and

get

Groovy; what's that?

- Scripting language based on Java syntax that compiles into bytecode for the same JVM Java runs on
- Similar to Ruby and Python
- Adds closures, dynamic method support, makes semi-colons optional
- Has JSR legitimacy

Grails; where's Indiana Jones?

- Modeled after Rails, follows many of the same conventions, but runs in a Servlet container with just one additional jar file.

- But build on existing Java technology: Spring, Hibernate, Sitemesh, prototype, ant, hsqldb (no not-invented-here-syndrome)

- Has an excellent plugin ecosystem, plugins for most popular Java libraries: Acegi, Jasper, terracotta, Axis2, YUI/ext/jquery and lots

Back to RESTful

- Some code examples from Scott Davis's developer works article. Nice guy, VP of Boulder Java Users Group.

- http://www.ibm.com/developerworks/library/j-grails09168/

get RESTful

- add “import grails.converters.*” to top of your controller class
- remember your url will map based on your controller name; so ClassController will map to http://localhost:8080/<app name>/class/

Tweak grails routing

```
class UrlMappings {
    static mappings = {
      "/class/$callNumber?"(controller:"class",action:"index")
      "/$controller/$action?/$id?"{
          constraints {
             // apply constraints here
          }
      }
      "500"(view:'/error')
    }
}
```

Tell controller to Accept put/post/delete

```
def index = {
      switch(request.method){
        case "POST":
          render "Create\n"
          break
        case "GET":
          render "Retrieve\n"
          break
        case "PUT":
          render "Update\n"
          break
        case "DELETE":
          render "Delete\n"
          break
    }
}
```

get

```
def get(params) {
        if (params.callNumber) {
            render CuClass.findByCallNumber(params.callNumber) as XML
        }
        else {
            render CuClass.list() as XML
        }
    }
```

put

```
def put(params) {
        def cuClass = new CuClass(params.cuClass)
          if(cuClass.save()){
            response.status = 201 // Created
            render cuClass as XML
          }
          else{
            response.status = 500 //Internal Server Error
            render "Could not create new CuClass due to errors:\n $
{cuClass.errors}"
          }
    }
```

post

```
def post(params) {
        def cuClass = CuClass.findByCallNumber(params.cuClass.callNumber)
        cuClass.properties = params.cuClass
        if(cuClass.save()){
            response.status = 200 // OK
            render cuClass as XML
        }
        else{
            response.status = 500 //Internal Server Error
            render "Could not update CuClass due to errors:\n ${cuClass.errors}"
        }
    }
```

delete

```
def delete(params) {
        if(params.callNumber){
            def cuClass = CuClass.findByCallNumber(params.callNumber)
            if(cuClass){
             cuClass.delete()
              render "Successfully Deleted."
            }
            else{
              response.status = 404 //Not Found
              render "${params.callNumber} not found.\n"
            }
        }
        else{
            response.status = 400 //Bad Request
            render "DELETE request must include the Call Number\n Example: /class/
CSCI7818 \n"
        }
    }
```

Tips

- new lines in XML are very bad, put everything on the same line

- grails can be a cruel mistress, I don't like its error reporting, but rails isn't any better. Django experience?

- curl reports content size wrong and the server doesn't read all of the XML