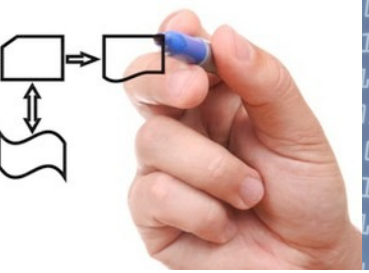
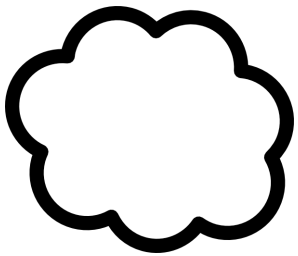
San Jose State University

CMPE 281

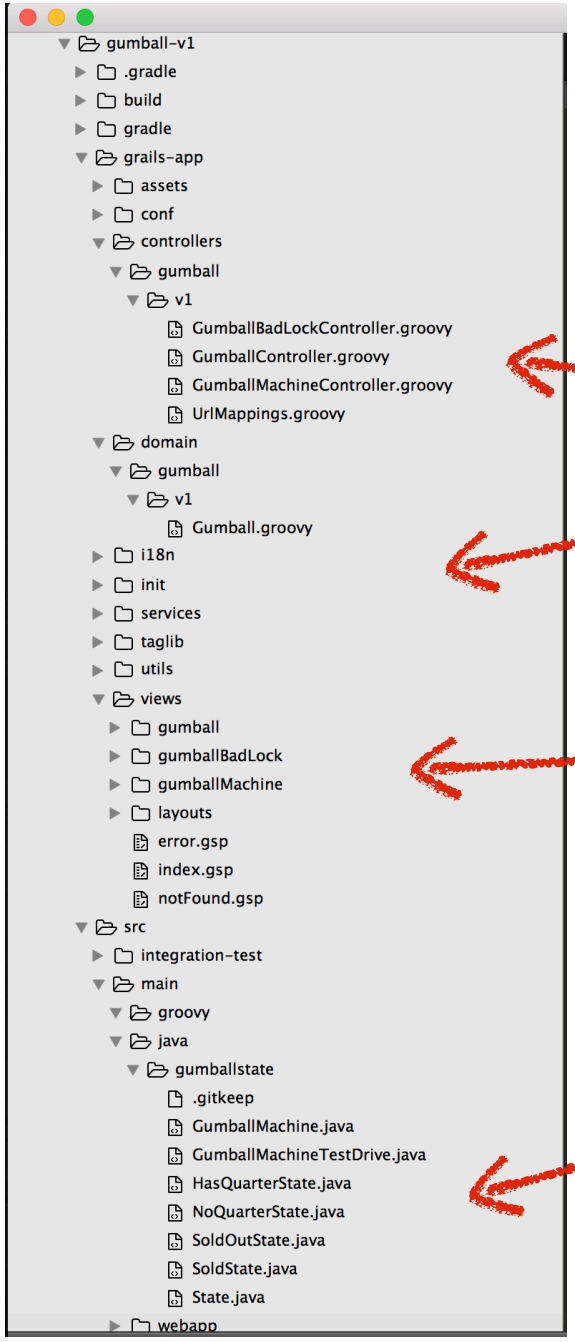
Grails Gumball Lab





Gumball Machine (v1)

