
<tr>
<td>Pier 2</td>
<td>14 hr (#225)</td>
<td>N/A</td>
<td>23 min</td>
</tr>
<tr>
<td>Seaside 2.8</td>
<td>21 hr (#157)</td>
<td>N/A</td>
<td>40 min</td>
</tr>
<tr>
<td>Seaside 3.0</td>
<td>16 hr (#301)</td>
<td>N/A</td>
<td>1 hr 44 min</td>
</tr>
<tr>
<td>TextLint</td>
<td>1 day 10 hr (#16)</td>
<td>N/A</td>
<td>1 hr 6 min</td>
</tr>
</tbody>
</table>
Project Seaside 3.0

A Seaside 3.0 image.

Workspace

Last Successful Artifacts
- Seaside-3.0-OneClick.zip
- seaside3.changes
- seaside3.image
- seaside3-swazoo.changes
- seaside3-swazoo.image
- seaside3-tests.changes
- seaside3-tests.image

Recent Changes

Latest Test Result (no failures)

Upstream Projects
- Development

Downstream Projects
- Magritte 2

Permalinks
- Last build (#301), 16 hr ago
- Last stable build (#301), 16 hr ago
- Last successful build (#301), 16 hr ago

Page generated: Aug 23, 2010 10:10:52 AM

Hudson ver. 1.373
Source Code Management
- None
- CVS
- Subversion

Build Triggers
- Build after other projects are built
  Projects names: Development
  Multiple projects can be specified like 'abc, def'
- Trigger builds remotely (e.g., from scripts)
- Build periodically
- Poll SCM
- Build when a URL's content changes
  URL: http://www.squeaksources.com/Seaside30/

Build
- Execute shell
  Command:
  ```bash
  build.sh -i omnibrowser -s seaside3 -s seaside3-komanche -s seaside-design -o seaside3 build.sh -i seaside3 -s testrunner -s seaside3-tests -o seaside3-tests build.sh -i seaside3 -s seaside3-swazoo -o seaside3-swazoo
  build-oneclick.sh -i seaside3 -o Seaside-3.0-OneClick -n Seaside -t Seaside -v 3.0 -c Seaside
  ```

See the list of available environment variables

Add build step

Post-build Actions
- Publish Checkstyle analysis results
  Checkstyle results: **/*-Lint.xml
  Fileset includes setting specifies the generated raw CheckStyle XML report files, such as **/checkstyle-result.xml. Basedir of the fileset is the workspace root. If no value is set, then the default **/checkstyle-result.xml is used. Be sure not to include any non-report files into this pattern.
- Publish Javadoc
- Archive the artifacts
Seaside 3.0 Config [Hudson]

Source Code Management
- None
- CVS
- Subversion

Build Triggers
- Build after other projects are built

Projects names: Development
- Multiple projects can be specified like 'abc, def'

- Trigger builds remotely (e.g., from scripts)
- Build periodically
- Poll SCM
- Build when a URL's content changes

URL: 
http://www.squeaksource.com/Seaside30/

Build
- Execute shell

Command:
build.sh -i omnibrowser -s seaside3 -s seaside3-komanche -s seaside-design -o seaside3 
bUILD -i seaside3 -s testrunner -s seaside3-tests -o seaside3-tests 
bUILD -i seaside3 -s swazoo -o seaside3-swazoo

build-oneclick.sh -i seaside3 -o Seaside-3.0-OneClick -n Seaside -t Seaside -v 3.0 -c Seaside

See the list of available environment variables

Add build step

Post-build Actions
- Publish Checkstyle analysis results

Checkstyle results:
- Fileset includes setting that specifies the generated raw CheckStyle XML report files, such as */checkstyle-result.xml. Basedir of the fileset is the workspace root. If no value is set, then the default */checkstyle-result.xml is used. Be sure not to include any non-report files into this pattern.

