

Making Web Programming Simpler: a Seaside Tutorial

Alexandre Bergel
SCG-IAM University of Bern – Switzerland
`bergel@iam.unibe.ch`

November 6, 2004

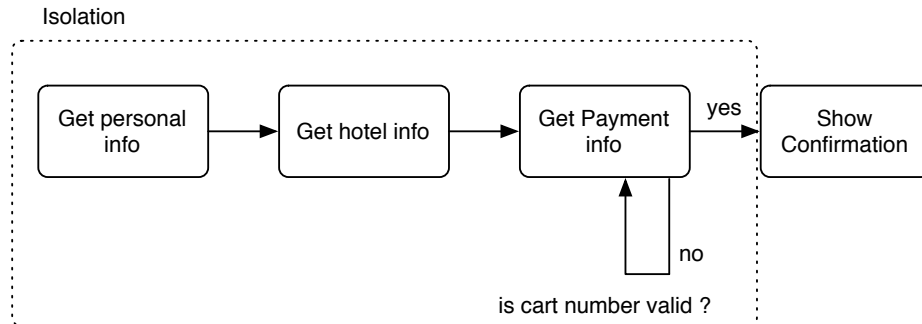
1 RegConf: An Application for Registering to a Conference

The goal of this tutorial is to give you a feeling on creating a web application using Seaside. RegConf is a tool intended to help people to register to a conference.

Four steps are necessary to complete such a registration:

1. A participant has to enter some personal data such as firstname, name, the institute where she is attached, and her email address. Then,
2. Some information about the hotel are required. For instance a room can be single or double in an hotel ranked between 1 and 4 stars. A price has then to be computed.
3. Finally informations regarding the payment are required. Once the credit card number, the issue date, and the type are entered,
4. A confirmation screen shows a summary of what was entered.

The flow of the application is described in the following figure.



The dashed rectangle designate the part of the application which is *isolated*. This means that once the flow of the running application leaves this box, there is no way to come back in it, specially using the back button.

2 Application Building Blocks

2.1 The Entry Point: RCMain

The control flow of the application has to be described in a task's `go` method. This method also represent the entry point of the application. Thus a name like `RCMain` sounds appropriated (RC stands for `RegConf`).

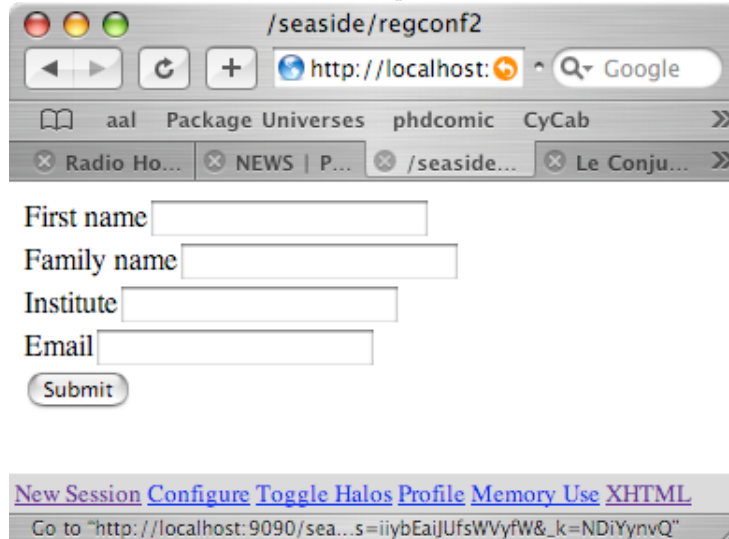
Your job: Create a task `RCMain` with a `go` method that describes the control flow of the application.

Your job: Start the web server on by executing `WAKom startOn: 9090`.

Your job: Create an `initialize` method on the class side to register your application in Seaside under the name `regconf`.