Grails Gumball Machine



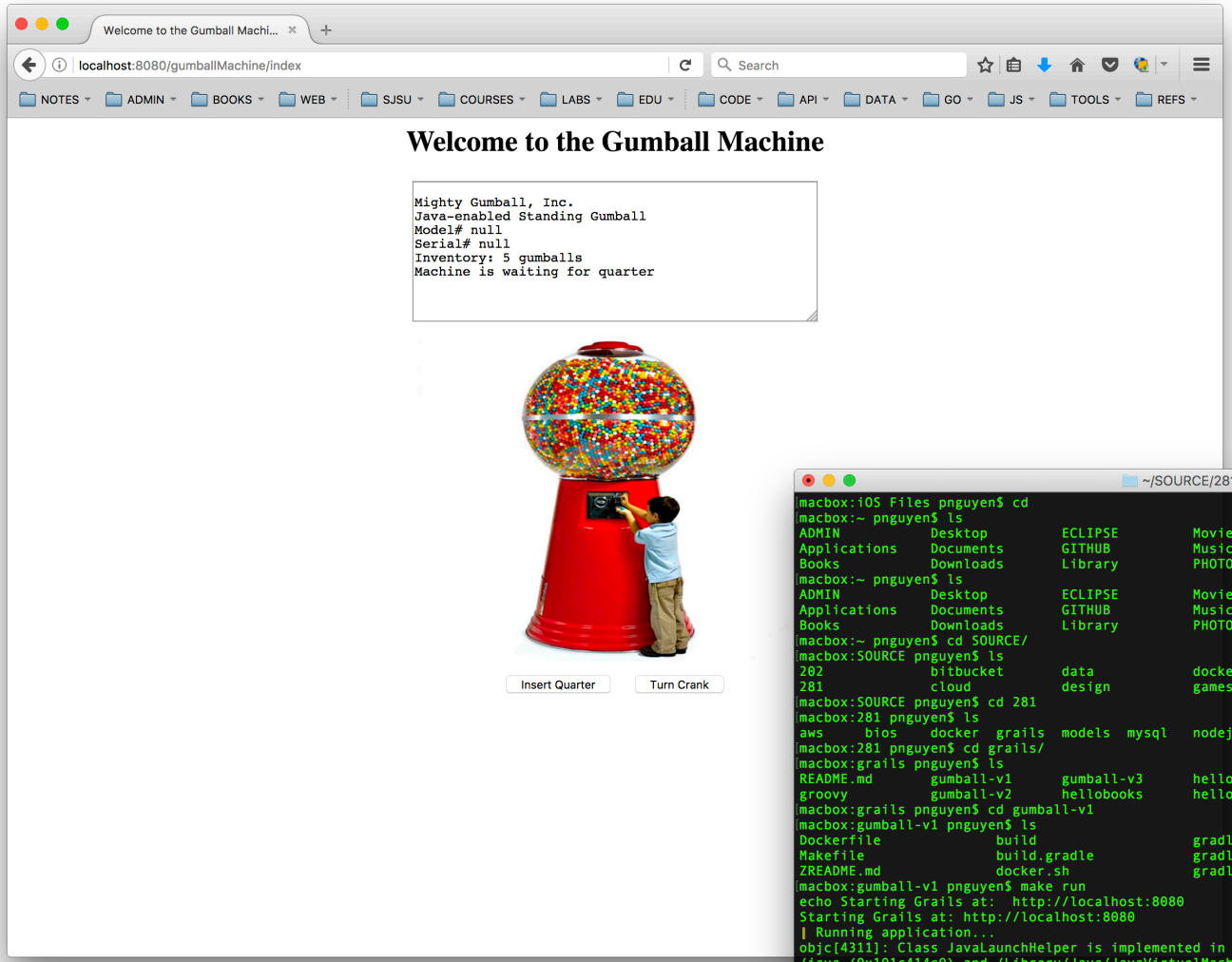
Controllers

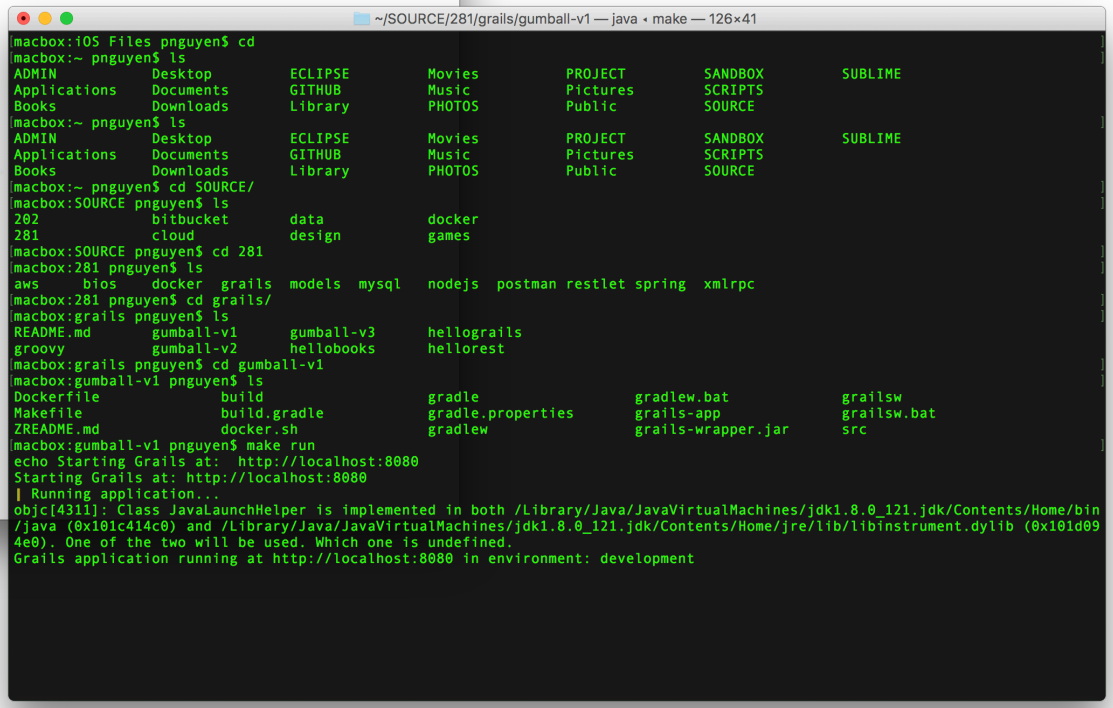
Domain Model

Views

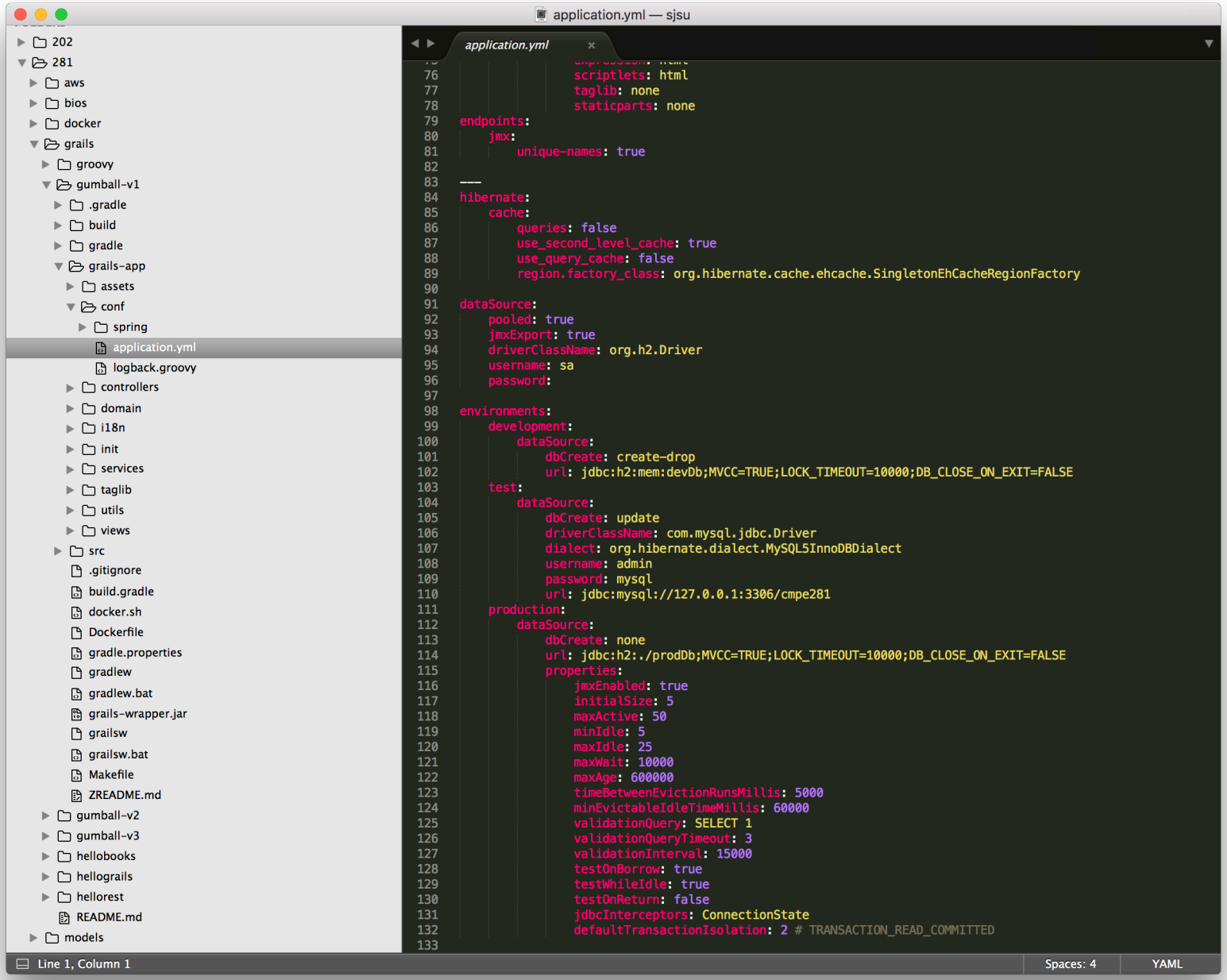
Java State Machine Code for Gumball

Running Locally

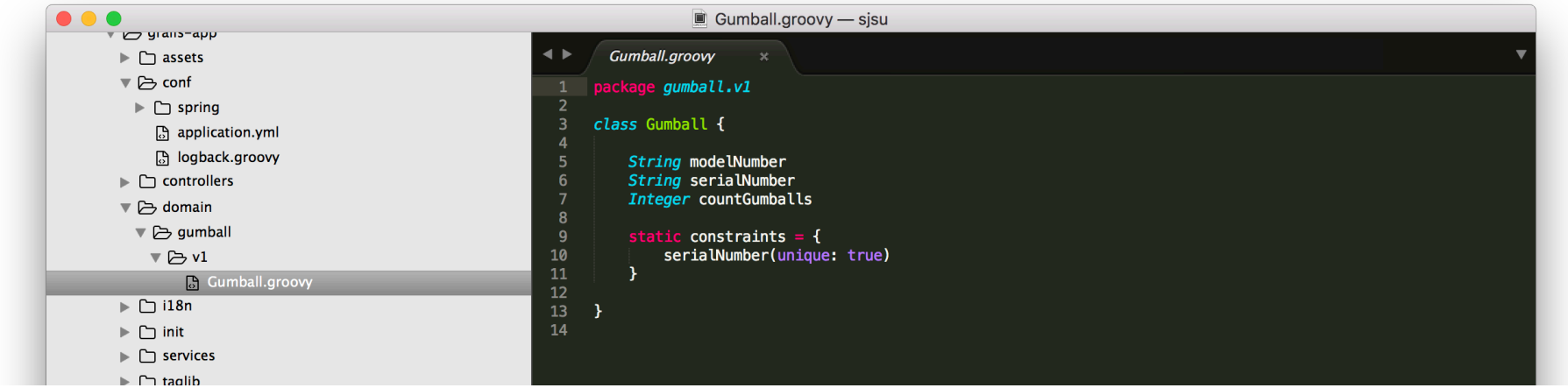


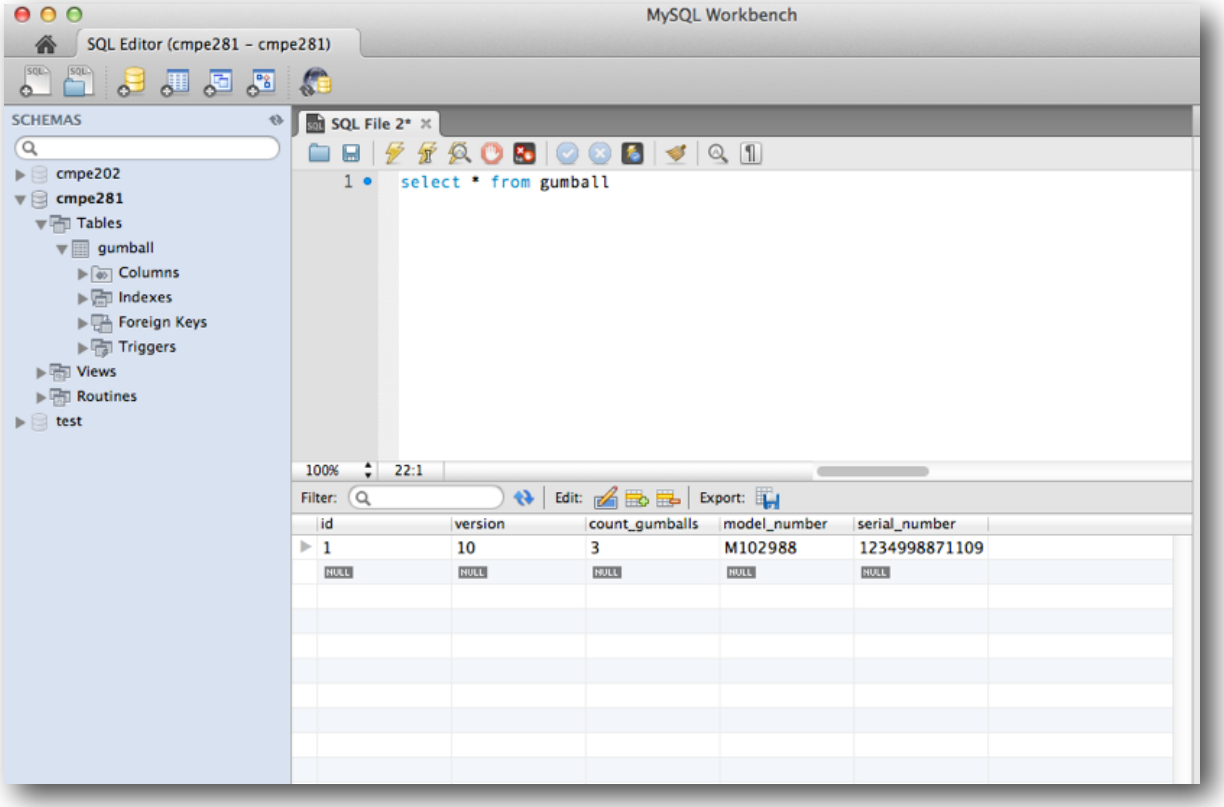


MySQL Config

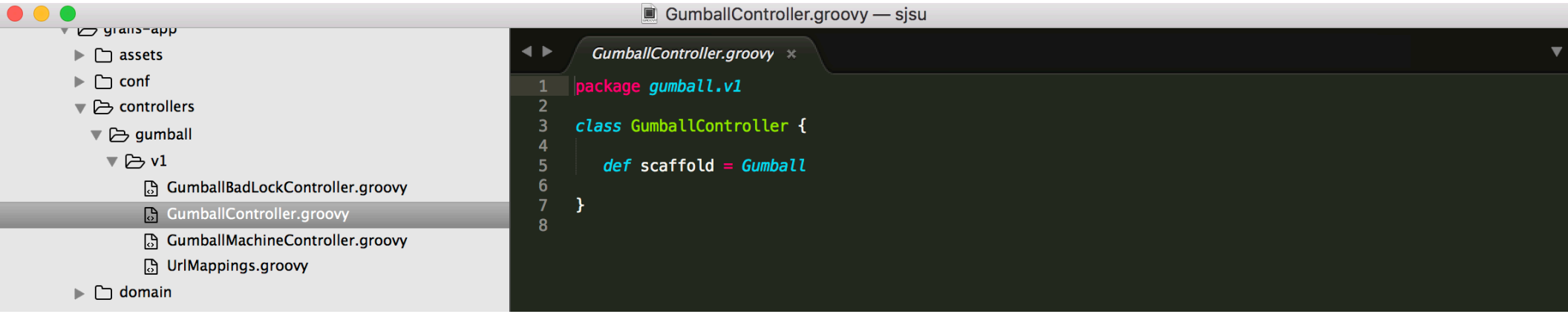


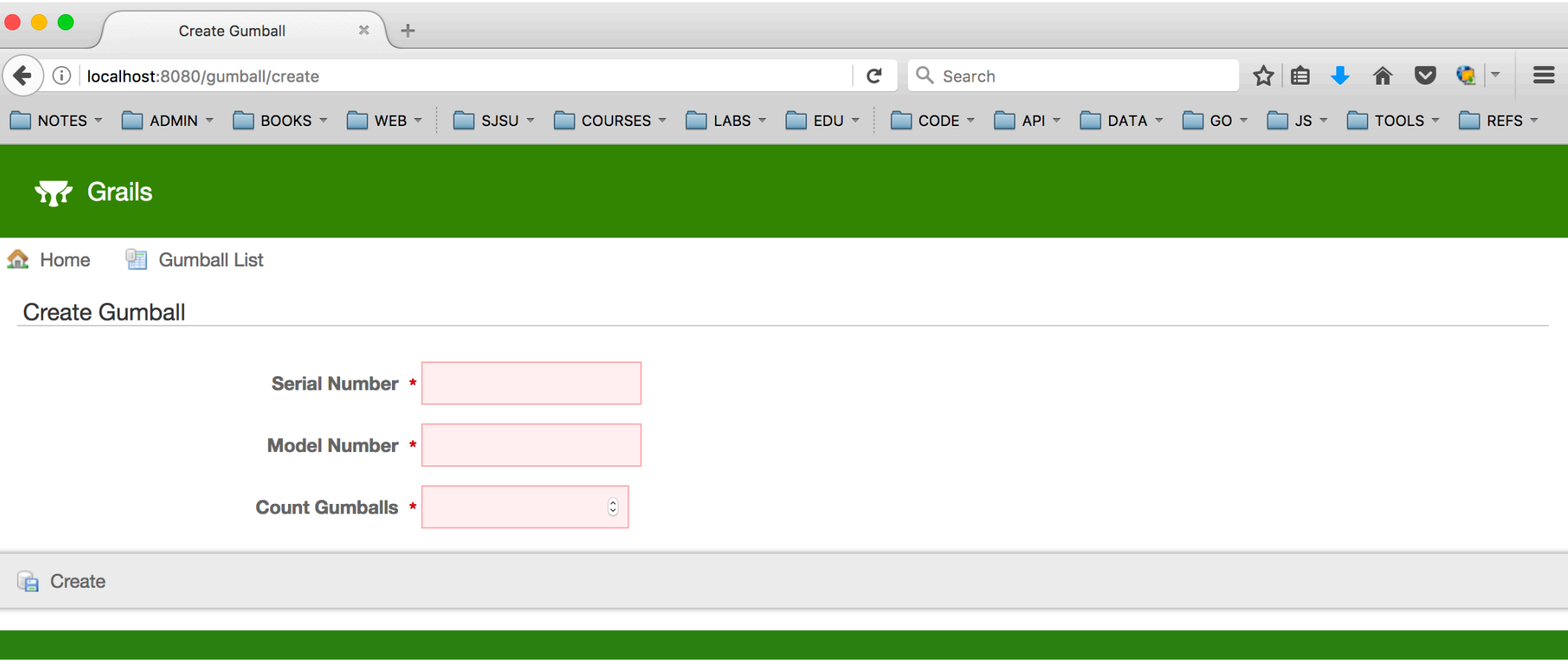
Domain Model





Domain Model Scaffolding



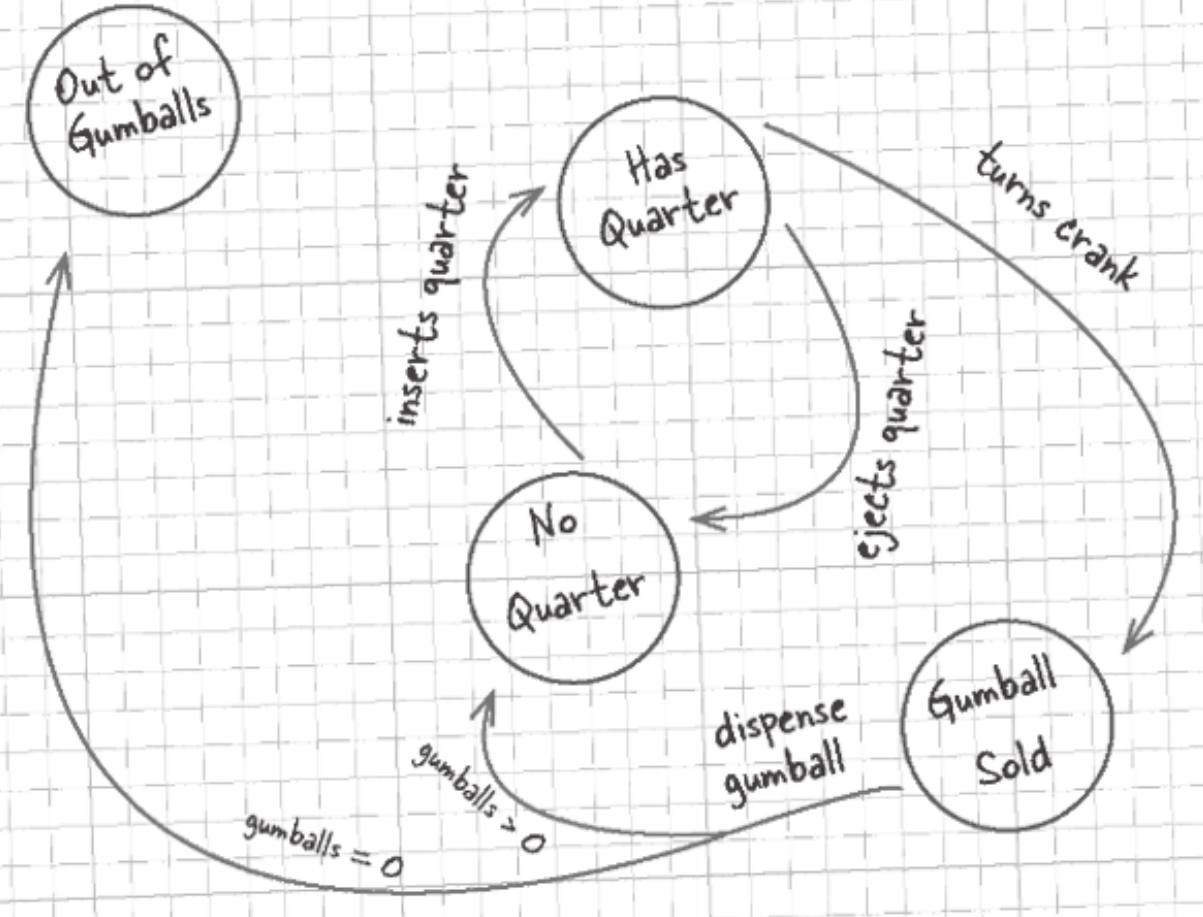


Mighty Gumball, Inc.

Where the Gumball Machine is Never Half Empty

Here's the way we think the gumball machine controller needs to work. We're hoping you can implement this in Java for us! We may be adding more behavior in the future, so you need to keep the design as flexible and maintainable as possible!

– Mighty Gumball Engineers



Java State Machine

First we need to implement the State interface.

We get passed a reference to the Gumball Machine through the constructor. We're just going to stash this in an instance variable.

```
public class NoQuarterState implements State {
    GumballMachine gumballMachine;

    public NoQuarterState(GumballMachine gumballMachine) {
        this.gumballMachine = gumballMachine;
    }

    public void insertQuarter() {
        System.out.println("You inserted a quarter");
        gumballMachine.setState(gumballMachine.getHasQuarterState());
    }

    public void ejectQuarter() {
        System.out.println("You haven't inserted a quarter");
    }

    public void turnCrank() {
        System.out.println("You turned, but there's no quarter");
     }

    public void dispense() {
        System.out.println("You need to pay first");
    }
}
```

If someone inserts a quarter, we print a message saying the quarter was accepted and then change the machine's state to the HasQuarterState.

You'll see how these work in just a sec...

You can't get money back if you never gave it to us!

And, you can't get a gumball if you don't pay us.

We can't be dispensing gumballs without payment.

Java State Machine

```
public class GumballMachine {

    State soldOutState;
    State noQuarterState;
    State hasQuarterState;
    State soldState;

    State state = soldOutState;
    int count = 0;

    public GumballMachine(int numberGumballs) {
        soldOutState = new SoldOutState(this);
        noQuarterState = new NoQuarterState(this);
        hasQuarterState = new HasQuarterState(this);
        soldState = new SoldState(this);

        this.count = numberGumballs;
        if (numberGumballs > 0) {
            state = noQuarterState;
        }
    }

    public void insertQuarter() {
        state.insertQuarter();
    }

    public void ejectQuarter() {
        state.ejectQuarter();
    }

    public void turnCrank() {
        state.turnCrank();
        state.dispense();
    }

    void setState(State state) {
        this.state = state;
    }
```

Here are all the States again...

...and the State instance variable.

The count instance variable holds the count of gumballs – initially the machine is empty.

Our constructor takes the initial number of gumballs and stores it in an instance variable.

It also creates the State instances, one of each.