- Publish Javadoc
- Archive the artifacts
Build triggers:
- Check "Build after other projects are built"
- Check "Build when a URL's content changes"
- Set URL to http://www.squeaksources.com/Seaside30/

Build actions:
- Publish Checkstyle analysis results
Seaside 3.0 Config [Hudson]

Source Code Management
- None
- CVS
- Subversion

Build Triggers
- Build after other projects are built
- Projects names: Development
  Multiple projects can be specified like 'abc, def'
- Trigger builds remotely (e.g., from scripts)
- Build periodically
- Poll SCM
- Build when a URL's content changes
  URL: http://www.squeaksource.com/Seaside30/

Build
- Execute shell
  Command:
  ```bash
  build.sh -i omnibrowser -s seaside3 -s seaside3-komanche -s seaside-design -o seaside3
  build.sh -i seaside3 -s testrunner -s seaside3-tests -o seaside3-tests
  build.sh -i seaside3 -s swazo -o seaside3-swazo
  build-oneclick.sh -i seaside3 -o Seaside-3.0-OneClick -n Seaside -t Seaside -v 3.0 -c Seaside
  ```
  See the list of available environment variables

Add build step

Post-build Actions
- Publish Checkstyle analysis results
  Checkstyle results:
  ```xml
  Fileset includes setting that specifies the generated raw CheckStyle XML report files, such as **/checkstyle-result.xml. Basedir of the fileset is the workspace root. If no value is set, then the default **/checkstyle-result.xml is used. Be sure not to include any non-report files into this pattern.
  ```
  ```xml
  Advanced...
  ```
- Publish Javadoc
- Archive the artifacts
Seaside 3.0 Config [Hudson]

Source Code Management
- None
- CVS
- Subversion

Build Triggers
- Build after other projects are built
- Projects names: Development
- URL: http://www.squeaksource.com/Seaside30/

Build
- Execute shell
  Command: build.sh -i omnibrowser -s seaside3 -s seaside3-komanche -s seaside-design -o seaside3
  build.sh -i seaside -s seaside3-tests -s seaside3-tests -o seaside3-tests
  build.sh -i seaside -s seaside3-swazoo -o seaside3-swazoo
  build-oneclick.sh -i seaside3 -o Seaside-3.0-OneClick -n Seaside -t Seaside -v 3.0 -c Seaside

Post-build Actions
- Publish Checkstyle analysis results
  Checkstyle results: **/**-Lint.xml
  Fileset includes setting specifies the generated raw CheckStyle XML report files, such as **/checkstyle-result.xml. Basedir of the fileset is the workspace root. If no value is set, then the default **/checkstyle-result.xml is used. Be sure not to include any non-report files into this pattern.
- Publish Javadoc
- Archive the artifacts
Test Results (SUnit)

### All Tests

<table>
<thead>
<tr>
<th>Package</th>
<th>Duration</th>
<th>Fail</th>
<th>Fail (diff)</th>
<th>Skip</th>
<th>Skip (diff)</th>
<th>Total</th>
<th>Total (diff)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comet.Tests.Core</td>
<td>12 ms</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Grease.Tests.Core</td>
<td>0.61 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>377</td>
<td>377</td>
</tr>
<tr>
<td>Grease.Tests.Pharo.Core</td>
<td>0.28 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>Grease.Tests.Slime</td>
<td>21 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>18</td>
<td>18</td>
</tr>
<tr>
<td>JQuery.Tests.Core.Unit</td>
<td>0.8 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>206</td>
<td>206</td>
</tr>
<tr>
<td>JQuery.Tests.UI.Unit</td>
<td>0.2 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>23</td>
<td>23</td>
</tr>
<tr>
<td>Javascript.Tests.Core</td>
<td>0.28 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>124</td>
<td>124</td>
</tr>
<tr>
<td>Prototype.Tests.Core</td>
<td>0.68 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>34</td>
<td>34</td>
</tr>
<tr>
<td>RSS.Tests.Core</td>
<td>24 ms</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Scriptaculous.Tests.Core.Unit</td>
<td>48 ms</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>Seaside.Tests.Adaptors.Camanche</td>
<td>12 ms</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Seaside.Tests.Canvas</td>
<td>0.37 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>82</td>
<td>82</td>
</tr>
<tr>
<td>Seaside.Tests.Component.Tests</td>
<td>36 ms</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>Seaside.Tests.Core</td>
<td>0 ms</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Seaside.Tests.Core.Backtracking</td>
<td>12 ms</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>Seaside.Tests.Core.Cache</td>
<td>0.29 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>Seaside.Tests.Core.Callbacks</td>
<td>48 ms</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>Seaside.Tests.Core.Configuration</td>
<td>24 ms</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Seaside.Tests.Core.Document</td>
<td>12 ms</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Seaside.Tests.Core.HTTP</td>
<td>0.32 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>221</td>
<td>221</td>
</tr>
<tr>
<td>Seaside.Tests.Core.Libraries</td>
<td>0.49 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>33</td>
<td>33</td>
</tr>
<tr>
<td>Seaside.Tests.Core.Rendering</td>
<td>12 ms</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Seaside.Tests.Core.RequestHandling</td>
<td>0.25 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>97</td>
<td>97</td>
</tr>
<tr>
<td>Seaside.Tests.Core.Server</td>
<td>0 ms</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Seaside.Tests.Core.Utilities</td>
<td>0.86 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>Seaside.Tests.Development</td>
<td>0 ms</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Seaside.Tests.Email</td>
<td>0 ms</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>
CheckStyle (SmallLint)
Working software
over comprehensive documentation

