

GETTING STARTED WITH THE COFFEESHOP DISPLAY FRAMEWORK

By Roberto Calderon
Media And Graphics Interdisciplinary Centre
University of British Columbia.
rvca@interchange.ubc.ca

The purpose of this guide is to get you started as quickly as possible with the MAGIC CoffeeShop framework, provided you have an Ubuntu machine and an Internet connection. This guide assumes you're installing everything from scratch. In a nutshell you will install a Broker2 server, Mysql, Apache2 (with PHP), Tomcat6, and deploy the CoffeeShop framework. You will then create your first web application.

DOWNLOAD AND START YOUR LOCAL OSGI BROKER (BROKER 2) SERVER.

If you haven't done yet, you need to configure a mysql server. In an Ubuntu box, you can do *sudo aptitude install mysql-server* follow the instructions and set up your root password.

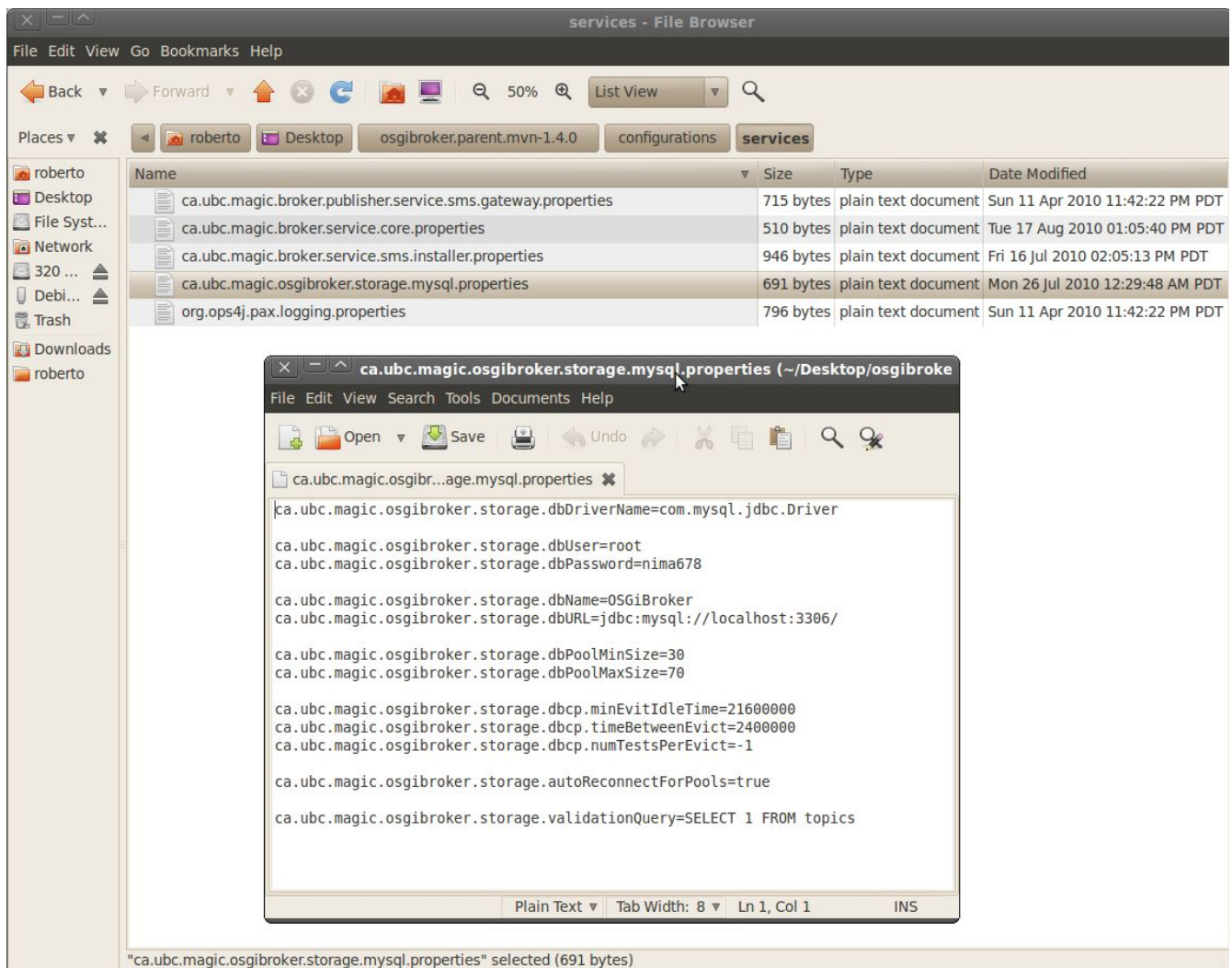
Download the broker code from the repository, or visit:

<http://pspi.magic.ubc.ca/code/index.php/p/osgibroker/downloads/>

Download version 1.40-beta4, this version solves some bugs that would otherwise slow down your applications. (<http://pspi.magic.ubc.ca/code/index.php/p/osgibroker/downloads/49/>)

Unpack the tar.gz file. Now there's a couple of things you need to configure first:

- 1) Go into the uncompressed folder and into *osgibroker.parent.mvn-1.4.0/configurations/services/*
- 2) Open the *mysql.properties* file and change the *dbUser* and *dbPassword* fields. Please note that using root as dbUser poses a high security risk. We have set it up this way to make it easier for you to deploy, but DO NOT USE ROOT IF YOU'RE PLANNING TO DEPLOY THIS FOR PRODUCTION.



You're set up!! Now run the *start.sh* file through a command line (*./start.sh*) or use *nohup* to run as a background process. You can test if all is good by subscribing to a topic with the url:

<http://localhost:8800/osgibroker/subscribe?topic=MAGIC&clientID=bigscreen1>

You can find more on deploying and configuring your server at:

<http://pspi.magic.ubc.ca/code/index.php/p/osgibroker/page/osgibroker-webservice/>

CONFIGURE THE COFFEESHOP COINTAINER.

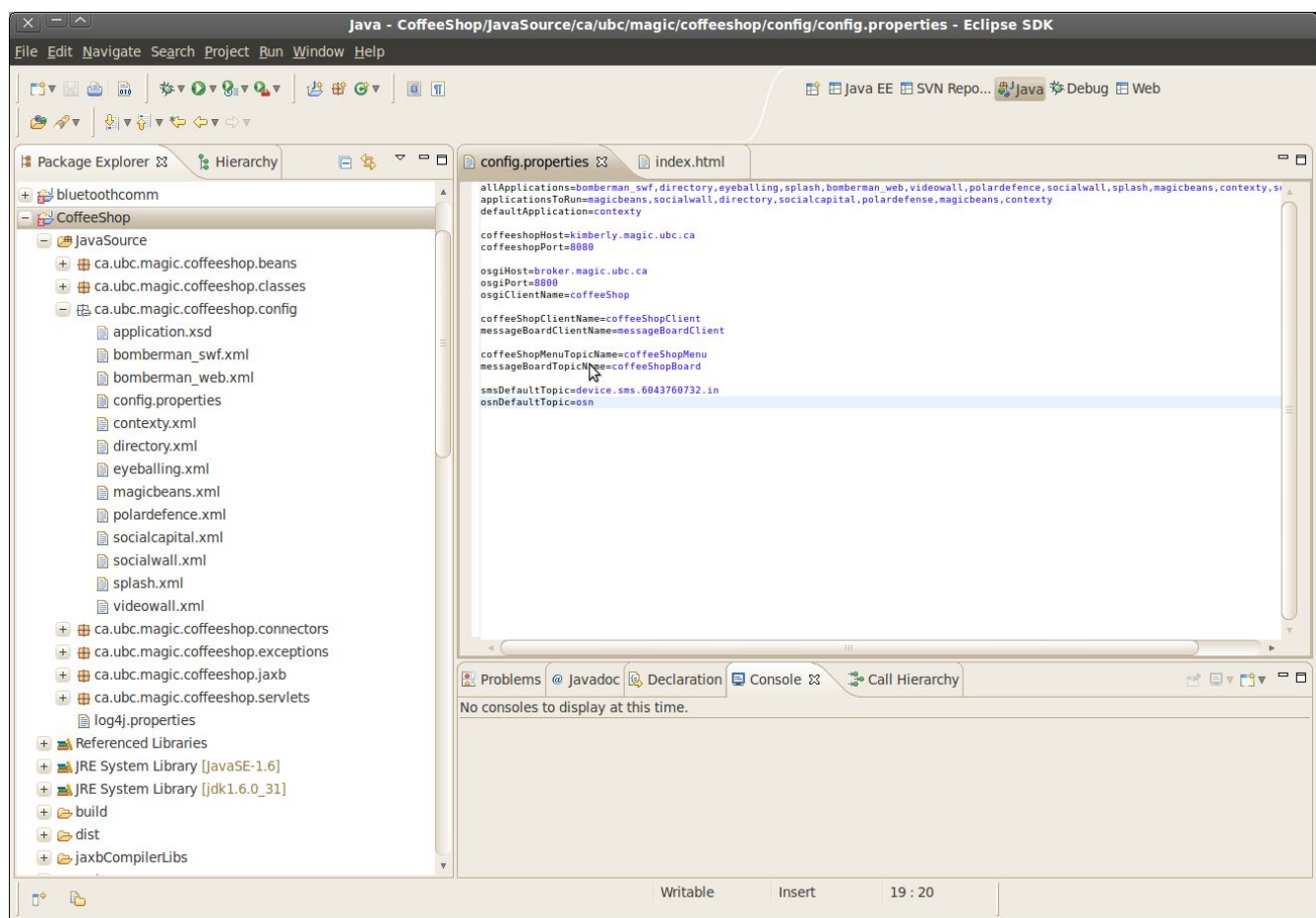
In the *ca.ubc.magic.coffeeshop.config* package open the file *config.properties* and configure