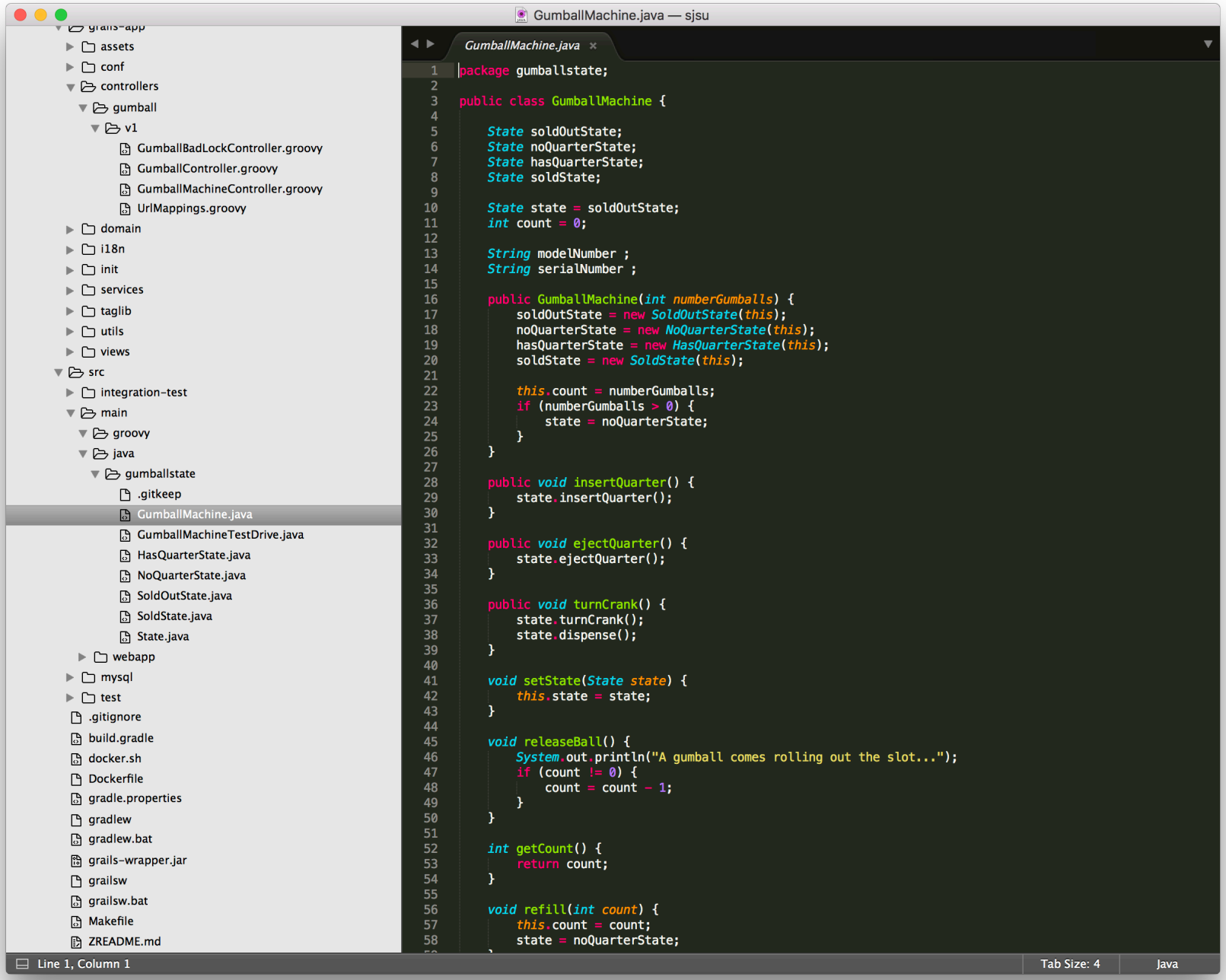
If there are more than 0 gumballs we set the state to the NoQuarterState.

Now for the actions. These are VERY EASY to implement now. We just delegate to the current state.

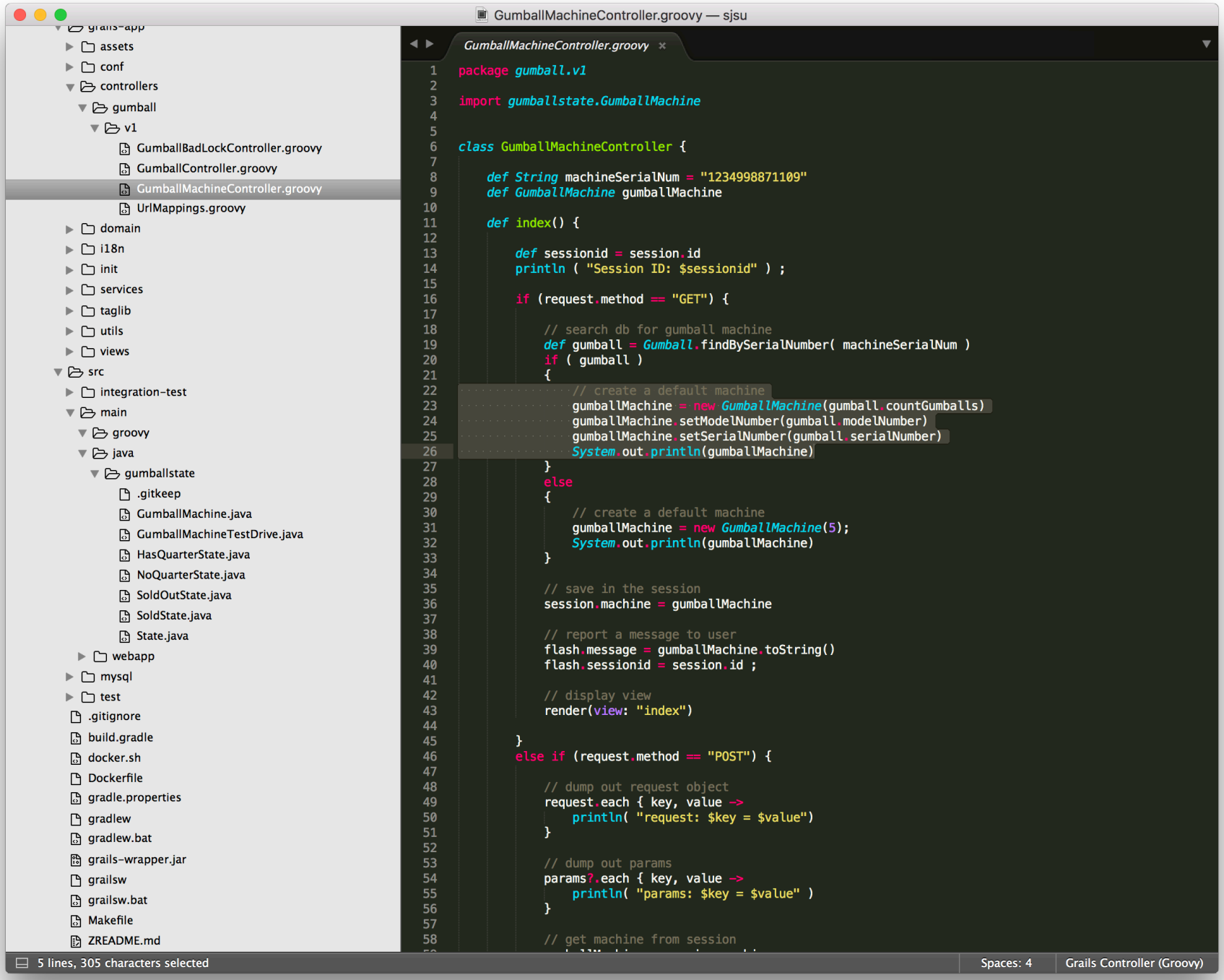
Note that we don't need an action method for dispense() in GumballMachine because it's just an internal action; a user can't ask the machine to dispense directly. But we do call dispense() on the State object from the turnCrank() method.

This method allows other objects (like our State objects) to transition the machine to a different state.

