Tyson Williams

Web Applications Midterm

1. The fundamental difference between the POST and PUT requests is reflected in the different meaning of the Request-URI. The URI in a POST request identifies the resource that will handle the enclosed entity. That resource might be a data-accepting process, a gateway to some other protocol, or a separate entity that accepts annotations. In contrast, the URI in a PUT request identifies the entity enclosed with the request -- the user agent knows what URI is intended and the server MUST NOT attempt to apply the request to some other resource.
2. Relative
3. An absolute URL tells the calling file exactly where the requested file will be, a relative URL’s path is relative to the calling file, so a relative path relies on the path of the calling file.
4. It generates a POST request
5. Yes it contains a Querystring, a querystring is the part of the url that is after the question mark, so request\_type=PUT is the querystring.
6. The link doesn’t have a tool tip, which would increase accessibility.
7. The database stores information required for the web page, in an online store for example it would hold all the customer information, the product information and the supplier information. The web browser interprets the code for the server and displays it for the user. It acts as the interface for the user to interact with the webpage.
8. No, the header doesn’t contain any of the content response headers (content-length, content-type, ect)
9. See end of file for code
10. There needs to be a method called fight for this to return true for the troll class
11. Yes, every object class will respond to a call by the name responds\_to, so there must be some polymorphism in the call.
12. We would expect a Boolean value (true or false)
13. The bang method affects the actual object it is called on, where the non-bang method would return a new object.
14. Ruby is dynamically typed, which means most of its type checking is done at runtime rather than compile time, and a variable can refer to any type.
15. It returns an array of strings broken up on the words by spaces.
16. Yes they are equivalent, one is just a short hand version
17. No, the return value is not required unless you are stopping a method early. Ruby returns the return value of the final line in a method by default.
18. before\_validation, after\_validation, before\_save, before\_create

|  |  |  |
| --- | --- | --- |
| HTTP METHOD | Controller action | CRUD operation |
| get | Index | read |
| get | new | read |
| post | create | create |
| get | edit | read |
| put | update | update |
| delete | delete | delete |

1. This is to accommodate lack of browser support for put and delete
2. The two environments have several differences, all of which can be found in the config/environments/production.rb and config/environments/development.rb files. The key differences are code is not reloaded in production, debug level log messages are ignored in prod, caching is only on in production, and error messages are more verbose in dev.
3. The plural names are for when the controller is controlling multiple objects, such as protests and revolutions, we would use a singular name if there was a singular object Geoencoding
4. Rails helper methods are used to keep code from repeating, and keeping code DRY, you can define these as a developer when you have chunks of code that repeat often.
5. The dual has\_and\_belongs\_to\_many requires a join table of the two element that has no corresponding model or primary key for the foreign key model to work.
6. We would really like to reduce the number of queries to the database to reduce traffic, so the number of queries on this kind of thing would be unacceptable. We can reduce the number of queries to the database by using the include tag in our find from above to include the bee name and the hive name in the first query, thereby reducing the queries dramatically.

Here is the code for the troll class:

class troll

attr\_accessor :grunt, :ugliness, :smelliness, :strength

def initialize

@grunt ="UNGAH"

end

def ugliness

@ugliness

end

def smelliness

@smelliness

end

def strength

@strength

end

def speak

(1..42).each { |i| puts @grunt }

end

def reverse

puts @grunt.reverse!

end

def self.propogate

strangetroll = troll.new

strangetroll.grunt = "eegah"

return strangetroll

end