

Une introduction pragmatique à Seaside

— Le framework en Smalltalk pour développer des applications Web sophistiquées —

**Dynamic
Web Development**

with

seaside



Luc Fabresse

luc.fabresse@mines-douai.fr

- 1 Introduction
- 2 Rendu HTML et CSS
- 3 Ancres et « callbacks »
- 4 Gestion des formulaires
- 5 Composer des composants
- 6 Support Javascript
- 7 Conclusion



- 1 Introduction
- 2 Rendu HTML et CSS
- 3 Ancres et « callbacks »
- 4 Gestion des formulaires
- 5 Composer des composants
- 6 Support Javascript
- 7 Conclusion



Qu'est-ce que Seaside ?

Un framework écrit en Smalltalk pour le développement d'applications Web

Pourquoi est-ce intéressant pour vous ?

- Étudier une technologie alternative (autre que PHP, Java, Flash/Flex)
- Découvrir un framework **innovant**



Le développement Web

- HTTP
- HTML, CSS
- Formulaires, POST, GET, ...
- Persistance

Le langage Smalltalk

- Langage orienté objets
- Dynamiquement typé
- Réflexif



Search the Seaside

seaside



The framework for developing sophisticated web applications in Smalltalk

About

- [Screenshots](#)
- [Examples](#)
- [Hosting](#)
- [Support](#)
- [Success Stories](#)

[more](#)

Documentation

Dynamic Web Development The open book with  [Development with Seaside](#) is available online. [Pdf](#) and [paper](#) versions are available too.

Also see: [FAQ](#), [Tutorials](#), [Migration](#), [Videos](#), and [more](#).

Community

- [Weblogs](#)
- [Mailing Lists](#)
- [Development](#)
- [Contribute](#)
- [Merchandise](#)
- [Extensions](#)
- [Projects](#)

[more](#)

Seaside 3.0

Unit Tests



Version	Unit Tests
Seaside 2.4	10
Seaside 2.5	20
Seaside 2.6	30
Seaside 2.7	100

News

[techninik: in comedy in cat-v](#) 2011-04-10T10:33:00-00:00
[01:14] smalltalk considered harmful? [01:14] EthanG, script kiddie :P [01:14] bah XD [01:15] cur...

[techninik: seaside smalltalk web framework smalltalk wow free book](#) 2011-04-10T07:53:00-00:00
seaside smalltalk web framework smalltalk wow free book.
<http://book.seaside.st/book/introduction/wh>.

[Mocktry: I](#) 2011-04-07T02:59:21-00:00
: Smalltalk, TDD, Seaside, etc. . . Pharo/Squeak? Mocktry. Posted on 07/04/2011 by c...

[Squeak Smalltalk: SqueakLicense](#) 2011-04-06T16:16:35-00:00
www.squeak.org/: Squeak: The best little smalltalk ever; and it's FREE. Squeak really is awesome; if...

[Smalltalk An Introduction to Application Development Using ...](#) 2011-04-06T16:16:35-00:00
Download Cincom Smalltalk 2008 Non Commercial VisualWorks 7.6 ObjectStudio 8.1 hotfile rapidshare me...

[more](#)

download



Seaside is a free and [Open Source™](#) web application framework distributed under the [MIT License](#).

Seaside is available on the following Smalltalk platforms:

- [Pharo Smalltalk](#) (download)
- [Cincom Smalltalk](#)
- [Dolphin Smalltalk](#)
- [GemStone Smalltalk](#)
- [GNU Smalltalk](#)
- [Squeak Smalltalk](#)
- [VA Smalltalk](#)

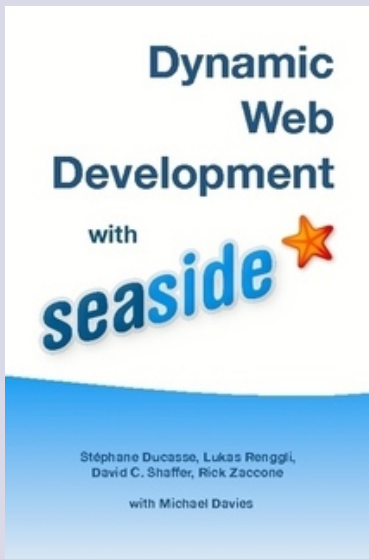
[more](#)



III www.esug.org



<http://book.seaside.st>



Démo WACounter

Plan

- 1 Introduction
- 2 Rendu HTML et CSS
- 3 Ancres et « callbacks »
- 4 Gestion des formulaires
- 5 Composer des composants
- 6 Support Javascript
- 7 Conclusion



Qu'est-ce qu'un *composant* ?

- un élément visuel d'une page HTML (généralement une `div`)
- une instance d'une sous-classe de `WComponent`

Notion de *composant principal*

Toute application Web en Seaside possède un composant principal



Affichage d'un composant en XHTML :

Un composant «sait s'auto-afficher» en HTML via sa méthode `renderContentOn:` (héritée de `WComponent`). Cette méthode est automatiquement appelée par Seaside.