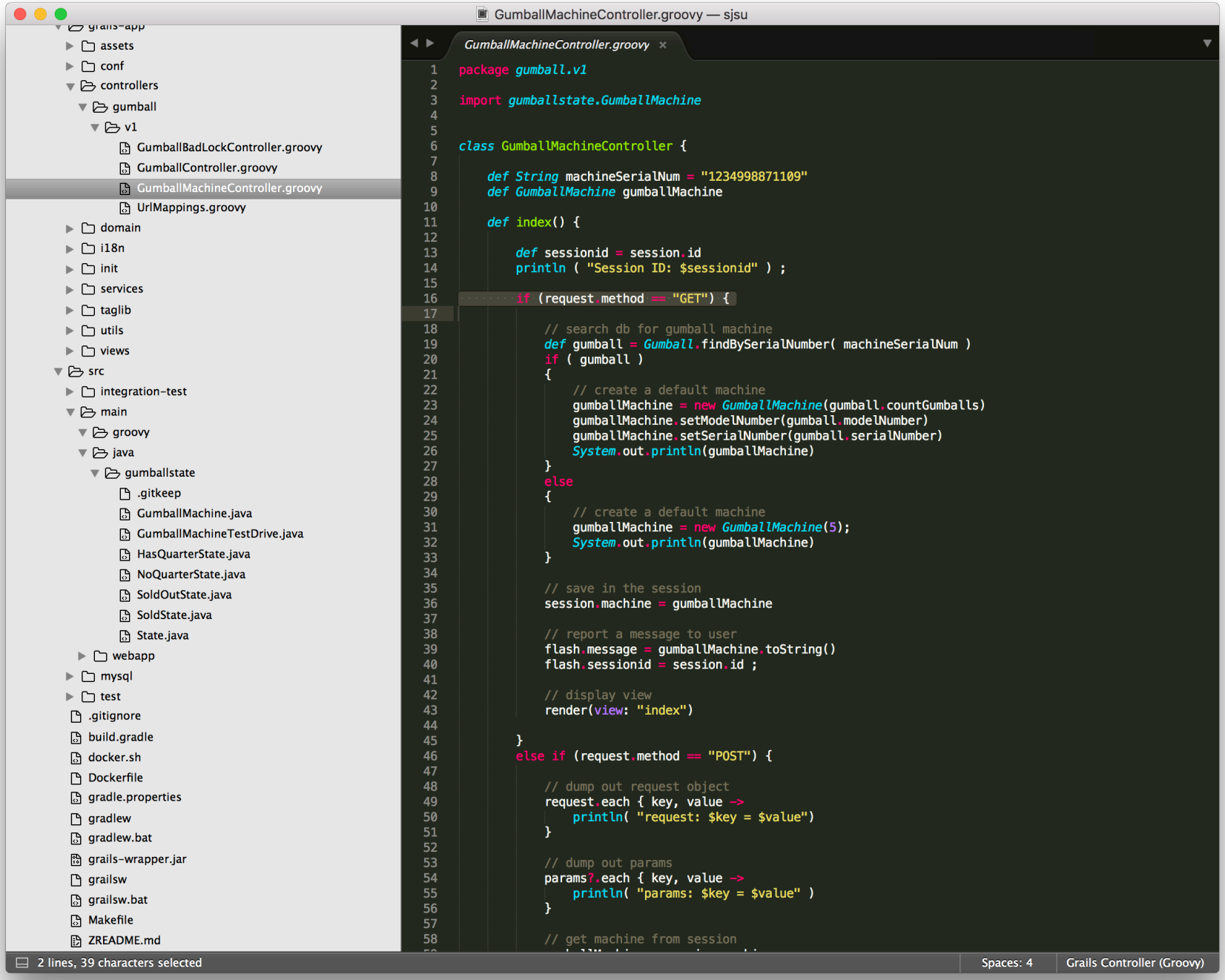
Groovy calling Java



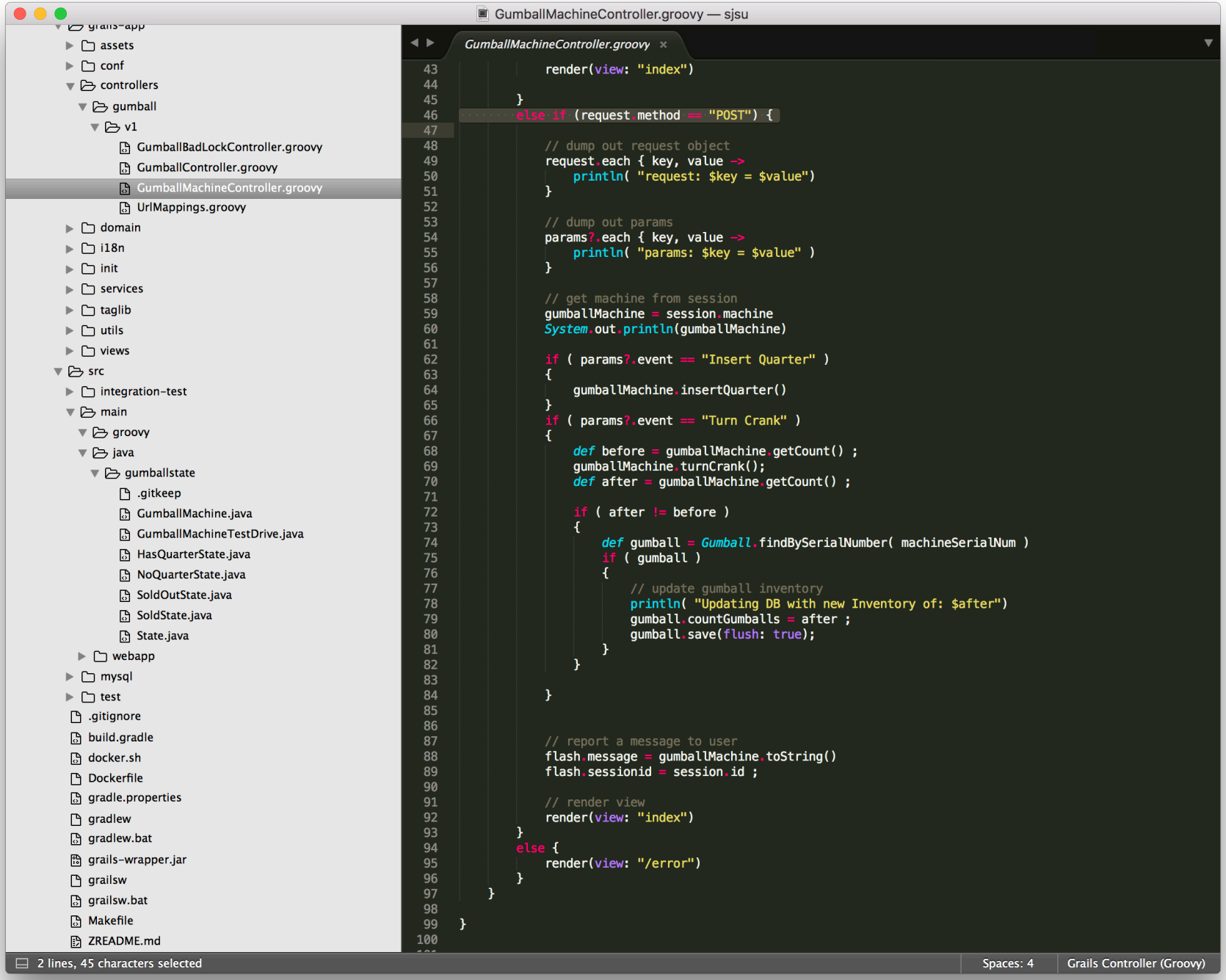
Groovy calling Java



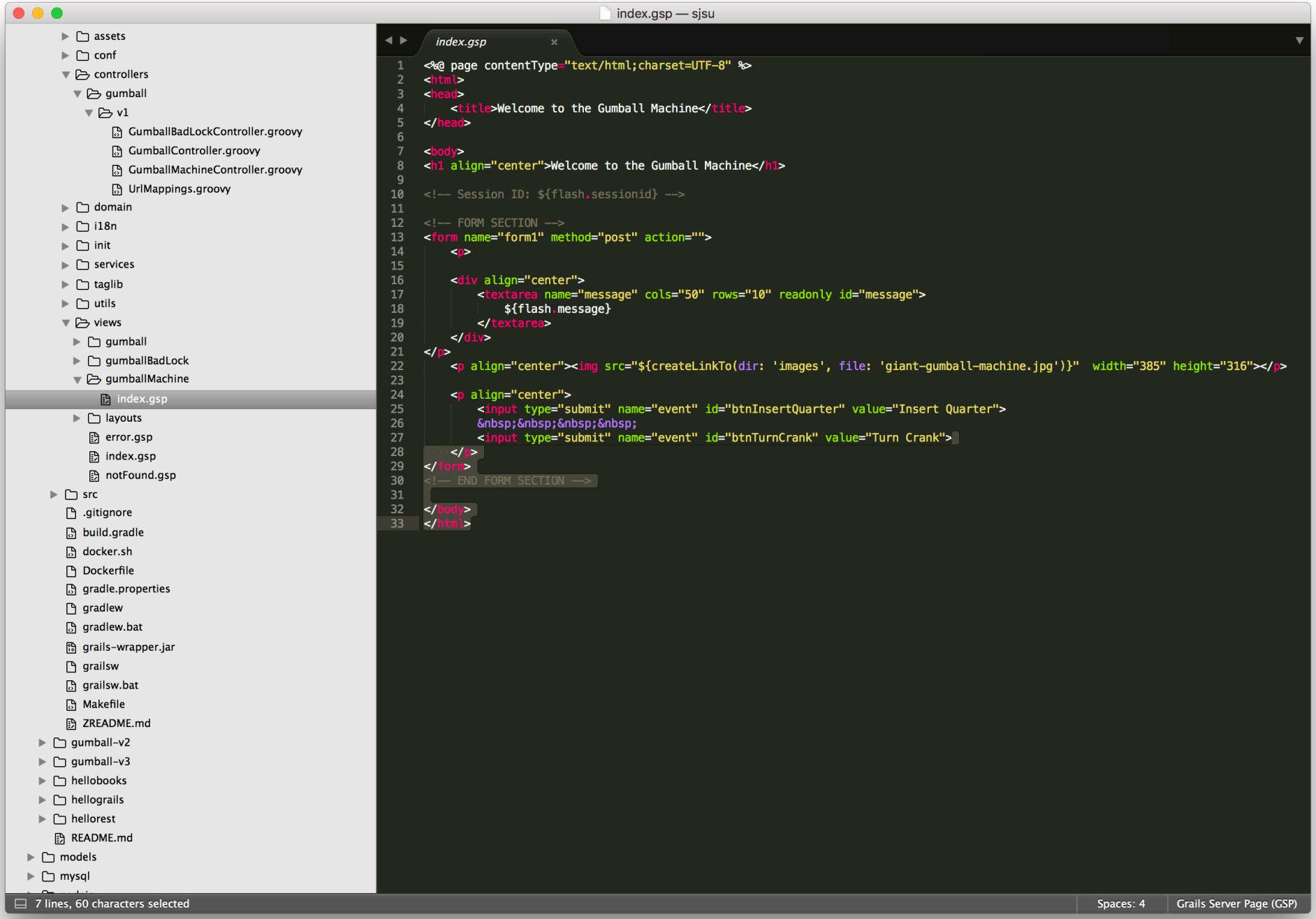
Gumball Machine Controller



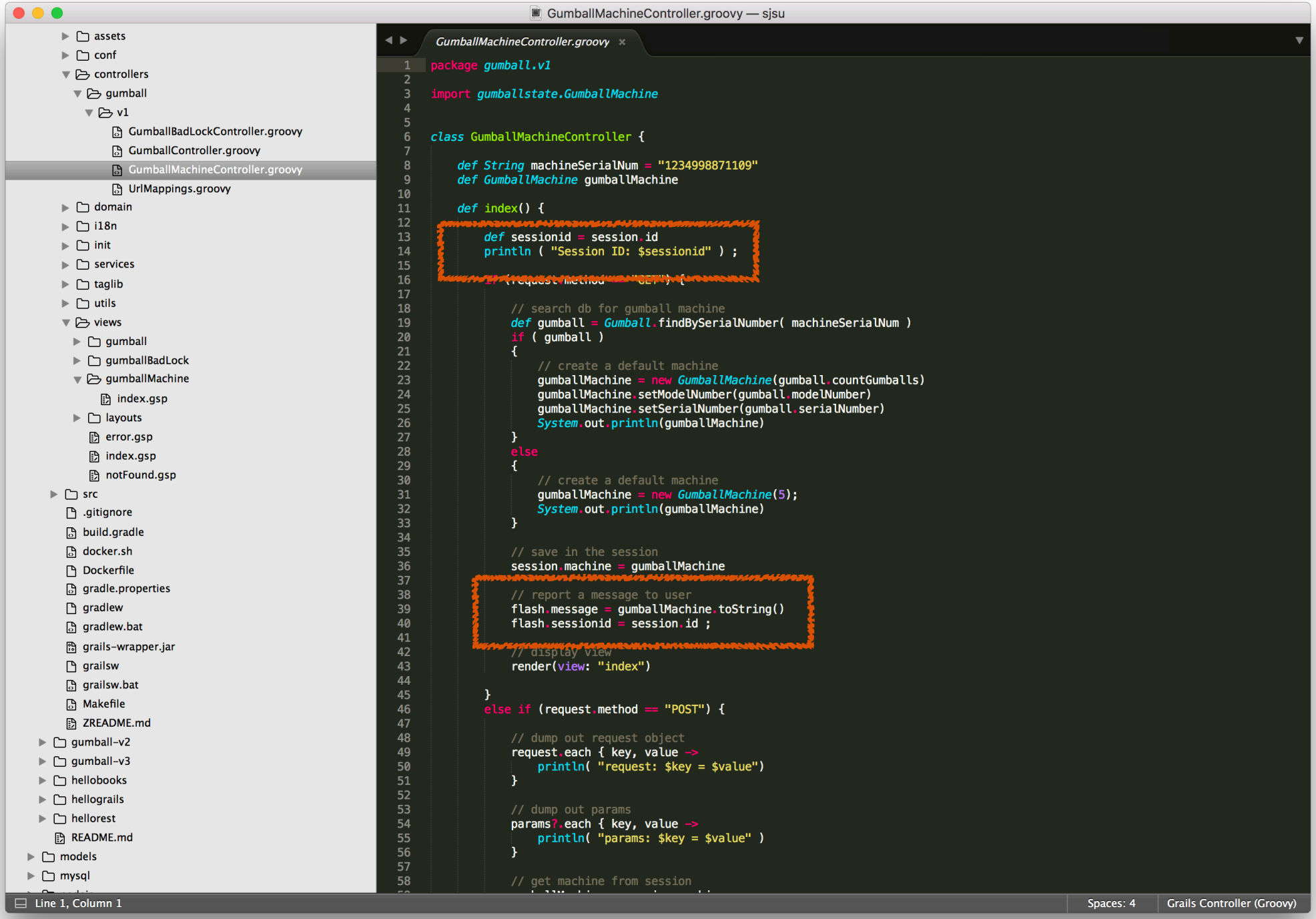
Gumball Machine Controller



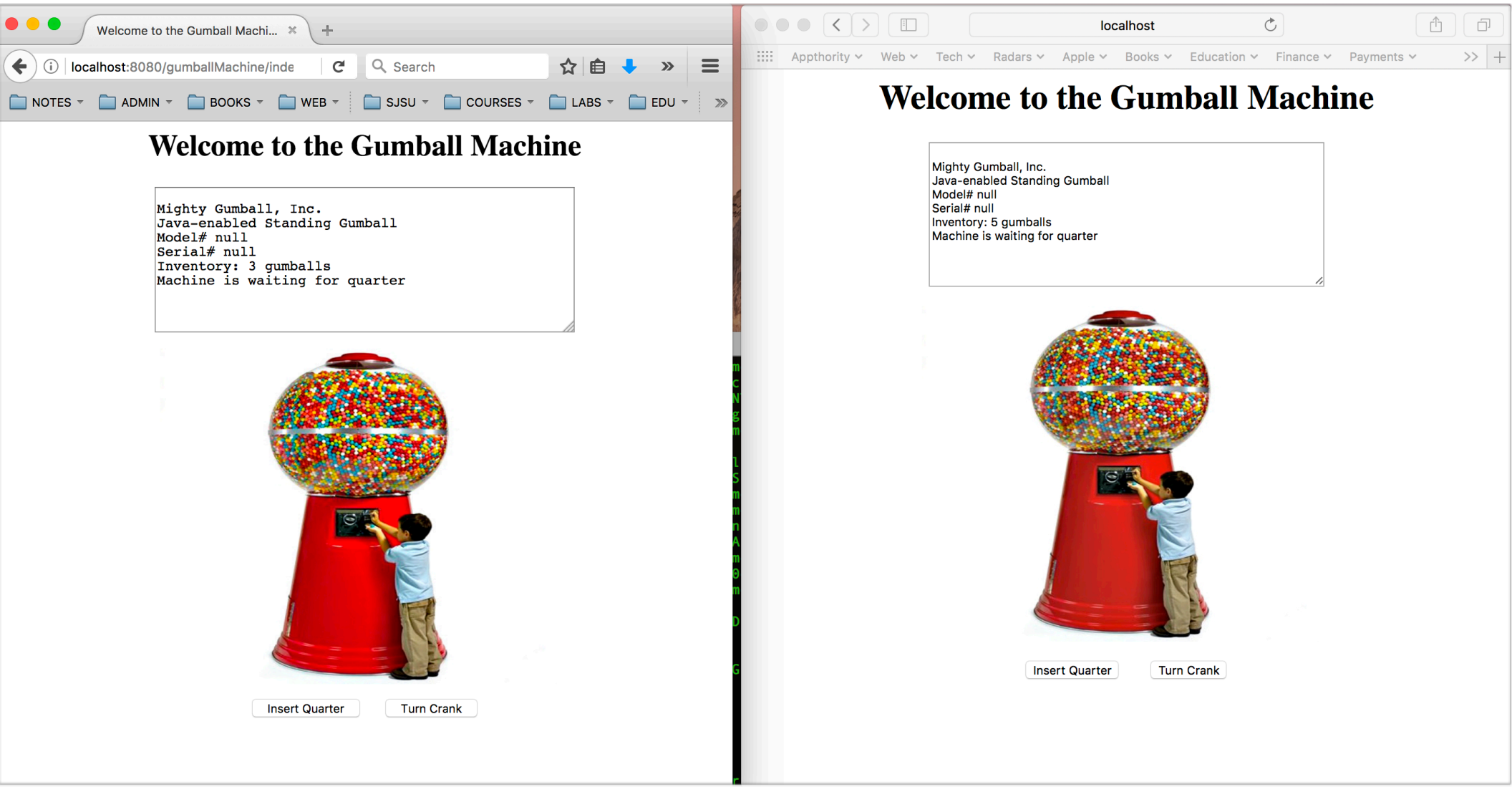
Gumball Machine View (GSP)



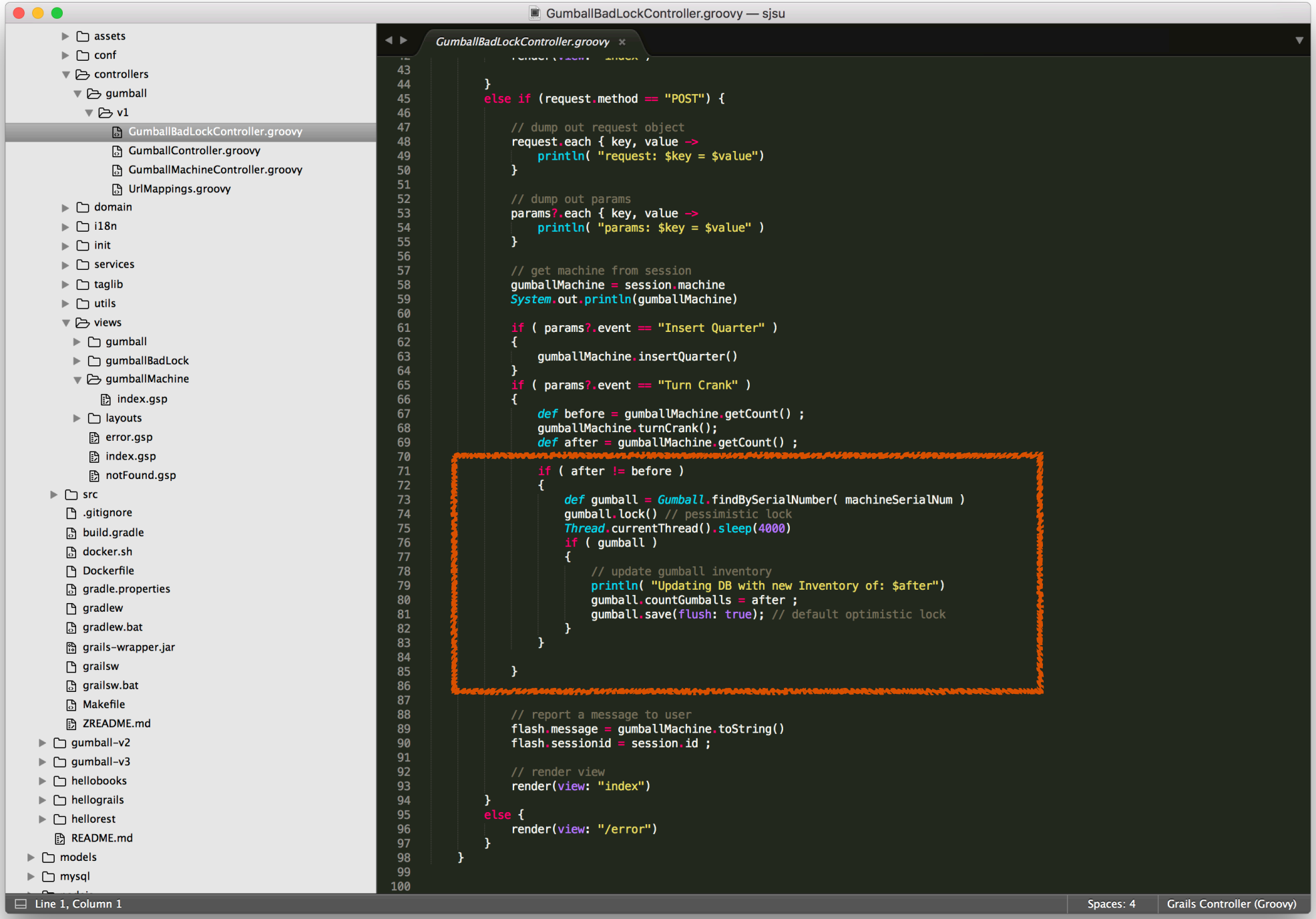
Using the “Sessions” and “Flash” Object



Problem with Sessions



Problem with Locks



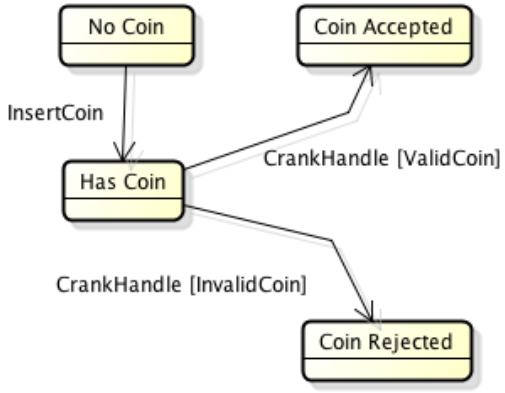
Gumball Machine (v2)

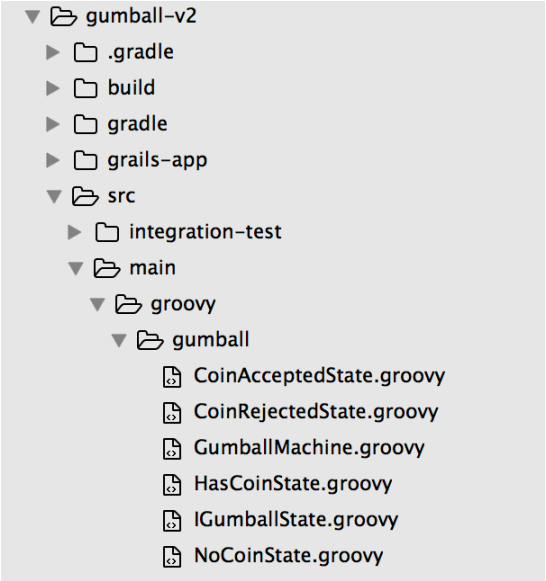
Refactoring's (from v1 to v2):

