10 Steps to Mastering

The Art of Seaside

Lukas Renggli
renggli@iam.unibe.ch

Different by Design

- We share as much state as possible.
- We don’t use clean, carefully chosen, or meaningful URLs.
- We don’t use templates to separate the model from the presentation.

Components

- Components are the Views (and Controllers) of a Seaside application.
- Components keep their state (model and state of user-interface) in instance-variables.

HelloWorld
registerAsApplication: aString

Rendering

- Override the template-method #render-ContentOn: to generate the view.
- Rendering is a read-only phase.

HelloWorld
rendererClass
renderContentOn: html

html render: ‘Hello World’
Canvas

- The argument html passed to #render-ContentOn: is an instance of a rendering-canvas.
- Render any object:
  html render: 'Hello World'
- Render a line-break:
  html break

Brushes

1. Ask the canvas for a brush:
   html div
2. Configure the brush:
   html div class: 'beautiful'
3. Render the contents of the brush:
   html div
      class: 'beautiful';
      with: 'Hello World'

Callbacks

1. Ask the canvas for an anchor:
   html anchor
2. Define the callback action:
   html anchor
      callback: [ self inform: 'Got it' ]
3. Render the contents of the anchor:
   html anchor
      callback: [ self inform: 'Got it' ];
      with: 'Get it'

Forms

- Render a from around your form elements:
  html form: [ ... ]
- Put the form elements inside the form:
  html form:
    html textInput
      value: text;
      callback: [ :value | text := value ].
    html submitButton

Call

• Temporarily replace the receiving component with a different component:
  answer := self call: aComponent

 1. call: B
 2. A

Answer

• Restore the calling component and return the resulting model object:
  self answer: anObject

 1. answer: 123
 2. A

Composition

• Nest components into each other using the composite pattern.
• Display subcomponents using the method #render: on the canvas.

  ParentComponent
  initialize
  children
  renderContentOn: html

  child := HelloWorld new
  ^ Array with: child
  html render: child

What is the Benefit?

• Did you notice, that ...
  – we talked about Web applications
  – we didn’t fiddle around with URLs
  – we didn’t serialized state back and forth
  – we implemented a complex workflow
  – we separated design and logic