- 1) the coffeeshopHost
- 2) the coffeeshopPort
- 3) the osgiHost
- 4) the osgiPort

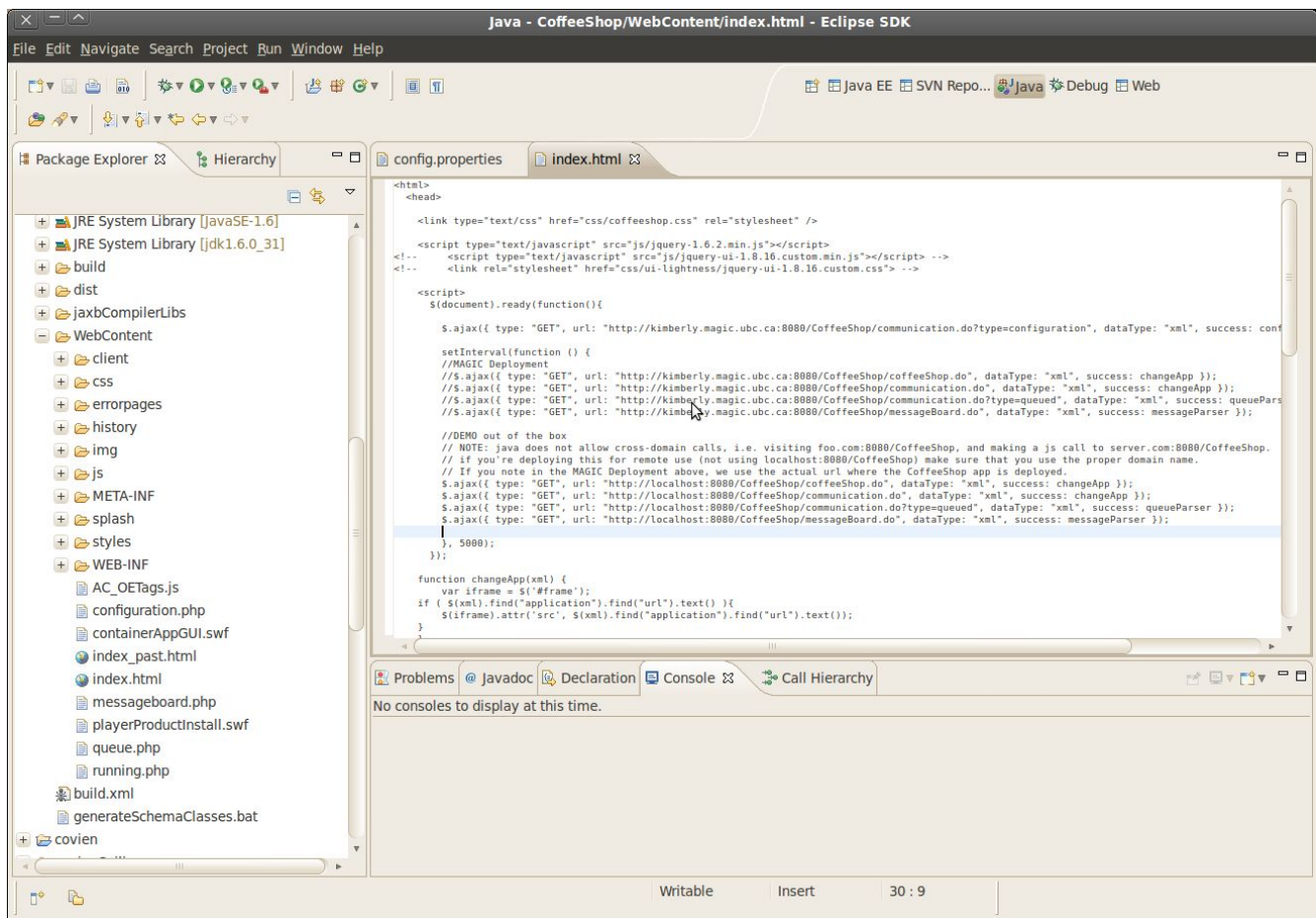
If you're supporting SMS interaction you need to configure