1. Gumball State Machine Responsible for Application State Only
2. Domain Object Persistent State moved to GORM (i.e. Inventory Count)

Refactoring's (from v1 to v2):

1. Gumball State Machine Responsible for Application State Only
2. Domain Object Persistent State moved to GORM (i.e. Inventory Count)





GumballMachine.groovy

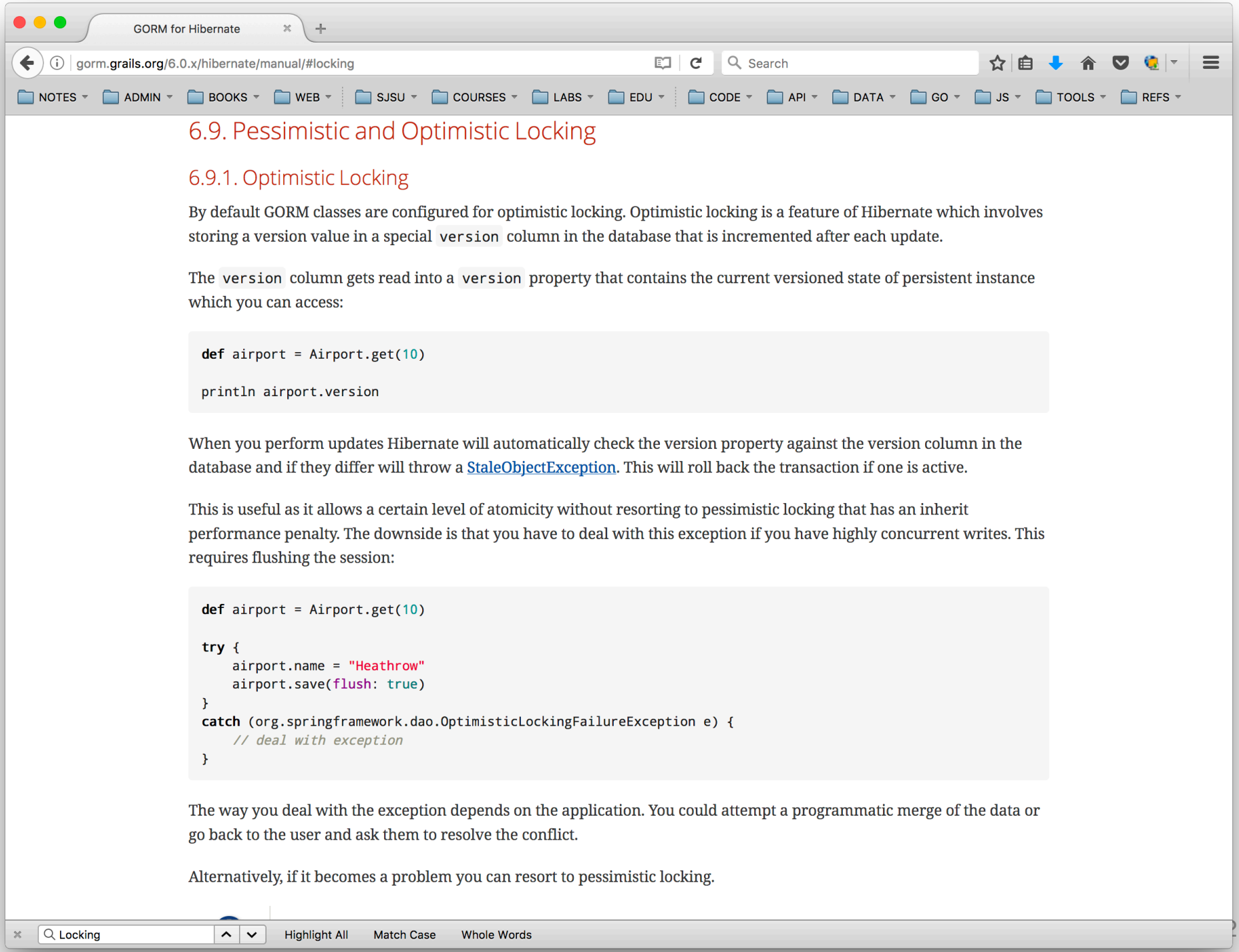
```groovy
package gumball

class GumballMachine {

    def String _model_number
    def String _serial_number
    def IGumballState _current_state

    NoCoinState _no_coin = new NoCoinState(this)
    CoinAcceptedState _coin_accepted = new CoinAcceptedState(this)
    CoinRejectedState _coin_rejected = new CoinRejectedState(this)
    HasCoinState _has_coin = new HasCoinState(this)

    GumballMachine(model, serial)
    {
        _current_state = _no_coin
        _model_number = model
        _serial_number = serial
    }

    def insertCoin()
    {
        _current_state.insertCoin()

    }

    def crankHandle()
    {
        _current_state.crankHandle()
    }

    def setNoCoin() { _current_state = _no_coin }
    def setCoinAccepted() { _current_state = _coin_accepted }
    def setCoinRejected() { _current_state = _coin_rejected }
    def setHasCoin() { _current_state = _has_coin }
    def String getCurrentState() { return _current_state.getClass().getName() }

    def setCurrentState(state)
    {
        if (state.equals("gumball.NoCoinState")) { setNoCoin() ; }
        if (state.equals("gumball.CoinAcceptedState")) { setCoinAccepted() ; }
        if (state.equals("gumball.CoinRejectedState")) { seCoinRejected() ; }
        if (state.equals("gumball.HasCoinState")) { setHasCoin() ; }
    }


def String getAbout() {
        return """
-------------------------------------------
Mighty Gumball, Inc.
Groovy-Enabled Standing Gumball
Model# ${_model_number}
Serial# ${_serial_number}
-------------------------------------------
Gumball Machine: ${this.toString()}
Current State: ${_current_state.toString()}
"""
}
```

Refactoring's (from v1 to v2):

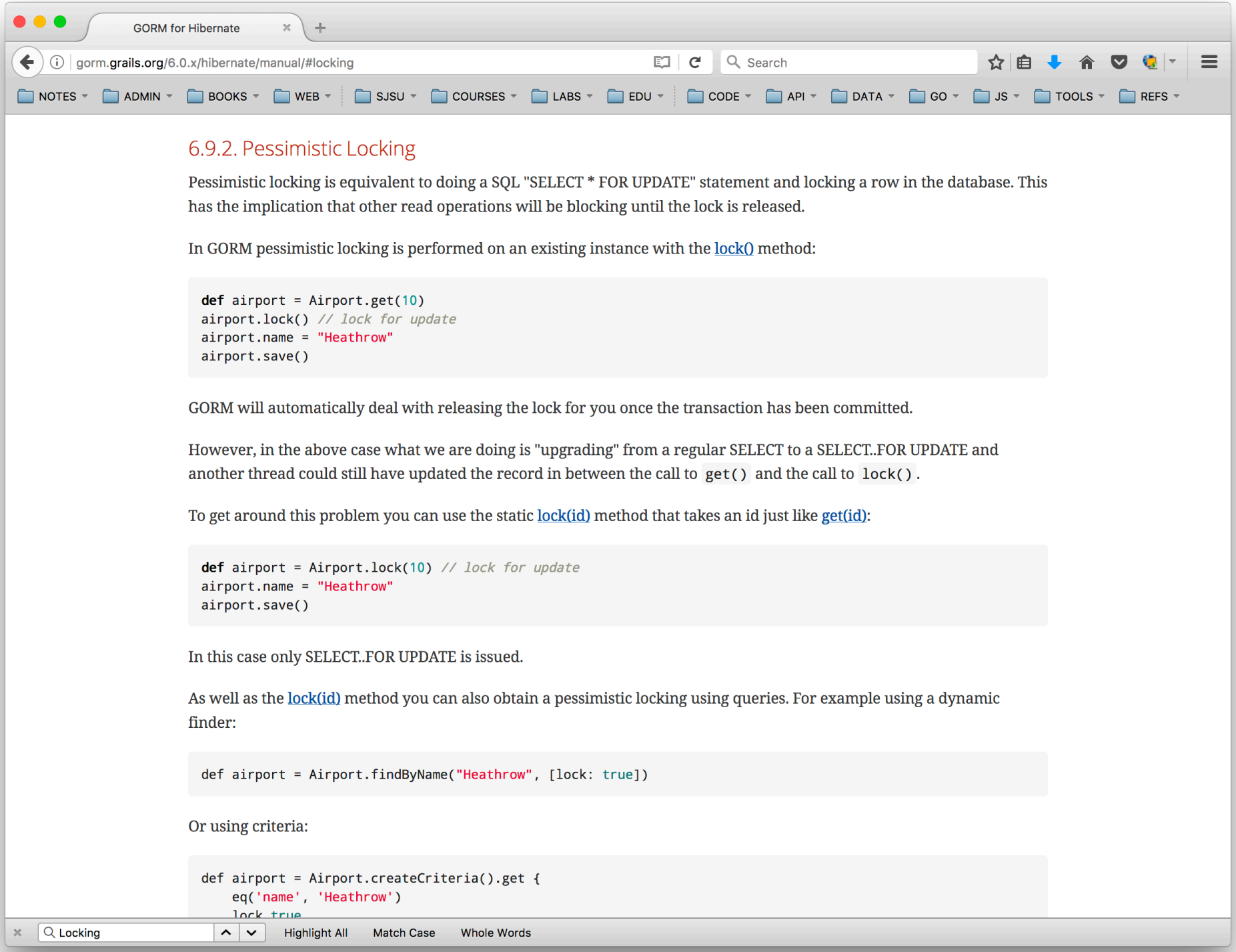
1. Gumball State Machine Responsible for Application State Only
2. Domain Object Persistent State moved to GORM (i.e. Inventory Count)

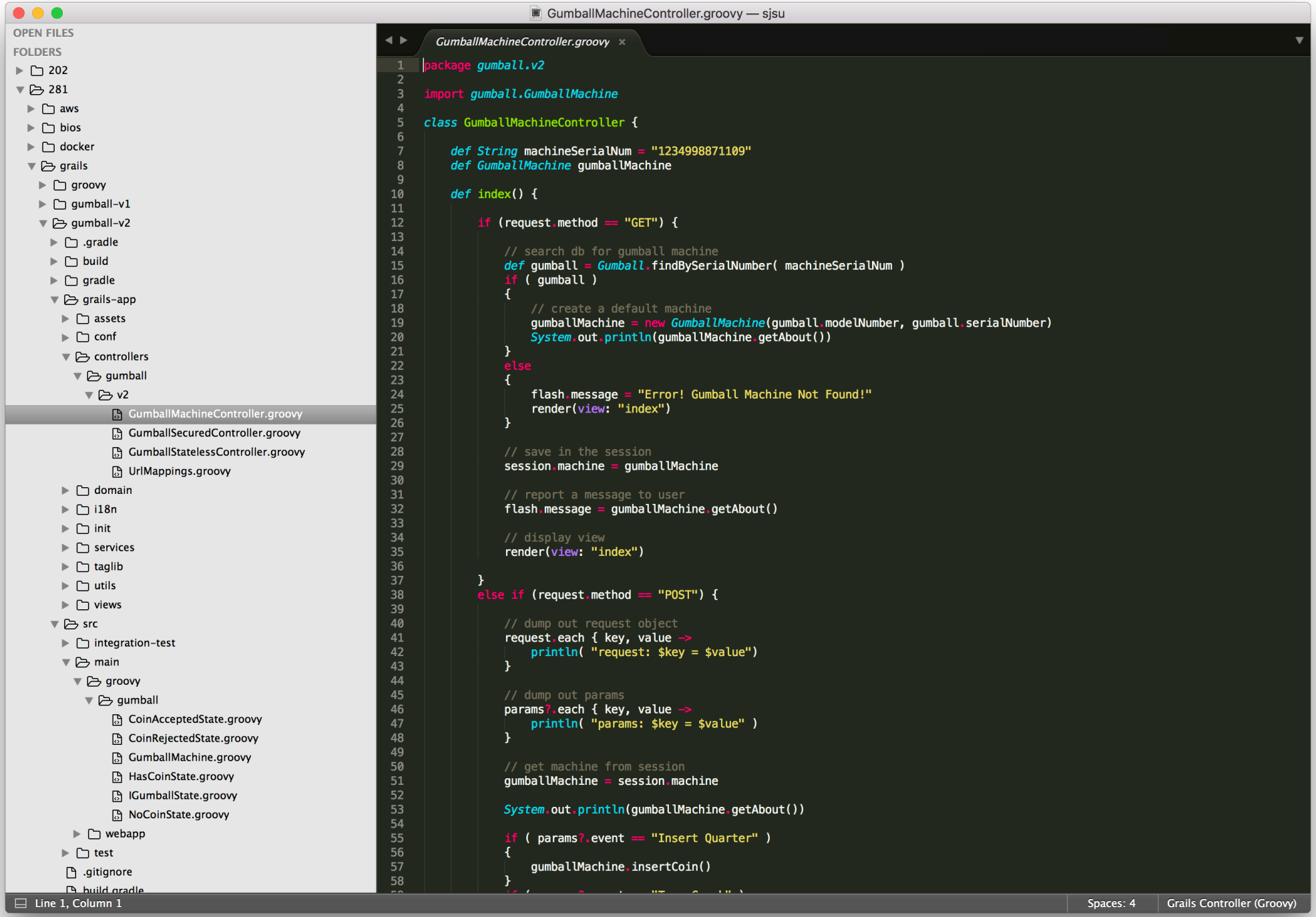
```
}
else if (request.method == "POST") {

    // dump out request object
    request.each { key, value ->
        println( "request: $key = $value")
    }

    // dump out params
    params?.each { key, value ->
        println( "params: $key = $value" )
    }

    // get machine from session
    gumballMachine = session.machine

    System.out.println(gumballMachine.getAbout())

    if ( params?.event == "Insert Quarter" )
    {
        gumballMachine.insertCoin()
    }
    if ( params?.event == "Turn Crank" )
    {
        gumballMachine.crankHandle();

        if ( gumballMachine.getCurrentState().equals("gumball.CoinAcceptedState") )
        {
            def gumball = Gumball.findBySerialNumber( machineSerialNum )
            if ( gumball )
            {
                // update gumball inventory
                // gumball.lock() // pessimistic lock
                if ( gumball.countGumballs > 0)
                    gumball.countGumballs-- ;
                gumball.save(flush: true); // default optimistic lock
            }
        }

    }
```