2.2 Getting User Information: RCGetUserInfo

All the control flow is defined in the class you previously defined. Getting user information is implemented as a normal seaside component (i.e., subclass of `WACComponent`). Instance variables of this class should reflect the structure of a user. Pressing the *submit* button returns to the caller component using `answer: .` Fetching the participant's informations can be done using text fields and submit button. Here is an example:



The screenshot shows a web browser window with the title `/seaside/regconf2`. The address bar shows `http://localhost:9090`. The browser has several tabs open: `Radio Ho...`, `NEWS | P...`, `/seaside...`, and `Le Conju...`. The main content area displays a form with the following fields and a button:

- First name
- Family name
- Institute
- Email
-

At the bottom of the page, there is a navigation bar with links: [New Session](#), [Configure](#), [Toggle Halos](#), [Profile](#), [Memory Use](#), and [XHTML](#). Below the navigation bar, there is a text field containing the URL: `Go to "http://localhost:9090/sea...s=iiybEaiJUfsWVvfW&_k=NDiYynvQ"`.

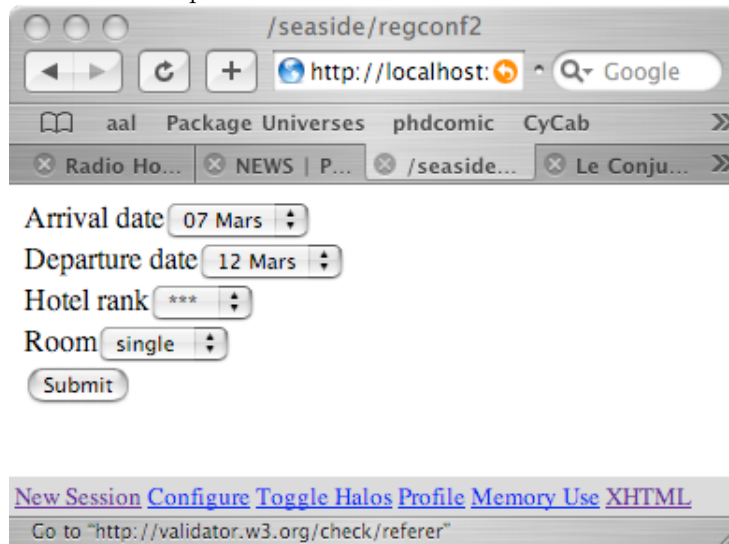
Your job: Write the method `renderContentOn:` in `RCGetUserInfo` .

Your job: Try your application using your favorite web browser. Make it point to `http://localhost:9090/seaside/regconf` .

The information passed around different states of the application can be contained in a dictionary. A more advanced design would require a class `User` for which an instance is passed around through.

2.3 Getting Hotel Information: `RCGetHotelInfo`

A list of choices is pleasant to fetch informations of the hotel.



The screenshot shows a web browser window with the address bar set to `http://localhost:9090/seaside/regconf2`. The browser's tab bar shows several tabs, including `/seaside...`. The main content area displays a form with the following fields and controls:

- Arrival date:** A dropdown menu showing `07 Mars`.
- Departure date:** A dropdown menu showing `12 Mars`.
- Hotel rank:** A dropdown menu showing `***`.
- Room:** A dropdown menu showing `single`.
- Submit:** A button to submit the form.

At the bottom of the browser window, there is a status bar with links: [New Session](#), [Configure](#), [Toggle Halos](#), [Profile](#), [Memory Use](#), and [XHTML](#). Below these links, it says "Go to 'http://validator.w3.org/check/referer'".

Your job: Write the class `RCGetHotelInfo`

2.4 Payment: `RCPayment`

The payment is valid only if 16 number was provided and if the issue date is not over.

The screenshot shows a web browser window with the title bar "/seaside/regconf2". The address bar contains "http://localhost:". The browser has several tabs open: "aal", "Package Universes", "phdcomic", "CyCab", "Radio Ho...", "NEWS | P...", "/seaside...", and "Le Conju...". The main content area displays a registration form with the following fields and controls:

- Cart Number**: A text input field containing the value "1234 1234 1234 1234".
- Issue date**: A text input field containing the value "0205".
- Card type**: A dropdown menu currently showing "mastercard".
- Submit**: A button to submit the form.

At the bottom of the browser window, there is a horizontal menu bar with the following links: [New Session](#), [Configure](#), [Toggle Halos](#), [Profile](#), [Memory Use](#), and [XHTML](#).

Your job: Write the class RCPayment

2.5 Confirmation: RCTConfirmation

Once the payment is done, it is nice to show a summary of what was done.

Your job: Write the class RCTConfirmation