- 1) smsDefaultTopic

To the osgi broker topic you're using to broadcast SMS messages in your broker configuration



If you are going to deploy and access the CoffeShop framework in other host other than localhost you need to configure the GUI. Within *WebContent* open the file *index.html* and configure the domain used by the ajax calls.



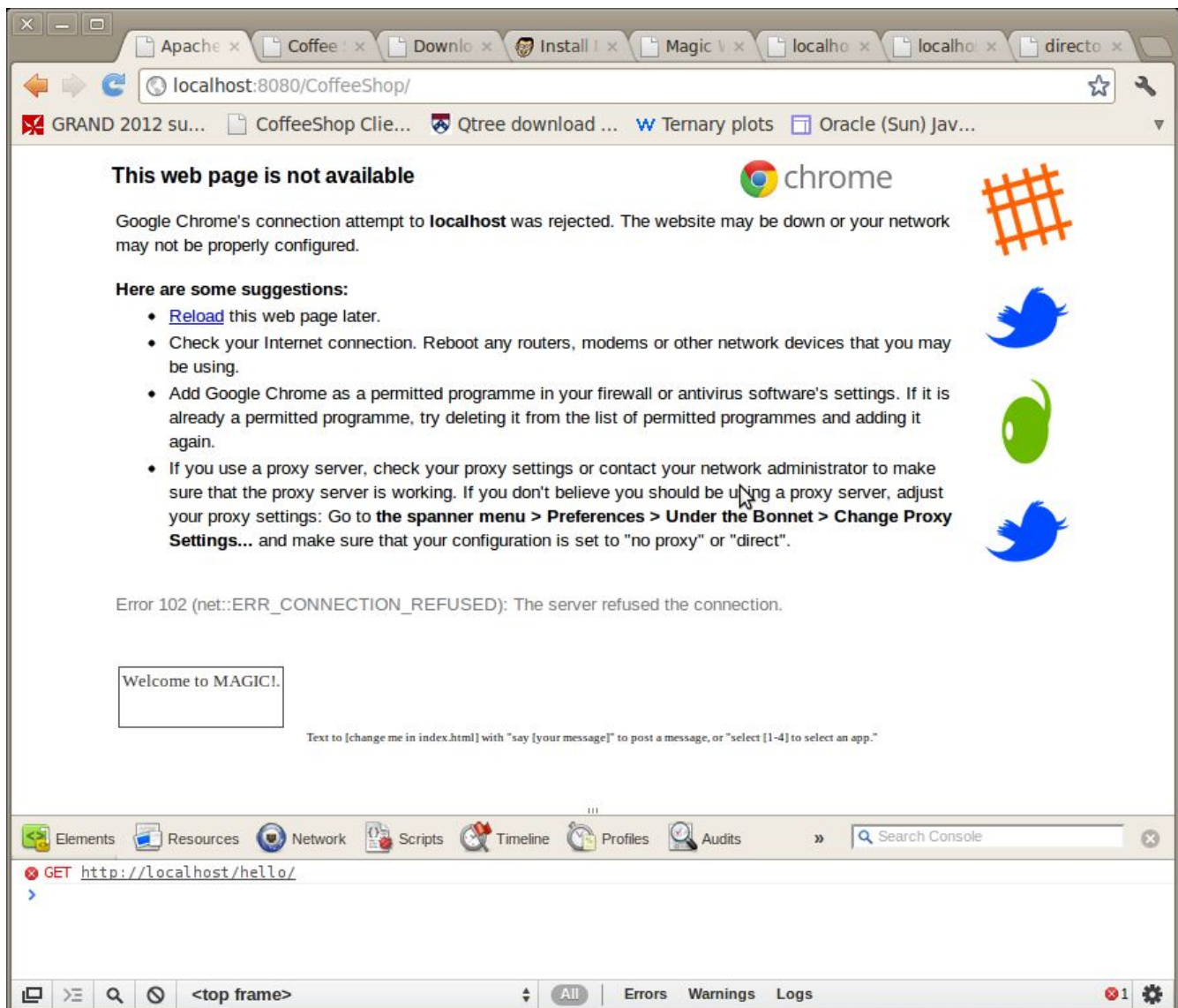
BUILD AND DEPLOY.