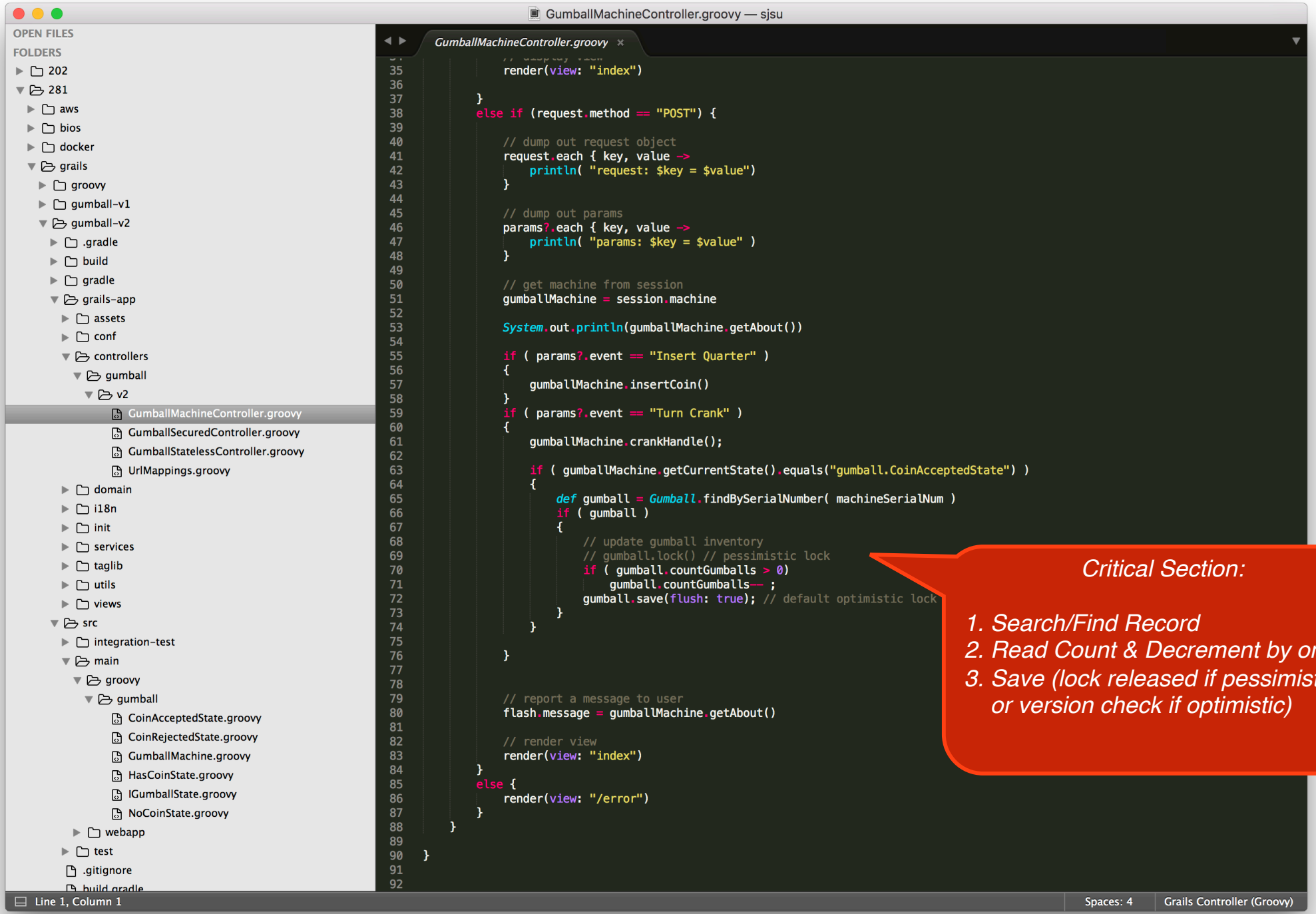
http://gorm.grails.org/6.0.x/hibernate/manual/#locking

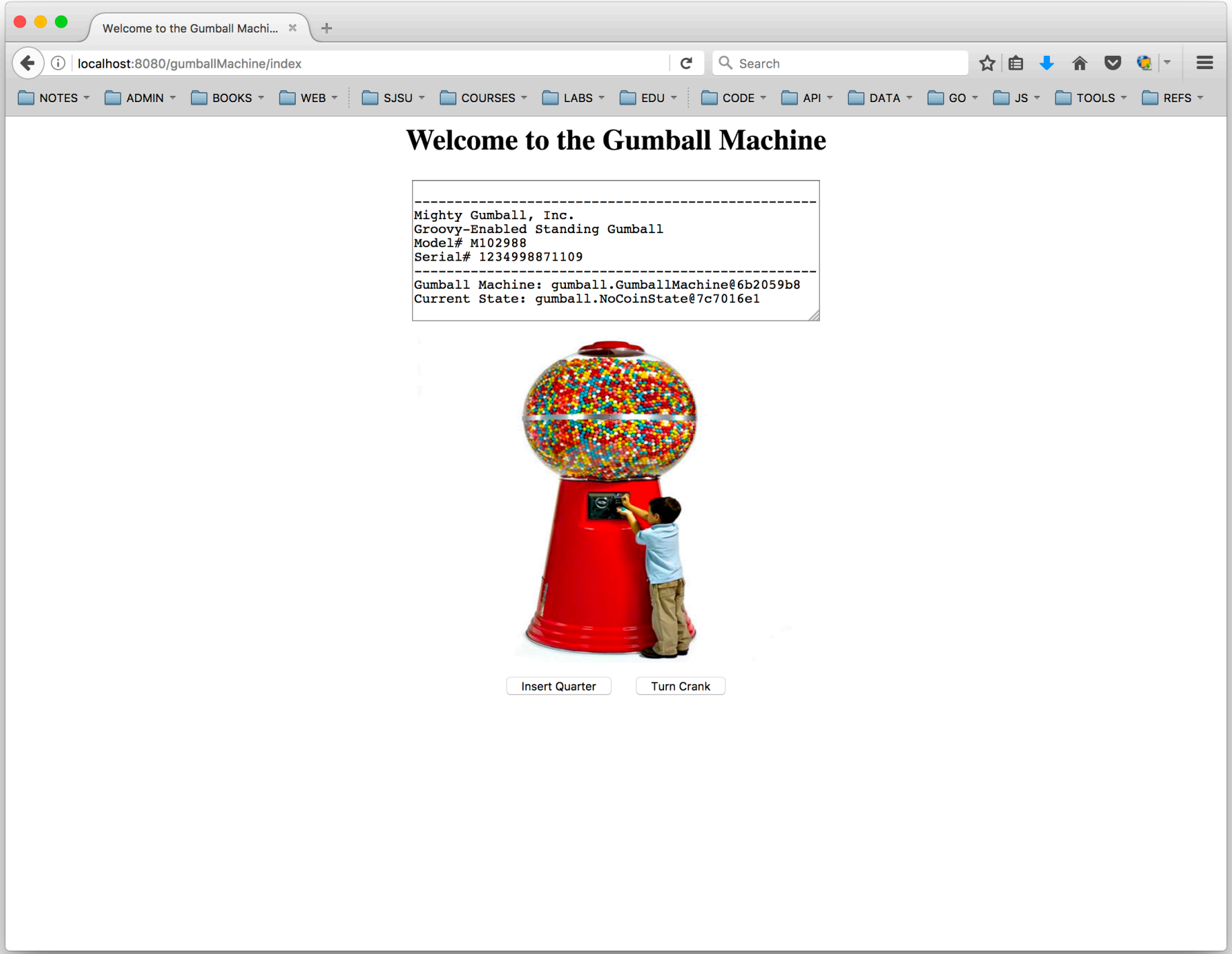


http://gorm.grails.org/6.0.x/hibernate/manual/#locking




GumballMachineController.groovy — sjsu
OPEN FILES
FOLDERS
202
281
aws
bios
docker
grails
groovy
gumball-v1
gumball-v2
.gradle
build
gradle
grails-app
assets
conf
controllers
gumball
v2
GumballMachineController.groovy
GumballSecuredController.groovy
GumballStatelessController.groovy
UrlMappings.groovy
domain
i18n
init
services
taglib
utils
views
src
integration-test
main
groovy
gumball
CoinAcceptedState.groovy
CoinRejectedState.groovy
GumballMachine.groovy
HasCoinState.groovy
IGumballState.groovy
NoCoinState.groovy
webapp
test
.gitignore
build.gradle
GumballMachineController.groovy
package gumball.v2
import gumball.GumballMachine
class GumballMachineController {
def String machineSerialNum = "1234998871109"
def GumballMachine gumballMachine
def index() {
if (request.method == "GET") {
// search db for gumball machine
def gumball = Gumball.findBySerialNumber(machineSerialNum)
if (gumball)
{
// create a default machine
gumballMachine = new GumballMachine(gumball.modelNumber, gumball.serialNumber)
System.out.println(gumballMachine.getAbout())
}
else
{
flash.message = "Error! Gumball Machine Not Found!"
render(view: "index")
}
// save in the session
session.machine = gumballMachine
// report a message to user
flash.message = gumballMachine.getAbout()
// display view
render(view: "index")
}
else if (request.method == "POST") {
// dump out request object
request.each { key, value ->
println("request: $key = $value")
}
// dump out params
params?.each { key, value ->
println("params: $key = $value")
}
// get machine from session
gumballMachine = session.machine
System.out.println(gumballMachine.getAbout())
if (params?.event == "Insert Quarter")
{
gumballMachine.insertCoin()
}
Line 1, Column 1
Spaces: 4
Grails Controller (Groovy)


GumballMachineController.groovy — sjsu
OPEN FILES
FOLDERS
202
281
aws
bios
docker
grails
groovy
gumball-v1
gumball-v2
.gradle
build
gradle
grails-app
assets
conf
controllers
gumball
v2
GumballMachineController.groovy
GumballSecuredController.groovy
GumballStatelessController.groovy
UrlMappings.groovy
domain
i18n
init
services
taglib
utils
views
src
integration-test
main
groovy
gumball
CoinAcceptedState.groovy
CoinRejectedState.groovy
GumballMachine.groovy
HasCoinState.groovy
IGumballState.groovy
NoCoinState.groovy
webapp
test
.gitignore
build.gradle
GumballMachineController.groovy
render(view: "index")
} else if (request.method == "POST") {
// dump out request object
request.each { key, value ->
println("request: $key = $value")
}
// dump out params
params?.each { key, value ->
println("params: $key = $value")
}
// get machine from session
gumballMachine = session.machine
System.out.println(gumballMachine.getAbout())
if (params?.event == "Insert Quarter")
{
gumballMachine.insertCoin()
}
if (params?.event == "Turn Crank")
{
gumballMachine.crankHandle();
if (gumballMachine.getCurrentState().equals("gumball.CoinAcceptedState"))
{
def gumball = Gumball.findBySerialNumber(machineSerialNum)
if (gumball)
{
// update gumball inventory
// gumball.lock() // pessimistic lock
if (gumball.countGumballs > 0)
gumball.countGumballs-- ;
gumball.save(flush: true); // default optimistic lock
}
}
}
// report a message to user
flash.message = gumballMachine.getAbout()
// render view
render(view: "index")
}
else {
render(view: "/error")
}
}
}
Critical Section:
1. Search/Find Record
2. Read Count & Decrement by one
3. Save (lock released if pessimistic, or version check if optimistic)
Line 1, Column 1
Spaces: 4
Grails Controller (Groovy)


Welcome to the Gumball Machi...
localhost:8080/gumballMachine/index
Search
NOTES
ADMIN
BOOKS
WEB
SJSU
COURSES
LABS
EDU
CODE
API
DATA
GO
JS
TOOLS
REFS
Welcome to the Gumball Machine
Mighty Gumball, Inc.
Groovy-Enabled Standing Gumball
Model# M102988
Serial# 1234998871109
Gumball Machine: gumball.GumballMachine@6b2059b8
Current State: gumball.NoCoinState@7c7016e1
Insert Quarter
Turn Crank

Gumball Machine (v2)

Refactoring's (from v1 to v2):

3. Remove Server Side “Session” State
4. Add “Secured” Client State Validation

Refactoring's (from v1 to v2):

3. Remove Server Side “Session” State
4. Add “Secured” Client State Validation

GumballStatelessController.groovy — sjsu

GumballStatelessController.groovy

```groovy
package gumball.v2

import gumball.GumballMachine

class GumballStatelessController {

    def String machineSerialNum = "1234998871109"
    def GumballMachine gumballMachine

    def index() {

        String VCAP_SERVICES = System.getenv('VCAP_SERVICES')

        if (request.method == "GET") {

            // search db for gumball machine
            def gumball = Gumball.findBySerialNumber( machineSerialNum )
            if ( gumball )
            {
                // create a default machine
                gumballMachine = new GumballMachine(gumball.modelNumber, gumball.serialNumber)
                System.out.println(gumballMachine.getAbout())
            }
            else
            {
                flash.message = "Error! Gumball Machine Not Found!"
                render(view: "index")
            }

            // don't save in the session
            // session.machine = gumballMachine

            // send machine state to client (instead)
            flash.state = gumballMachine.getCurrentState() ;
            flash.model = gumball.modelNumber ;
            flash.serial = gumball.serialNumber ;

            // report a message to user
            flash.message = gumballMachine.getAbout()

            // display view
            render(view: "index")

        }
        else if (request.method == "POST") {

            // dump out request object
            request.each { key, value ->
                println( "request: $key = $value")
            }

            // dump out params
            params?.each { key, value ->
                println( "params: $key = $value" )
            }
```

WebUI sends State (of State Machine) to Browser via HTML “hidden fields”

```
<!-- FORM SECTION -->
<form name="form1" method="post" action="">
    <p>
    <input type="hidden" name="state" id="state" value="${flash.state}" />
    <input type="hidden" name="model" id="model" value="${flash.model}" />
    <input type="hidden" name="serial" id="serial" value="${flash.serial}" />

    <div align="center">
        <textarea name="message" cols="50" rows="15" readonly id="message">
            ${flash.message}
        </textarea>
    </div>
```

Refactoring's (from v1 to v2):

3. Remove Server Side “Session” State
4. Add “Secured” Client State Validation

```
}
else if (request.method == "POST") {

    // dump out request object
    request.each { key, value ->
        println( "request: $key = $value")
    }

    // dump out params
    params?.each { key, value ->
        println( "params: $key = $value" )
    }

    // don't get machine from session
    // gumballMachine = session.machine

    // restore machine to client state (instead)
    def state = params?.state
    def modelNum = params?.model
    def serialNum = params?.serial
    gumballMachine = new GumballMachine(modelNum, serialNum) ;
    gumballMachine.setCurrentState(state) ;

    System.out.println(gumballMachine.getAbout())
```

Upon receiving each request, must reset State Machine to previous State tracked by Client Browser “hidden field”.

