RUBY WEB APPLICATIONS - CGI PROGRAMMING

Ruby is a general-purpose language; it can't properly be called a *web language* at all. Even so, web applications and web tools in general are among the most common uses of Ruby.

Not only can you write your own SMTP server, FTP daemon, or Web server in Ruby, but you can also use Ruby for more usual tasks such as CGI programming or as a replacement for PHP.

Please spend few minutes with CGI Programming Tutorial for more detail on CGI Programming.

Writing CGI Scripts:

The most basic Ruby CGI script looks like this:

```
#!/usr/bin/ruby

puts "HTTP/1.0 200 OK"
puts "Content-type: text/html\n\n"
puts "<html><body>This is a test</body></html>"
```

If you call this script *test.cgi* and uploaded it to a Unix-based Web hosting provider with the right permissions, you could use it as a CGI script.

For example, if you have the Web site http://www.example.com/ hosted with a Linux Web hosting provider and you upload *test.cgi* to the main directory and give it execute permissions, then visiting http://www.example.com/test.cgi should return an HTML page saying ***This is a test***.

Here when *test.cgi* is requested from a Web browser, the Web server looks for *test.cgi* on the Web site, and then executes it using the Ruby interpreter. The Ruby script returns a basic HTTP header and then returns a basic HTML document.

Using cgi.rb:

Ruby comes with a special library called **cgi** that enables more sophisticated interactions than those with the preceding CGI script.

Let's create a basic CGI script that uses cgi:

```
#!/usr/bin/ruby

require 'cgi'

cgi = CGI.new
puts cgi.header
puts "<html><body>This is a test</body></html>"
```

Here, you created a CGI object and used it to print the header line for you.

Form Processing:

Using class CGI gives you access to HTML query parameters in two ways. Suppose we are given a URL of /cgi-bin/test.cgi?FirstName=Zara&LastName=Ali.

You can access the parameters *FirstName* and *LastName* using CGI#[] directly as follows:

```
#!/usr/bin/ruby

require 'cgi'
cgi = CGI.new
cgi['FirstName'] # =>  ["Zara"]
cgi['LastName']  # =>  ["Ali"]
```

There is another way to access these form variables. This code will give you a hash of all the key and values:

```
#!/usr/bin/ruby

require 'cgi'
cgi = CGI.new
h = cgi.params  # =>  {"FirstName"=>["Zara"],"LastName"=>["Ali"]}
h['FirstName']  # =>  ["Zara"]
h['LastName']   # =>  ["Ali"]
```

Following is the code to retrieve all the keys:

```
#!/usr/bin/ruby

require 'cgi'
cgi = CGI.new
cgi.keys         # =>  ["FirstName", "LastName"]
```

If a form contains multiple fields with the same name, the corresponding values will be returned to the script as an array. The [] accessor returns just the first of these.index the result of the params method to get them all.

In this example, assume the form has three fields called "name" and we entered three names "Zara", "Huma" and "Nuha":

```
#!/usr/bin/ruby

require 'cgi'
cgi = CGI.new
cgi['name']        # => "Zara"
cgi.params['name'] # => ["Zara", "Huma", "Nuha"]
cgi.keys           # => ["name"]
cgi.params         # => {"name"=>["Zara", "Huma", "Nuha"]}
```

Note: Ruby will take care of GET and POST methods automatically. There is no separate treatment for these two different methods.

An associated, but basic, form that could send the correct data would have HTML code like so:

```
<html>
<body>
<form method="POST" action="http://www.example.com/test.cgi">
First Name :<input type="text" name="FirstName" value="" />
<br />
Last Name :<input type="text" name="LastName" value="" />

<input type="submit" value="Submit Data" />
</form>
</body>
</html>
```

Creating Forms and HTML:

CGI contains a huge number of methods used to create HTML. You will find one method per tag. In order to enable these methods, you must create a CGI object by calling CGI.new.

To make tag nesting easier, these methods take their content as code blocks. The code blocks should return a *String*, which will be used as the content for the tag. For example:

```
#!/usr/bin/ruby

require "cgi"
cgi = CGI.new("html4")
cgi.out{
   cgi.html{
      cgi.head{ "\n"+cgi.title{"This Is a Test"} } +
      cgi.body{ "\n"+
         cgi.form{"\n"+
            cgi.hr +
            cgi.h1 { "A Form: " } + "\n"+
            cgi.textarea("get_text") +"\n"+
            cgi.br +
            cgi.submit
         }
      }
   }
}
```

NOTE: The *form* method of the CGI class can accept a method parameter, which will set the HTTP method $GET, POST, and so on\ldots$ to be used on form submittal. The default, used in this example, is POST.

This will produce the following result:

```
Content-Type: text/html
Content-Length: 302
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Final//EN">
<HTML>
<HEAD>
<TITLE>This Is a Test</TITLE>
</HEAD>
<BODY>
<FORM METHOD="post" ENCTYPE="application/x-www-form-urlencoded">
<HR>
<H1>A Form: </H1>
<TEXTAREA COLS="70" NAME="get_text" ROWS="10"></TEXTAREA>
<BR>
<INPUT TYPE="submit">
</FORM>
</BODY>
</HTML>
```

Quoting Strings:

When dealing with URLs and HTML code, you must be careful to quote certain characters. For instance, a slash character / has special meaning in a URL, so it must be **escaped** if it's not part of the pathname.

For example, any / in the query portion of the URL will be translated to the string %2F and must be translated back to a / for you to use it. Space and ampersand are also special characters. To handle this, CGI provides the routines **CGI.escape** and **CGI.unescape**.

```
#!/usr/bin/ruby

require 'cgi'
puts CGI.escape(Zara Ali/A Sweet & Sour Girl")
```

This will produce the following result:

```
Zara+Ali%2FA Sweet+%26+Sour+Girl")
```

```
#!/usr/bin/ruby

require 'cgi'
puts CGI.escapeHTML('<h1>Zara Ali/A Sweet & Sour Girl</h1>')
```

This will produce the following result:

```
&lt;h1&gt;Zara Ali/A Sweet & Sour Girl&lt;/h1&gt;'
```

Useful Methods in CGI Class:

Here is complete listing of all the methods related to CGI class:

- The Ruby CGI - Methods related to Standard CGI library.

Cookies and Sessions:

I have explained these two concepts in different links. Please follow these links:

- The Ruby CGI Cookies - How to handle CGI Cookies.
- The Ruby CGI Sessions - How to manage CGI sessions.

Web Hosting Servers:

You could check following links to host yoru website on a Unix-based Server:

- Unix-based Web hosting

Loading [MathJax]/jax/output/HTML-CSS/jax.js