as long as that software is comprehensively documented.

http://halfarsedagilemanifesto.org/
Seaside 2.8

- Commented Classes: 44%
- Commented Methods: 17%

Seaside 3.0

- Commented Classes: 38%
- Commented Methods: 23%
Dynamic Web Development with Seaside

Stéphane Ducasse, Lukas Renggli, C. David Shaffer, Rick Zaccom, with Michael Davies

Seaside is the open source framework of choice for developing sophisticated and dynamic web applications. Seaside uses the power of objects to master the web. With Seaside web applications is as simple as building desktop applications. Seaside lets you build highly dynamic and interactive web applications.

Seaside supports agile development through interactive debugging and unit testing. Seaside is based on Smalltalk, a proven and robust language implemented by different vendors. Seaside is now available for all the major Smalltalk including Pharo, Squeak, GNU Smalltalk, Cincom Smalltalk, GemStone Smalltalk, and VA Smalltalk.
2.1.1 Of Mice and Menus

Because Smalltalk images are intended to work identically on many different operating systems, you may find some of the user interface may be slightly different from what you’re used to. In order to help you understand the differences, we will outline the common stumbling points here.

Click. This is a standard mouse click, and is used to move focus to an item, to select an item in a list, and to select sections of text.

Right Click. We will use right-click to describe the action that will bring up the “context menu” on an item. This menu holds a list of actions relevant to the selected item. Mac users who are using a single-button mouse will generally
Online
€ 0
Working Software
Customer Collaboration
over contract negotiation
Who is our customer?
Customers

- Ourselves (primarily)
- Consulting clients
- Mailing-list participants
- Platform vendors
If I miss something, I add it
If I break something,
I fix it
What about you?
Ask in the mailing-list

Create a issue in the bug tracker

Choose one of the following:
  - Wait for somebody to fix it
  - Submit a fix, patch, change
  - ‘Earn’ commit rights
Customer Collaboration
Responding to Change
over following a plan
Team
<table>
<thead>
<tr>
<th>Authors</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avi, Julian</td>
<td>2002</td>
</tr>
<tr>
<td>Avi, Julian, Andrew</td>
<td>2003</td>
</tr>
<tr>
<td>Avi, Andrew</td>
<td>2004</td>
</tr>
<tr>
<td>Avi, Andrew, Michel, Lukas</td>
<td>2005</td>
</tr>
<tr>
<td>Lukas, Michel, Philippe, Avi</td>
<td>2006</td>
</tr>
<tr>
<td>Lukas, Philippe, Michel</td>
<td>2007</td>
</tr>
<tr>
<td>Lukas, Philippe</td>
<td>2008</td>
</tr>
<tr>
<td>Julian, Philippe, Lukas</td>
<td>2009</td>
</tr>
<tr>
<td>Julian, Philippe, Nick, Lukas</td>
<td>2010</td>
</tr>
</tbody>
</table>
Portability
GNU Smalltalk
Pharo Smalltalk
(Development Platform)
Javascript
Seaside Async 2004