Welcome to the Gumball Machi...

localhost:8080/gumballStateless/index

Search

NOTES ADMIN BOOKS WEB SJSU COURSES LABS EDU CODE API DATA GO JS TOOLS REFS

Welcome to the Gumball Machine

```
------------------------------------------------------
Mighty Gumball, Inc.
Groovy-Enabled Standing Gumball
Model# M102988
Serial# 1234998871109
------------------------------------------------------
Gumball Machine: gumball.GumballMachine@30b67365
Current State: gumball.NoCoinState@492f4991
```

Insert Quarter Turn Crank

Welcome to the Gumball Machi... http://localhost:8080/gumballS...

view-source:http://localhost:8080/gumballStateless/index

Search

NOTES ADMIN BOOKS WEB SJSU COURSES LABS EDU CODE API DATA GO JS TOOLS REFS

```
1
2
3
4  <html>
5  <head>
6      <title>Welcome to the Gumball Machine (New Improved Version)</title>
7  </head>
8
9  <body>
10 <h1 align="center">Welcome to the Gumball Machine</h1>
11
12 <!-- FORM SECTION -->
13 <form name="form1" method="post" action="">
14     <p>
15     <input type="hidden" name="state" id="state" value="gumball.NoCoinState" />
16     <input type="hidden" name="model" id="model" value="M102988" />
17     <input type="hidden" name="serial" id="serial" value="1234998871109" />
18
19     <div align="center">
20         <textarea name="message" cols="50" rows="15" readonly id="message">
21
22 ------------------------------------------------------
23 Mighty Gumball, Inc.
24 Groovy-Enabled Standing Gumball
25 Model# M102988
26 Serial# 1234998871109
27 ------------------------------------------------------
28 Gumball Machine: gumball.GumballMachine&#64;30b67365
29 Current State: gumball.NoCoinState&#64;492f4991
30
31         </textarea>
32     </div>
33 </p>
34     <p align="center"><img src="/assets/giant-gumball-machine.jpg"  width="385" height="316"></p>
35
36     <p align="center">
37         <input type="submit" name="event" id="btnInsertQuarter" value="Insert Quarter">
38             
39         <input type="submit" name="event" id="btnTurnCrank" value="Turn Crank">
40     </p>
41 </form>
42 <!-- END FORM SECTION -->
43
44 </body>
45 </html>
```

Refactoring's (from v1 to v2):

3. Remove Server Side “Session” State
4. Add “Secured” Client State Validation

For Demo Only. Do not include Credentials in Source Code / GitHub for Production Systems

GumballSecuredController.groovy — sjsu

GumballSecuredController.groovy

```groovy
package gumball.v2

import gumball.GumballMachine
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import java.security.InvalidKeyException;

class GumballSecuredController {

    def String machineSerialNum = "1234998871109"
    def String secretKey = "kwRg54x2Go9iEdl49jFENRM12Mp711QI"
    def GumballMachine gumballMachine
    def String msg
    def hash

    // REF:  http://www.jokecamp.com/blog/examples-of-creating-base64-hashes-using-hmac-sha256-in-different-languages/#groovy
    def hmac_sha256(String secretKey, String data) {
     try {
        Mac mac = Mac.getInstance("HmacSHA256")
        SecretKeySpec secretKeySpec = new SecretKeySpec(secretKey.getBytes(), "HmacSHA256")
        mac.init(secretKeySpec)
        byte[] digest = mac.doFinal(data.getBytes())
        return digest
       } catch (InvalidKeyException e) {
        throw new RuntimeException("Invalid key exception while converting to HMac SHA256")
      }
    }

    def index() {

        String VCAP_SERVICES = System.getenv('VCAP_SERVICES')

        if (request.method == "GET") {

            // search db for gumball machine
            def gumball = Gumball.findBySerialNumber( machineSerialNum )
            if ( gumball )
            {
                // create a default machine
                gumballMachine = new GumballMachine(gumball.modelNumber, gumball.serialNumber)
                System.out.println(gumballMachine.getAbout())
            }
            else
            {
                flash.message = "Error! Gumball Machine Not Found!"
                render(view: "index")
            }

            // don't save in the session
            // session.machine = gumballMachine

            // send machine state to client (instead)
            flash.state = gumballMachine.getCurrentState() ;
            flash.model = gumball.modelNumber ;
```

Use HMAC SHA256 Hash

Refactoring's (from v1 to v2):

3. Remove Server Side “Session” State
4. Add “Secured” Client State Validation

On GET, generate “Clear Text String” and Send to Browser Timestamp and Hash

```
if (request.method == "GET") {

    // search db for gumball machine
    def gumball = Gumball.findBySerialNumber( machineSerialNum )
    if ( gumball )
    {
        // create a default machine
        gumballMachine = new GumballMachine(gumball.modelNumber, gumball.serialNumber)
        System.out.println(gumballMachine.getAbout())
    }
    else
    {
        flash.message = "Error! Gumball Machine Not Found!"
        render(view: "index")
    }

    // don't save in the session
    // session.machine = gumballMachine

    // send machine state to client (instead)
    flash.state = gumballMachine.getCurrentState() ;
    flash.model = gumball.modelNumber ;
    flash.serial = gumball.serialNumber ;

    // get system date
    flash.ts = System.currentTimeMillis().toString();
    msg = flash.state + "|" + flash.model + "|" + flash.serial + "|" + flash.ts + "|" + secretKey
    hash = hmac_sha256(secretKey, msg)
    println ( "MSG: " + msg )
    println ( "HASH: " + hash.encodeBase64() )
    flash.hash = hash.encodeBase64()

    // report a message to user
    flash.message = gumballMachine.getAbout()

    // display view
    render(view: "index")

}
```

```
<!-- FORM SECTION -->
<form name="form1" method="post" action="">
    <p>
    <input type="hidden" name="state" id="state" value="${flash.state}" />
    <input type="hidden" name="model" id="model" value="${flash.model}" />
    <input type="hidden" name="serial" id="serial" value="${flash.serial}" />
    <input type="hidden" name="ts" id="ts" value="${flash.ts}" />
    <input type="hidden" name="hash" id="hash" value="${flash.hash}" />

    <div align="center">
        <textarea name="message" cols="50" rows="15" readonly id="message">
            ${flash.message}
        </textarea>
    </div>
```

Refactoring's (from v1 to v2):

3. Remove Server Side “Session” State
4. Add “Secured” Client State Validation

```
// restore machine to client state (instead)
def state = params?.state
def modelNum = params?.model
def serialNum = params?.serial
def tsString = params?.ts

// check hash and time stamp
def long ts = Long.parseLong( tsString )
def long cts = System.currentTimeMillis()
def long diff = cts - ts
println ( diff/1000 ) // seconds
def hash1 = params?.hash
def String msg = state + "|" + modelNum + "|" + serialNum + "|" + tsString + "|" + secretKey
def hashBytes = hmac_sha256(secretKey, msg)
def hash2 = hashBytes.encodeBase64().toString()
println ( "MSG: " + msg )
println ( "HASH1: " + hash1 )
println ( "HASH2: " + hash2 )

def invalidTS = ((diff/1000) > 120)
def invalidHASH = (hash1 != hash2)
println( "invalid ts: " + invalidTS )
println( "invalid hash: " + invalidHASH )
```

On POST, validate Timestamp and Hash.

Welcome to the Gumball Machi... | http://localhost:8080/gumballS...

localhost:8080/gumballSecured/index

Welcome to the Gumball Machine

```
------------------------------------------------------
Mighty Gumball, Inc.
Groovy-Enabled Standing Gumball
Model# M102988
Serial# 1234998871109
------------------------------------------------------
Gumball Machine: gumball.GumballMachine@5e2451ae
Current State: gumball.NoCoinState@29866e1a
```

Insert Quarter | Turn Crank

view-source:http://localhost:8080/gumballSecured/index

```
<html>
<head>
    <title>Welcome to the Gumball Machine (New Improved Version)</title>
</head>

<body>
<h1 align="center">Welcome to the Gumball Machine</h1>

<!-- FORM SECTION -->
<form name="form1" method="post" action="">
    <p>
    <input type="hidden" name="state" id="state" value="gumball.NoCoinState" />
    <input type="hidden" name="model" id="model" value="M102988" />
    <input type="hidden" name="serial" id="serial" value="1234998871109" />
    <input type="hidden" name="ts" id="ts" value="1487408567581" />
    <input type="hidden" name="hash" id="hash" value="RpX7ciFt7H0DccY1Eplxgk7w7XMmFVq25nf/DT5hm8Y=" />

    <div align="center">
        <textarea name="message" cols="50" rows="15" readonly id="message">

------------------------------------------------------
Mighty Gumball, Inc.
Groovy-Enabled Standing Gumball
Model# M102988
Serial# 1234998871109
------------------------------------------------------
Gumball Machine: gumball.GumballMachine&#64;5e2451ae
Current State: gumball.NoCoinState&#64;29866e1a

        </textarea>
    </div>
</p>
    <p align="center"><img src="/assets/giant-gumball-machine.jpg"  width="385" height="316"></p>

    <p align="center">
        <input type="submit" name="event" id="btnInsertQuarter" value="Insert Quarter">
            
        <input type="submit" name="event" id="btnTurnCrank" value="Turn Crank">
    </p>
</form>
<!-- END FORM SECTION -->

</body>
</html>
```

Gumball (v3) - REST Service

Gumball REST Service (v3):

3. Transaction Object (Domain Model Changes)
4. REST Order Status and Place Order API

Gumball REST Service (v3):

1. Transaction Object (Domain Model Changes)
2. REST Order Status and Place Order API

Gumball.groovy

```groovy
package gumball.v3

class Gumball {

    String modelNumber
    String serialNumber
    Integer countGumballs

    static constraints = {
        serialNumber(unique: true)
    }
}
```

GumballOrder.groovy

```groovy
package gumball.v3

class GumballOrder {

    String  orderstatus ;

    static mapping = {
        id generator: 'increment'
    }

}
```

Why?

More Scalable and Friendly to Sharding to always Create Orders (i.e. No Read/Write Locks?).

Gumball REST Service (v3):

1. Transaction Object (Domain Model Changes)
2. REST Order Status and Place Order API

UrlMappings.groovy

```groovy
package gumball.v3

class UrlMappings {

    static mappings = {
        delete "/$controller/$id(.$format)?"(action:"delete")
        get "/$controller(.$format)?"(action:"index")
        get "/$controller/$id(.$format)?"(action:"show")
        post "/$controller(.$format)?"(action:"save")
        put "/$controller/$id(.$format)?"(action:"update")
        patch "/$controller/$id(.$format)?"(action:"patch")


        "/order/$id?"(controller: "GumballRest", parseRequest: true) {
            action = [GET: "orderStatus", POST: "placeOrder"]
        }

        get "/gumball"(controller: 'GumballRest', action:'machineStatus')

        "/"(controller: 'application', action:'index')
        "500"(view: '/error')
        "404"(view: '/notFound')
    }
}

```

Get Order Status API (GET with Order ID)
Place Order API (POST)

Gumball REST Service (v3):

1. Transaction Object
   (Domain Model Changes)
2. REST Order Status and
   Place Order API

Find GumballOrder.
If Found, Send back
in JSON Format

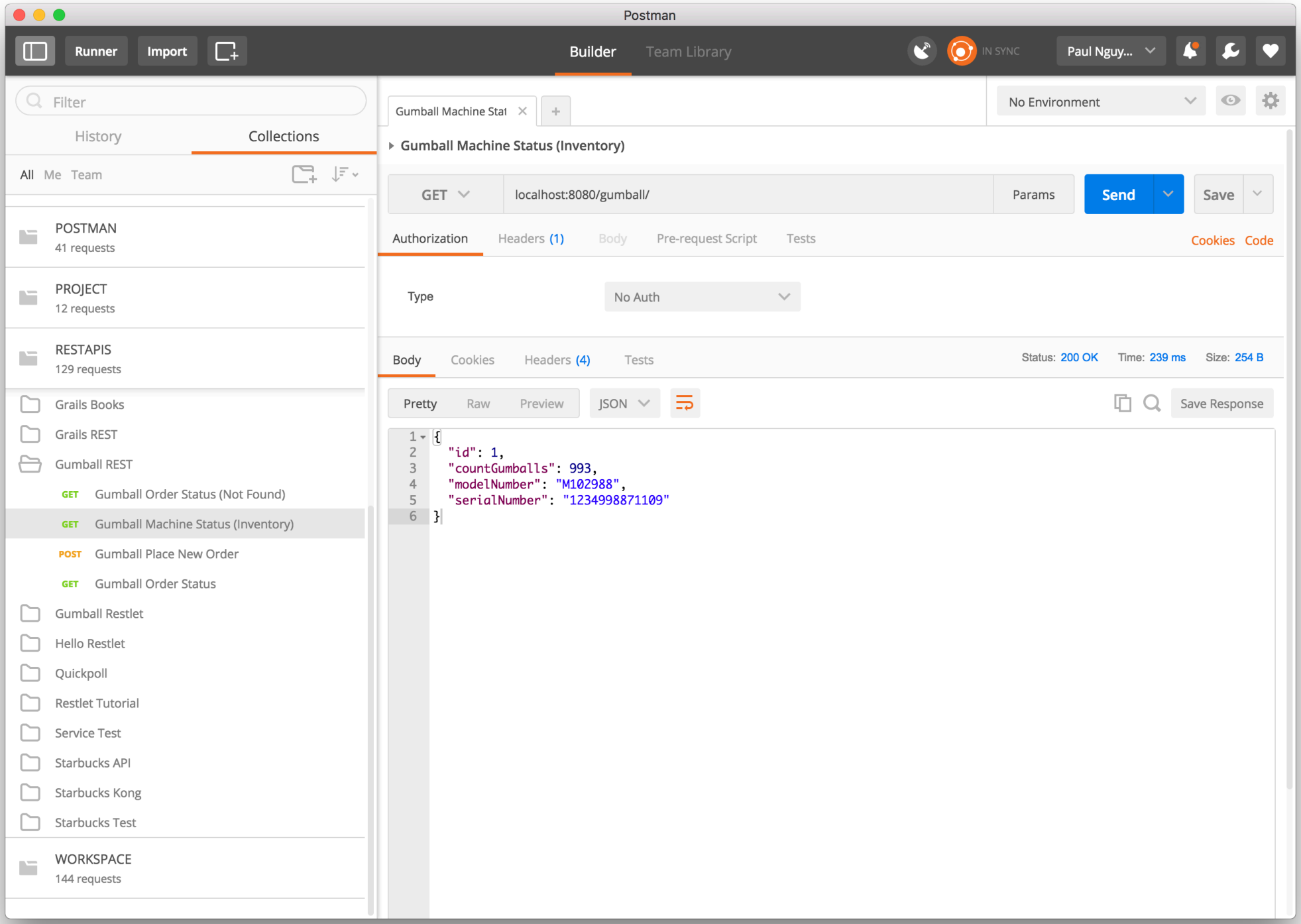
```
// HTTP GET -- Get Order Status
def orderStatus() {

    // dump out request object
    request.each { key, value ->
        println( "request: $key = $value")
    }

    // dump out params
    params?.each { key, value ->
        println( "params: $key = $value" )
    }

    if ( params.id )
    {
        def ord = GumballOrder.findById(params.id)
        if ( ord )
        {
            render ord as JSON
        }
        else {
            render(contentType: "application/json") {
                Error( message: "Order ${params.id} Not Found." )
            }
        }
    }
    else
    {
        render(contentType: "application/json") {
            Error( message: "Order Number Missing." )
        }
    }

}
```

