

Making Web Programming Simpler: a Seaside Tutorial

Alexandre Bergel
LERO & DSG, Trinity College Dublin, Ireland
`Alexandre.Bergel@cs.tcd.ie`

November 15, 2006

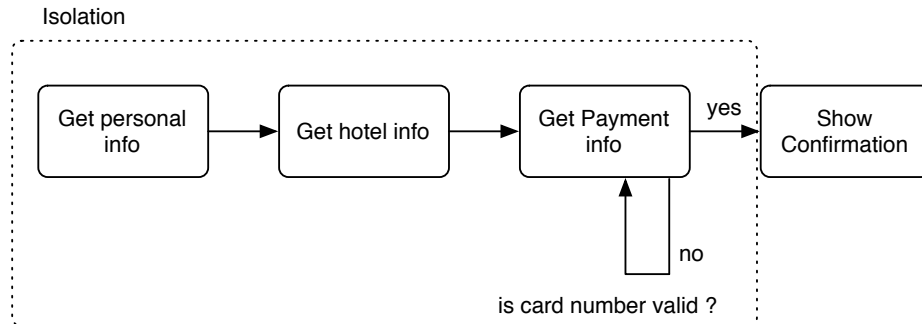
1 RegConf: An Application for Registering to a Conference

The goal of this tutorial is to give you a feeling on creating a web application using Seaside. ConfReg 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: **ConferenceRegistration**

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 `ConferenceRegistration` sounds appropriated.

Your job: Create a task `ConferenceRegistration` with a `go` method that describes the control flow of the application. The code of this method is:

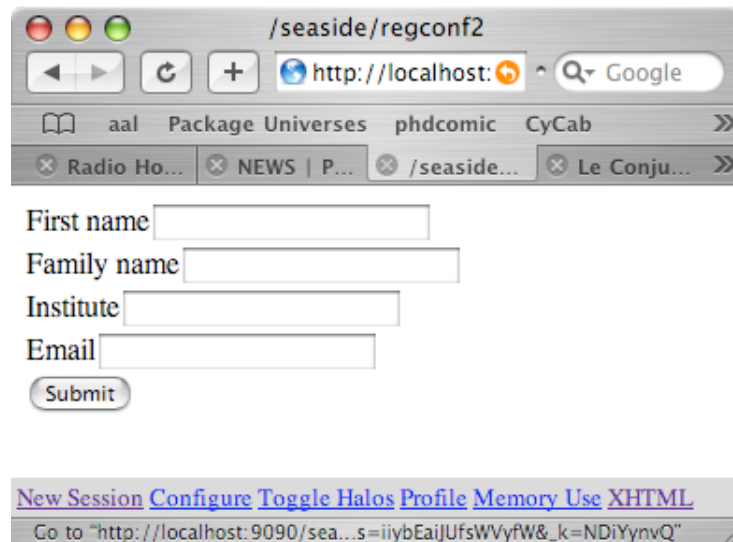
```
go
  |person hotelInfo|
  self isolate: [
    person := self openPersonalInformationEditor.
    hotelInfo := self openHotelInformationEditorFor: person.
    self openPaymentEditorFor: person].
  self displayConfirmationFor: hotelInfo
```

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: **PersonalInformationEditor**

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:



Your job: Write the method `renderContentOn:` in `PersonInformationEditor`. Here is what it should look like:

```
renderContentOn: html
  html form: [
    html text: 'First name' ;
    textInputWithCallback: [:w| firstname := w]; break.
    ...
    html submitButtonWithAction: [self answer: self createUser]
  ].
```

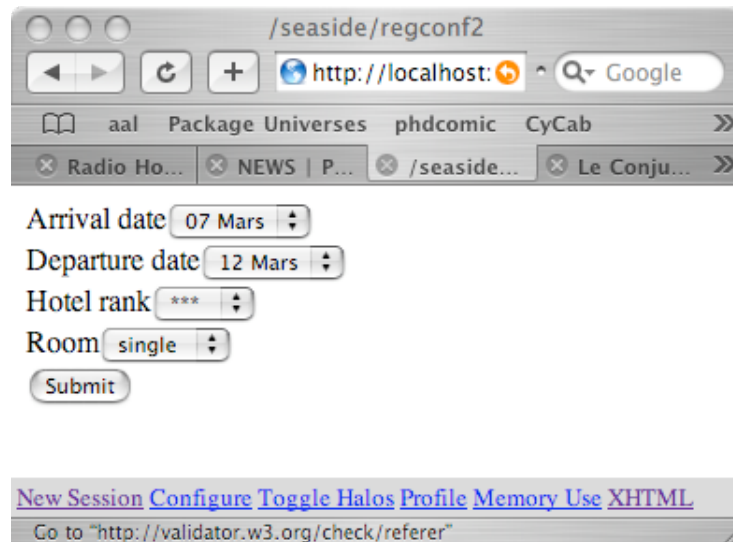
The information passed around different states of the application can be contained in a dictionary. A proper design requires a class `Person` for which an instance is passed around through.

Your job: Create the class `Person` and define the method `createUser`.

Your job: Try your application using your favorite web browser. Make it point to `http://localhost:9090/seaside/regconf2`.

2.3 Getting Hotel Information: `HotelInformationEditor`

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



Your job: Write the class `HotelInformationEditor` . The method `renderContentOn:` should look like:

```
renderContentOn: html
    html form: [
        html text: 'Arrival date' ;
        selectFromList: self dates selected: self dates first
            callback: [:w| arrivalDate := w]; break.
        ...
        html submitButtonWithAction: [self returnHotelInfo]
    ].
```

Your job: Create the class `Hotel` and create the method `returnHotelInfo` .

2.4 Payment: `PaymentInformation`

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:" followed by a search icon and the text "Google". The browser's tab bar shows several tabs: "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 located below the form fields.

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 `PaymentInformation`

2.5 Confirmation: Confirmation

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

Your job: Write the class `CRConfirmation`