La méthode `renderContentOn:`

prend un paramètre (nommé `html`) instance de `WRenderCanvas`

La classe `WRenderCanvas`

fournit une interface permettant de facilement générer du XHTML **valide** en utilisant des « pinceaux »



Les « pinceaux » XHTML

Le pinceau paragraph

```
html paragraph: 'a simple text'.  
html paragraph  
  with: 'a simple text'
```

Le pinceau div

```
html div: 'a simple text'.  
html div  
  with: 'a simple text'
```

Utilisation combinée

```
html div  
  with: [html paragraph: [ 'a simple text' ].  
        html image url: 'http://www.seaside.st/styles/logo-plain.png']
```

Le pinceau universel render:

```
MyComponent>>renderContentOn: html  
  html render: 'une chaine'.  
  html render: anotherComponent
```



- Un composant MyApp est une sous-classe de WComponent
- Pour qu'un composant puisse être une application (point d'entrée), ajouter la méthode :

```
MyApp class>>canBeRoot  
  ^true
```

- Pour que ce composant affiche du HTML, définissez la méthode :

```
MyApp>>renderContentOn: html  
  html text: 'My App is so cool'
```

- Utiliser les pinceaux
<http://book.seaside.st/book/fundamentals/rendering-components/learning-canvas-and-brush>

A propos de renderContentOn:

- Ne fait qu'afficher
- Pas d'instanciation, pas de modification, ...



- 1 Introduction
- 2 Rendu HTML et CSS**
- 3 Ancres et « callbacks »
- 4 Gestion des formulaires
- 5 Composer des composants
- 6 Support Javascript
- 7 Conclusion



Le message class: des pinceaux

```
html div
  class: 'center';
  with: 'Seaside is cool'.
html paragraph
  class: 'highlight';
  with: 'Highlighted text'
```

Le message class:if: des pinceaux

```
html unorderedList with: [
  1 to: 10 do: [ :index |
    html listItem
      class: 'even' if: index even;
      with: index ] ]
```



Solution 1 : la méthode style des composants

```
MyComponent>>style  
  ^ '.even { background-color: light-gray;}'
```

Solution 2 : un fichier CSS unique

- ① dans un fichier externe (<link ...)

```
MyComponent>>updateRoot: anHtmlRoot  
  super updateRoot: anHtmlRoot.  
  anHtmlRoot stylesheet url: 'http://car.ensm-douai.fr/site21/mystyle.'
```

- ② dans une FileLibrary (sous classe de WFileLibrary)

```
MyFileLibrary addAllFilesIn: '/path/to/directory'  
MyFileLibrary addFileAt: '/path/to/mystyle.css'
```

```
MyComponent>>updateRoot: anHtmlRoot  
  super updateRoot: anHtmlRoot.  
  anHtmlRoot stylesheet url: MyFileLibrary / #mystyleCss
```



[http ://pharo.pharocloud.com/](http://pharo.pharocloud.com/)



- 1 Introduction
- 2 Rendu HTML et CSS
- 3 Ancres et « callbacks »**
- 4 Gestion des formulaires
- 5 Composer des composants
- 6 Support Javascript
- 7 Conclusion



Affichage d'un lien

```
MyComponent>>renderContentOn: html
  html anchor
    url: 'http://www.seaside.st';
    with: 'Seaside Website'
```

Notion de callback

Un block (fonction anonyme) qui sera executé *automatiquement* lorsque l'ancre sera cliquée. **Pas d'affichage** (le renderer html est invalide) dans un callback.

Exemple de callback

```
MyComponent>>renderContentOn: html
  html anchor
    callback: [ self clicked ] ;
    with: 'You already clicked me ', number, ' times'
MyComponent>>clicked
  number := number + 1
```



Démo WAMultiCounter



- 1 Introduction
- 2 Rendu HTML et CSS
- 3 Ancres et « callbacks »
- 4 Gestion des formulaires**
- 5 Composer des composants
- 6 Support Javascript
- 7 Conclusion



Edition des données d'un utilisateur

```
UserEditorComponent>>renderContentOn: html
  html form: [
    html text: 'Login: '.
    html textInput
      callback: [ :value | self user login: value ];
      value: self login name.
    html break.
    html text: 'Email address: '.
    html textInput
      callback: [ :value | self user emailAddress: value ];
      value: self user emailAddress.
    html break.
    html submitButton
      callback: [ self save ];
      value: 'Save']
UserEditorComponent>>save
self inform: self user login , '--' , self user emailAddress
```



`http://book.seaside.st/book/fundamentals/forms/convenience`

Une application exemple : Gestion d'une todo-list
`http://book.seaside.st/book/in-action/todo`



- 1 Introduction
- 2 Rendu HTML et CSS
- 3 Ancres et « callbacks »
- 4 Gestion des formulaires
- 5 Composer des composants**
- 6 Support Javascript
- 7 Conclusion



Documentation : <http://book.seaside.st/book/components/calling>



- 1 Introduction
- 2 Rendu HTML et CSS
- 3 Ancres et « callbacks »
- 4 Gestion des formulaires
- 5 Composer des composants
- 6 Support Javascript**
- 7 Conclusion



Scriptaculous

Documentation+démo : <http://scriptaculous.seasidehosting.st/>

JQuery

Documentation+démo : <http://demo.seaside.st/javascript/jquery>

JQueryUI

Documentation+démo : <http://demo.seaside.st/javascript/jquery-ui>



- 1 Introduction
- 2 Rendu HTML et CSS
- 3 Ancres et « callbacks »
- 4 Gestion des formulaires
- 5 Composer des composants
- 6 Support Javascript
- 7 Conclusion**

Concepts et mécanismes innovants en Seaside :

- un modèle de *composants* et *callbacks*
- du code HTML toujours valide
- les objets métiers sont toujours présents (paramètres GET/POST gérés automatiquement)
- possibilité de déboguer facilement les applications Web
- encapsulation de technologies/frameworks Web 2.0 : AJAX, JQuery, ...

Alors ?