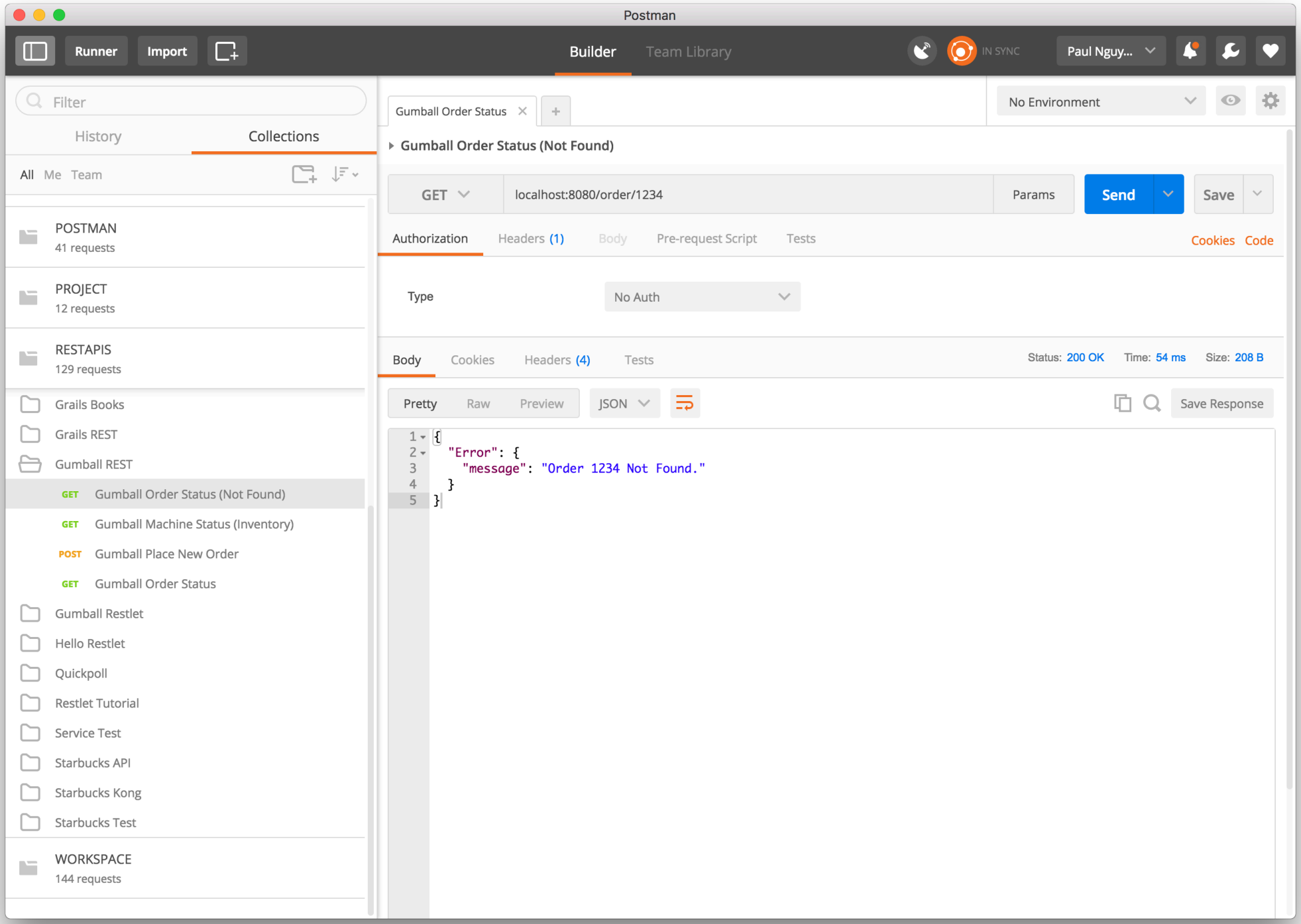
Gumball REST Service (v3):

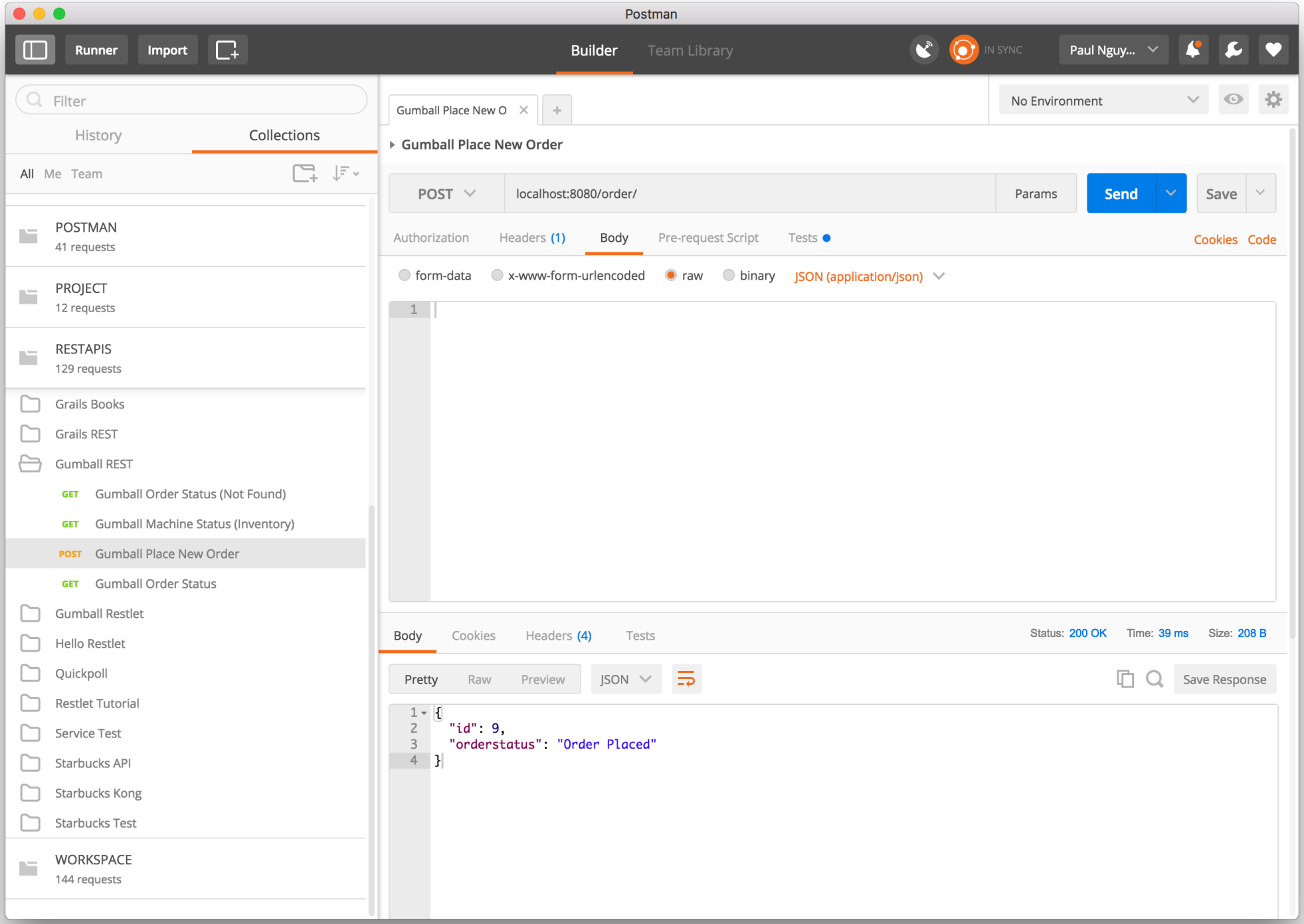
1. Transaction Object (Domain Model Changes)
2. REST Order Status and Place Order API

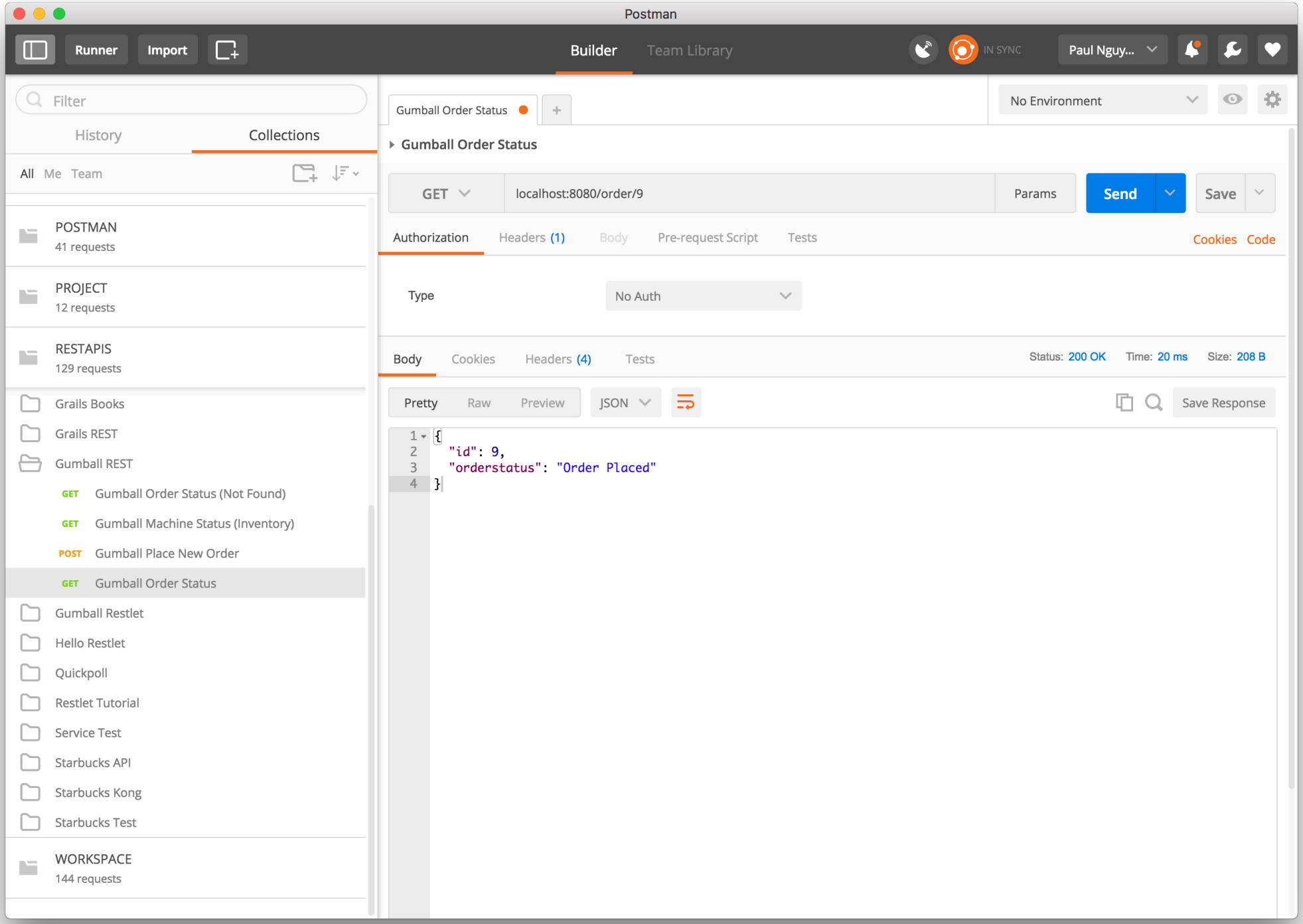
Check Inventory Count
If Low, Create a "Backorder"
Else, Create an Order and
Update Inventory QTY.

```
// HTTP POST -- Create a new Order
def placeOrder() {

    // dump out request object
    request.each { key, value ->
        println( "request: $key = $value")
    }

    // dump out params
    params?.each { key, value ->
        println( "params: $key = $value" )
    }

    def gumball = Gumball.findBySerialNumber( machineSerialNum )
    if ( gumball )
    {
        if ( gumball.countGumballs > 0)
        {
                gumball.lock() // pessimistic lock
                gumball.countGumballs-- ;
                gumball.save(flush: true);
                def newOrder = new GumballOrder(orderstatus: "Order Placed") ;
                newOrder.save(flush: true) ;
                render newOrder as JSON ;
        }
        else
        {
            def newOrder = new GumballOrder(orderstatus: "Backed Ordered") ;
            newOrder.save(flush: true) ;
            render newOrder as JSON ;
        }
    }
    else {
        render(contentType: "application/json") {
            Error( message: "System Error." )
        }
    }

}
```


Postman
Runner
Import
Builder
Team Library
IN SYNC
Paul Nguy...
Filter
History
Collections
All
Me
Team
POSTMAN
41 requests
PROJECT
12 requests
RESTAPIS
129 requests
Grails Books
Grails REST
Gumball REST
GET Gumball Order Status (Not Found)
GET Gumball Machine Status (Inventory)
POST Gumball Place New Order
GET Gumball Order Status
Gumball Restlet
Hello Restlet
Quickpoll
Restlet Tutorial
Service Test
Starbucks API
Starbucks Kong
Starbucks Test
WORKSPACE
144 requests
Gumball Machine Stat
No Environment
Gumball Machine Status (Inventory)
GET
localhost:8080/gumball/
Params
Send
Save
Authorization
Headers (1)
Body
Pre-request Script
Tests
Cookies
Code
Type
No Auth
Body
Cookies
Headers (4)
Tests
Status: 200 OK
Time: 239 ms
Size: 254 B
Pretty
Raw
Preview
JSON
Save Response
{
"id": 1,
"countGumballs": 993,
"modelNumber": "M102988",
"serialNumber": "1234998871109"
}


Postman
Runner
Import
Builder
Team Library
IN SYNC
Paul Nguy...
Filter
History
Collections
All
Me
Team
POSTMAN
41 requests
PROJECT
12 requests
RESTAPIS
129 requests
Grails Books
Grails REST
Gumball REST
GET Gumball Order Status (Not Found)
GET Gumball Machine Status (Inventory)
POST Gumball Place New Order
GET Gumball Order Status
Gumball Restlet
Hello Restlet
Quickpoll
Restlet Tutorial
Service Test
Starbucks API
Starbucks Kong
Starbucks Test
WORKSPACE
144 requests
Gumball Order Status
No Environment
Gumball Order Status (Not Found)
GET
localhost:8080/order/1234
Params
Send
Save
Authorization
Headers (1)
Body
Pre-request Script
Tests
Cookies
Code
Type
No Auth
Body
Cookies
Headers (4)
Tests
Status: 200 OK
Time: 54 ms
Size: 208 B
Pretty
Raw
Preview
JSON
Save Response
{
"Error": {
"message": "Order 1234 Not Found."
}
}


Postman
Runner
Import
Builder
Team Library
IN SYNC
Paul Nguy...
Filter
History
Collections
All
Me
Team
POSTMAN
41 requests
PROJECT
12 requests
RESTAPIS
129 requests
Grails Books
Grails REST
Gumball REST
GET Gumball Order Status (Not Found)
GET Gumball Machine Status (Inventory)
POST Gumball Place New Order
GET Gumball Order Status
Gumball Restlet
Hello Restlet
Quickpoll
Restlet Tutorial
Service Test
Starbucks API
Starbucks Kong
Starbucks Test
WORKSPACE
144 requests
Gumball Place New O
No Environment
Gumball Place New Order
POST
localhost:8080/order/
Params
Send
Save
Authorization
Headers (1)
Body
Pre-request Script
Tests
Cookies
Code
form-data
x-www-form-urlencoded
raw
binary
JSON (application/json)
1
Body
Cookies
Headers (4)
Tests
Status: 200 OK
Time: 39 ms
Size: 208 B
Pretty
Raw
Preview
JSON
Save Response
1 {
2 "id": 9,
3 "orderstatus": "Order Placed"
4 }


Postman
Runner
Import
Builder
Team Library
IN SYNC
Paul Nguy...
Filter
History
Collections
All
Me
Team
POSTMAN
41 requests
PROJECT
12 requests
RESTAPIS
129 requests
Grails Books
Grails REST
Gumball REST
GET Gumball Order Status (Not Found)
GET Gumball Machine Status (Inventory)
POST Gumball Place New Order
GET Gumball Order Status
Gumball Restlet
Hello Restlet
Quickpoll
Restlet Tutorial
Service Test
Starbucks API
Starbucks Kong
Starbucks Test
WORKSPACE
144 requests
Gumball Order Status
No Environment
Gumball Order Status
GET
localhost:8080/order/9
Params
Send
Save
Authorization
Headers (1)
Body
Pre-request Script
Tests
Cookies
Code
Type
No Auth
Body
Cookies
Headers (4)
Tests
Status: 200 OK
Time: 20 ms
Size: 208 B
Pretty
Raw
Preview
JSON
Save Response
{
"id": 9,
"orderstatus": "Order Placed"
}