To build the project run the build.xml ant file located at the root of the project. Right click on the file and choose *Run As > Ant Build*.

A *CoffeeShop.war* file will be created in the *dist* folder. You can upload this file to your Tomcat installation through the tomcat's manager portal, or copy the file to */var/lib/tomcat6/webapps* in an Ubuntu box. We have tested this war file under Tomcat6 with success.

IF YOU DON'T WANT TO COMPILE THE CODE WE HAVE PROVIDED A READY-FOR-DEPOLYMENT WAR FILE TO BE USED WITH A DEFAULT TOMCAT6 INSTALLATION. YOU CAN FIND THIS WAR UNDER THE “DIST” FOLDER.

Now let's visit our container at <http://localhost:8080/CoffeeShop>



Oops! It looks like something is broken! Luckily we are going to fix it by creating our first application.

CREATING AND DEPLOYING YOUR FIRST APPLICATION

The CoffeeShop framework is good at managing interaction with different applications. It can manage servlet and web-based applications and the brokerage to them. Once an application has been selected the CoffeeShop framework will keep it active until messages delivered to the broker and targeted to such application cease to exist. A more in-depth explanation of the system can be found in the file *CoffeeShop_TechnicalDocumentation.docx* within the “documentaion” folder in the root of the CoffeeShop project.

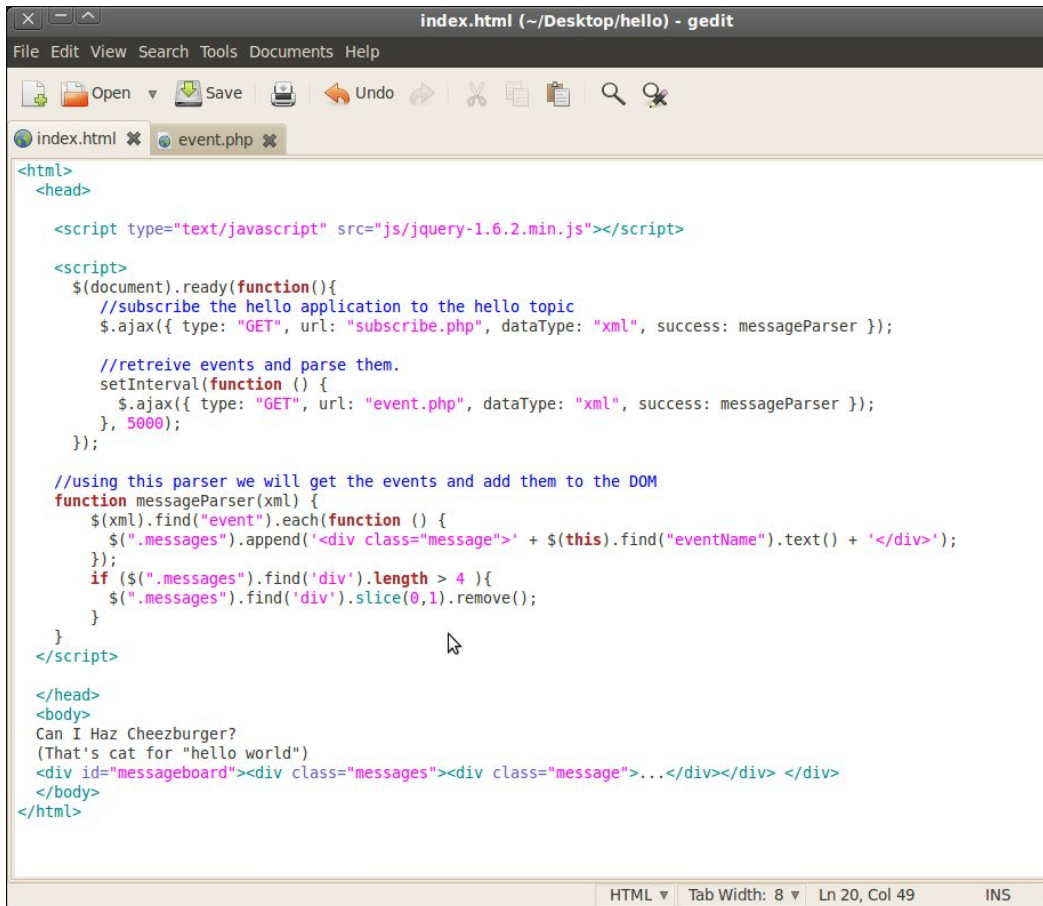
Now let's create our first application.

