sitepoint premium

Handout

Introduction to Ruby 2.0
Lesson 2 - Step 5
Darren Jones

Introduction

Now we're going to use Sinatra to create a web version of the Greeter program that we created earlier. Most of the code will be the same, but the way that we get the input from the user is going to be different.

There are basically two ways of getting user input on the web: from the URL string or from forms. In this example, we're going to use the URL in the address bar to grab the user's name.

Using Sinatra to Create a Web Application

```
require 'sinatra'

get '/:name' do
  name = params[:name]
  "Hello #{name}! Did you know that your name has #{name.length} letters in it and written backwar
end
```

Here is the code that we'll be using. As usual with our web programs, we start by requiring the Sinatra gem in the first line.

Then there's a route handler. The string in this route contains a value that starts with a colon - it is called a **named parameter**. These are special tokens in the route that can have any value. The value gets stored in the `params` hash that acts as a special container holding any values entered by the user. It works in a similar way as a variable. We're going to be covering hashes in much more detail in lesson four.

Testing the Application

Sinatra outputs the last line of a route handler to the browser, so the last string will be displayed by default. Don't forget to boot your webserver by running

```
ruby web_greeter.rb
```

in your terminal.

Next navigate to `http://localhost:4567/your_name_here`.

Displaying the Application in the Browser

So whatever appears after the forward slash (`http://localhost:4567/`) gets placed into the `params` hash. Note that the symbol in the route handler (`'/:name'`) must match the symbol used inside the handler (`params[:name]`).