Mitchell Wright

Web Applications

Take-Home Midterm

**Architecture**

1. What is the difference between an HTTP PUT request and a POST request?

The main difference between HTTP PUT and POST is that POST is used to create and PUT is used to create or update. In PUT, adding the same object twice has no effect.

You see a link in an interface whose markup is as follows:

<a href="images/new?request\_type=PUT" method="POST">create a new image</a>

1. Is the target URL relative or absolute?

Relative.

1. What is the difference between an absolute and relative URL?

An absolute URL is based from the system root and a relative URL is based from the file containing the URL.

1. If you clicked on this link, what kind of request would your browser generate? (which HTTP method?) Assume no JavaScript modifies the behavior of the link.

This link would generate a POST request.

1. Is there a querystring? What is it?

Yes, there is a querystring. Its value is “request\_type=PUT”

1. What is lacking from the link declaration that would otherwise enhance accessibility?

A tooltip could be set by setting the ‘title’ attribute of the link.

1. What are the roles of the database and Web browser in most Web applications? (One sentence for each.)

The database serves as a repository of volatile information needed for the application. (Information that is likely to change).

The web browser receives and shows HTML documents from the server, allowing the client to interact with the application.

Given the following HTTP response header:

HTTP/1.1 200 OK  
Date: Wed, 09 Mar 2011 16:43:33 GMT  
Server: Apache  
Connection: Keep-Alive  
Keep-Alive: timeout=2, max=100  
Etag: "110e412f-7df-49e0f6a106500"  
Vary: Accept-Encoding

1. Would an HTTP response that begins like that usually contain a body? Why or why not?

HTML status code 200 is a ‘success’ code. By definition, this response would contain the requested resource in its body.

**Ruby**

1. Write a Ruby class definition that meets the following criteria:
   * class is called Troll
   * class has publicly accessible attributes ugliness, smelliness, and strength
   * upon instantiation, an object of this class has a member variable, a String, called grunt, whose initial value is "UNGAH" (that's pronounce "oon-guh").
   * class has an instance method called speak() that prints the value of the instance variable grunt 42 times
   * class has an instance method called reverse() that prints the value of the instance variable grunt backwards
   * class has a static/class method called propogate(), which returns a Troll instance whose grunt attribute is "eegah"

class Troll

attr\_accessor :ugliness, :smelliness, :strength

def initialize( grunt = “UNGAH” )

@grunt = grunt

end

def speak

42.times { puts @grunt }

end

def reverse

puts @grunt.reverse

end

def self.propogate

Troll.new(“eegah”)

end

end

1. Imagine a Troll instance fred, which, when the following method is called:

fred.respond\_to?("fight")

returns true. What is missing from your class definition in order for this example to be accurate?

We would need to define a “fight” method.

1. Does the respond\_to?() method illustrate object-oriented polymorphism? If so, in what manner?

Yes. respond\_to? is inherited from the Object class.

1. According to Ruby conventions, what kind of value would you expect to receive from a method that ends in a question mark (?) ?

A boolean! True or false! 0 or anything else!

1. According to Ruby conventions, what is the difference between pairs of methods like do\_this and do\_this! (notice the bang) ?

Method names with an exclamation point (the bang) indicate mutator methods and operations that change the state of the object.

1. Briefly explain Ruby's type system. What is it (by name)? What does it mean?

Ruby has a dynamic typing system, particularly a subset called ‘duck’ typing. Ruby will infer type based on an object’s methods and properties.

1. What type of Ruby object does the following expression yield? %w( master rails and then try another framework you'll never go back)

This expression yields an Array of double-quoted string literals.

1. Given an array of strings called @happy\_places, would these two snippets of code do the same thing?

@happy\_places.each do |happy\_place|  
  puts happy\_place  
end

and

@happy\_places.each {|hp| puts hp}

**YES**

1. Given a function that needs to return a value to its caller, does the function need an explicit return statement? If so, explain why. If not, then what can you always expect a Ruby function to return?

No. A method in Ruby will return the value of the last expression evaluated in that method.

**Rails**

1. Name four ActiveRecord callbacks that you can bind methods to.

Create, update, delete, validate

1. The Rails convention maps HTTP methods to certain controller methods, and those methods usually involve specific CRUD operations on models. Given the following CRUD database methods:  
   create, read, update, and delete  
   and the following HTTP methods:  
   GET, PUT, POST, DELETE  
   and the following controller actions:  
   index, new, create, edit, update, destroy Complete the following table.

|  |  |  |
| --- | --- | --- |
| **HTTP method** | **controller action** | **CRUD operation** |
| GET | index | read |
| GET | new | read |
| POST | create | update |
| GET | edit | read |
| PUT | update | update |
| DELETE | delete | delete |

1. Rails "simulates" PUT and DELETE requests. Why?

Browsers don’t support PUT and DELETE requests, so Rails has to simulate them.

1. What is the difference between the two Rails environments 'production' and 'development' ?

In the development environment, code is re-interpreted on every refresh of a page. In production, this does not happen. This allows developers to immediately see changes in code without restarting the server.

1. Usually, Rails controllers incorporate plural nouns, such as ProtestsController and RevolutionsController. In what case should a controller have a singular name like GeocodingController?

A controller should have a singular name when the model is a singleton object.

1. What is a Rails "helper method" and when should they be defined and used by you, the developer?

A Rails “helper method” encapsulates behavior and is often used to keep functionality out of the view and reduce repeated code.

Assume you have a Flower AR class that has\_and\_belongs\_to\_many :bees, and a Bee class that has\_and\_belongs\_to\_many :flowers.

1. What must exist in the database schema in order for AR to infer the proper foreign key / relationship?

A join table named “bees\_flowers” must be in the database schema. Otherwise :through => <table\_name> must be used.

Assume that a Bee :belongs\_to a Hive and a Hive has\_many :bees. Also assume a GET request is sent to the FlowersController#show action, which contains a finder method call @flower = Flower.find(params[:id]. Assume the view app/views/flowers/show.html.haml displays the name of the Flower and each Flower's bee's name and Hive name like so:

- @flower.bees.each do |b|  
  %h1= b.name  
  %p= b.hive.name

If you were tailing the log of your application during the rendering of the response, you would notice tons of database queries.

1. Are all of those queries ok? If so, explain why. If not, explain how you would reduce the number of database queries (without hand-rolling your own SQL query).

I would reduce the number of database queries by using eager loading. In the controller, I would add the following in the show action for Flower:

@bees\_for\_flowers = Bee.find\_all\_by\_flower( params[:id], :include => :hive )