First let's set up our Apache2 (with PHP support) server. Install with `sudo aptitude install apache2 php5` (We need php to bypass Access-Control-Allow-Origin in Ajax calls configured out-of-the-box, and make it easier for you to just get going. If you wish to use Javascript-pure applications either use

full domains instead of localhost or use json within your application.)

Now let's write a very quick and easy web application using jquery. We will use PHP to avoid same-domain problems using ajax. So let's write that code. When the document is ready we subscribe to a hello topic. Then every 5000 ms we will make a call for events on the broker and parse them. If we get events we will change the DOM and display it in our application.

Ok... I'll make your life easier. The app is already written and lives in the CoffeeShop project under the folder "hello". Copy this folder to your /var/www/ folder in Ubuntu (or wherever your public html folder is)



```
<html>
<head>

<script type="text/javascript" src="js/jquery-1.6.2.min.js"></script>

<script>
$(document).ready(function(){
    //subscribe the hello application to the hello topic
    $.ajax({ type: "GET", url: "subscribe.php", dataType: "xml", success: messageParser });

    //retrieive events and parse them.
    setInterval(function () {
        $.ajax({ type: "GET", url: "event.php", dataType: "xml", success: messageParser });
    }, 5000);
});

//using this parser we will get the events and add them to the DOM
function messageParser(xml) {
    $(xml).find("event").each(function () {
        $(".messages").append('<div class="message">' + $(this).find("eventName").text() + '</div>');
    });
    if $(".messages").find('div').length > 4 ){
        $(".messages").find('div').slice(0,1).remove();
    }
}
</script>

</head>
<body>
Can I Haz Cheezburger?
(That's cat for "hello world")
<div id="messageboard"><div class="messages"><div class="message">...</div></div> </div>
</body>
</html>
```

event.php (~/Desktop/hello) - gedit

File Edit View Search Tools Documents Help

Open Save Undo Redo Cut Copy Paste Find

index.html event.php

```
<?php
// Set your return content type
//header('Content-type: application/xml');

$dauurl = 'http://localhost:8800/osgibroker/event?topic=hello&clientID=hello&timeOut=2';

// Get that website's content
$handle = fopen($dauurl, "r");

// If there is something, read and return
if ($handle) {
    while (!feof($handle)) {
        $buffer = fgets($handle, 4096);
        echo $buffer;
    }
    fclose($handle);
}
?>
```