Prototype, script.aculo.us 2005

Comet 2006

jQuery, jQueryUI (+ WidgetBox) 2008

(Dojo Toolkit, Raphael, ...) 2010

... and many other libraries
seaside
squeak enterprise aubergines server

powered by

seaside

seaside

2002

2005

2007
Website
seaside
squeak enterprise aubergines server

download: seaside-0.93.cs

tutorials: part.1, part.2
documentation: design overview
resources: swiki, mailing list
bugs: mantis
Seaside: Squeak Enterprise Aubergines Server

In all fiction, when a man is faced with alternatives he chooses one at the expense of the others. In the almost unfathomable Ta'ui Pan, he chooses - simultaneously - all of them. Be thus creates various futures, various times which start others that will in turn branch out and bifurcate in other times...
- Jorge Luis Borges, The Garden of Forking Paths

Downloads:
Seaside2.st.gz

Links:
Mailing List
Swiki

Documentation:
Tutorial: A Walk on the Seaside
Recommended for first time users
Renderer API
Covers WAHtmlGenerator and WAHtmlRenderer

Design Documentation
For developers of the Seaside framework.

Seaside is a framework for developing sophisticated web applications in Smalltalk.

Its most unique feature is its approach to session management: unlike servlet models which require a separate handler for each page or request, Seaside models an entire user session as a continuous piece of code, with natural, linear control flow - pages can call and return to each other like subroutines, complex sequences of forms can be managed from a single method, objects are passed by reference rather than marshalled into URLs or hidden fields - while fully supporting the backtracking and parallelism inherent to the web browser.

Seaside also features a callback-based event model, a "transaction" system for auto-expiring pages, programmer-friendly HTML generation and designer-friendly templates, a system of reusable and embeddable UI components, and handy web-based development tools.

Seaside is maintained and supported by Avi Bryant and Julian Fizell of Beta4 Productions. We are based out of Vancouver, Canada, and available for Seaside-related consulting services.

Information on earlier versions of Seaside is available here.
Seaside

Seaside is a framework for developing sophisticated web applications in Smalltalk.

Seaside provides a layered set of abstractions over HTTP and HTML that let you build highly interactive web applications quickly, reusable and maintainably. Seaside includes:

- **Programmatic HTML generation.** A lot of markup is boilerplate: the same patterns of lists, links, forms and tables show up on page after page. Seaside has a rich API for generating HTML that lets you abstract these patterns into convenient methods rather than pasting the same sequence of tags into templates every time.

- **Callback-based request handling.** Why should you have to come up with a unique name for every link and form input on your page, only to extract them from the URL and request fields later? Seaside automates this process by letting you associate blocks, not names, with inputs and links, so you can think about objects and methods instead of ids and strings.

- **Embedded components.** Stop thinking a whole page at a time; Seaside lets you build your UI as a tree of individual, stateful component objects, each encapsulating a small part of a page. Often, these can be used over and over again, within and between applications - nearly every application, for example, needs a way to present a batched list of search results, or a table with sortable columns, and Seaside includes components for these out the box.

- **Modal session management.** What if you could express a complex, multi-page workflow in a single method? Unlike servlet models which require a separate handler for each page or request, Seaside models an entire user session as a continuous piece of code, with natural, linear control flow. In Seaside, components can call and return to each other like subroutines; string a few of those calls together in a method, just as if you were using console I/O or opening modal dialog boxes, and you have a workflow. And yes, the back button will still work.

Seaside also has good support for CSS and Javascript, excellent web-based development tools and debugging support, a rich configuration and preferences framework, and more.

Seaside is currently developed and supported by Avi Bryant, with the help of the Seaside community. Commercial support is available, and donations are welcome. Special thanks to Julian Fitzell, Adrian Lienhard, and Lukas Renggli for their contributions to Seaside, to http://www.netstyle.ch for sponsoring this site, and to Peter Macsadi for providing the Seaside logo.
Responding to Change
Where do we go?
Seaside 3.0

- Better design
- Better code
- Better performance
- Better support
- Better portability
- Better experience
Seaside 3.1

- Bug-fixes (#isolate:, #lightbox:)
- Improving configurability
- Encoding improvements
- Getting more developers involved
- Shorter release cycle
Seaside BoF

Lunch Break 12:45
In this Room