PHP Tab Width: 8 Ln 18, Col 3 INS

subscribe.php (~/Desktop/hello) - gedit

File Edit View Search Tools Documents Help

Open Save Undo Redo Cut Copy Paste Find

index.html event.php subscribe.php

```
<?php
// Set your return content type
//header('Content-type: application/xml');

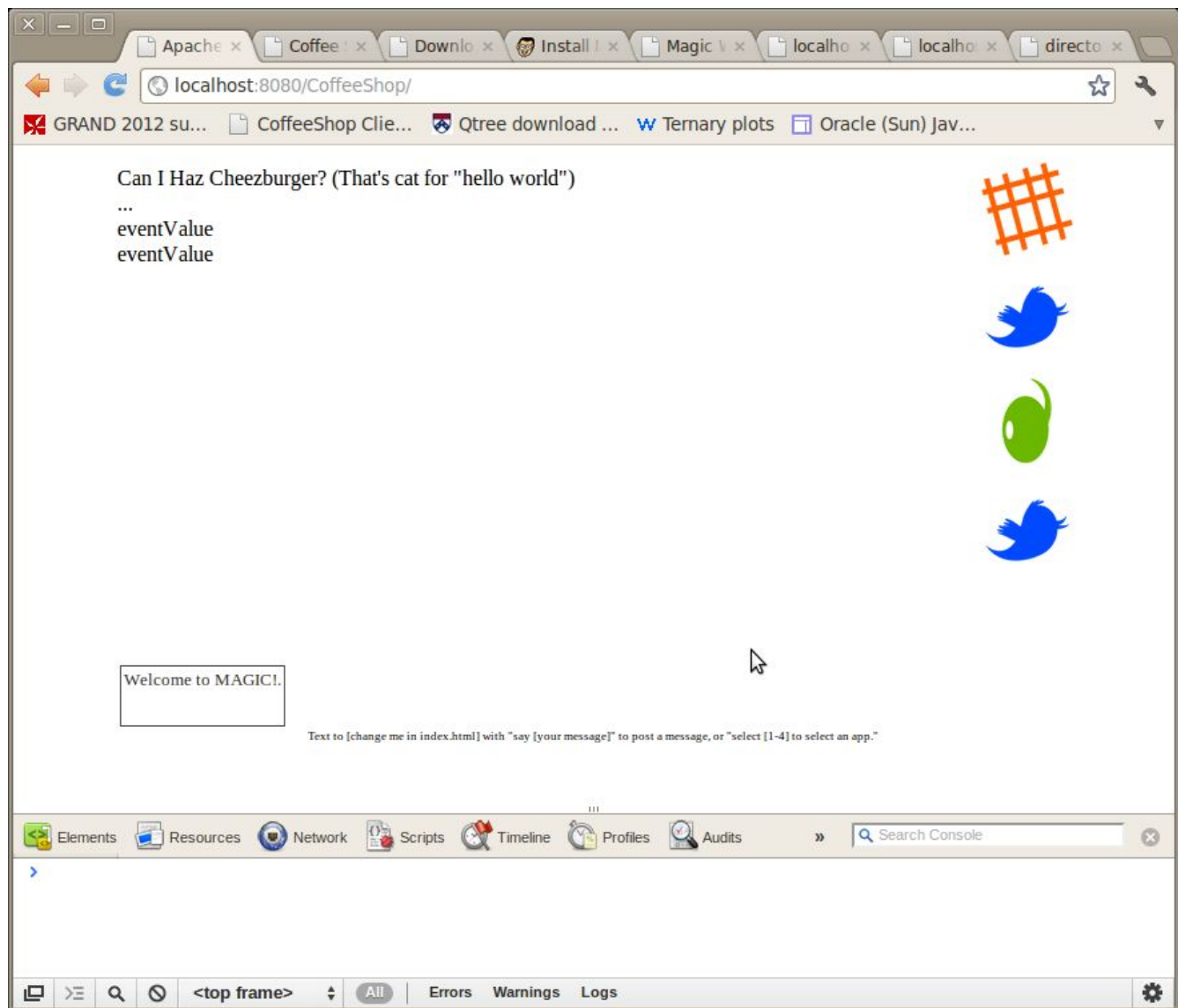
$dauurl = 'http://localhost:8800/osgibroker/subscribe?topic=hello&clientID=hello';

// Get that website's content
$handle = fopen($dauurl, "r");

// If there is something, read and return
if ($handle) {
    while (!feof($handle)) {
        $buffer = fgets($handle, 4096);
        echo $buffer;
    }
    fclose($handle);
}
?>
```

PHP Tab Width: 8 Ln 13, Col 36 INS

Let's re-load our CoffeeShop framework and... voila it works! Now that's exciting.

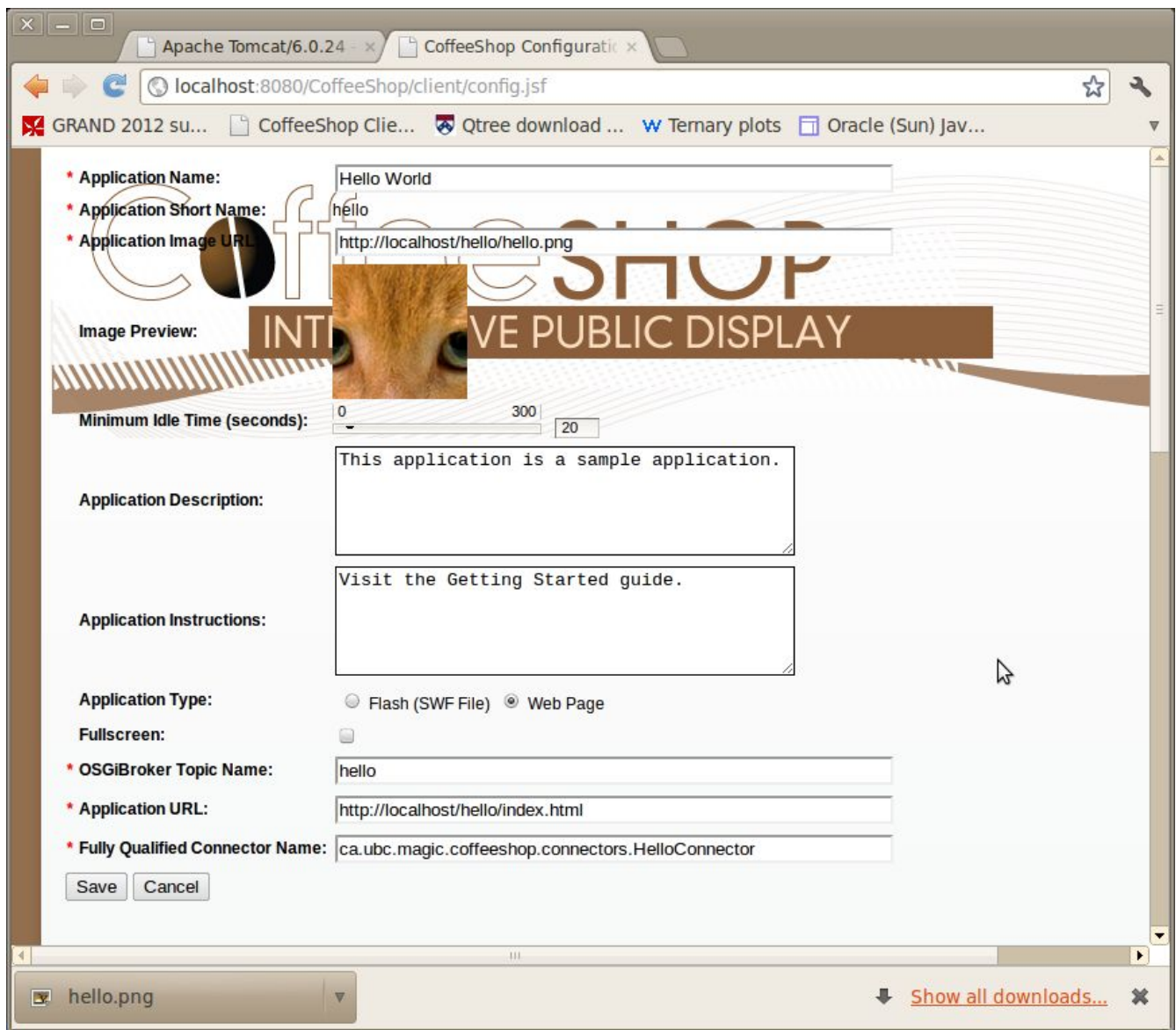


Now let's send an event to the broker. Open a web-browser and send the event with the url:

[http://localhost:8800/osgibroker/event?
topic=hello&clientID=hello&method=POST&eventName=eventValue](http://localhost:8800/osgibroker/event?topic=hello&clientID=hello&method=POST&eventName=eventValue)

Every time you visit that link you'll send an event to the app. Let's configure the framework using the provided manager applications within the app.

Visit <http://localhost:8080/CoffeeShop/client/config.jsf> . By clicking on view/edit you can configure the application name, image presented, idle time and most importantly the osgibroker topic name. So let's do that for the hello application.



You're done!

Well... this is not entirely true... The Hello application is configured out of the box. If you are going to deploy a new application you need to do two things:

- 1) Create a `<application>.xml` file under the package `ca.ubc.magic.coffeeshop.config`, just clone the `hello.xml` file.
- 2) Create a connector `<application>Connector.java` in the package `ca.ubc.magic.coffeeshop.connectors`, just clone the `HelloConnector.java` and change the topic names to match your application configuration.

It is important to have these connectors as they are used to queue and de-queue any new application.

Now let's queue an application with the web interface provided. Visit

<http://localhost:8080/CoffeeShop/client/select.jsf>

Select any application and click on *Queue this Application to Run* if you want to run it. Or send a message to the Message Board.

