Quick2Cloud for Cloud Systems

Built on IBM Cloud

CLOUD FOUNDRY

Quick2Cloud College - Course 100

CONGRATULATIONS!

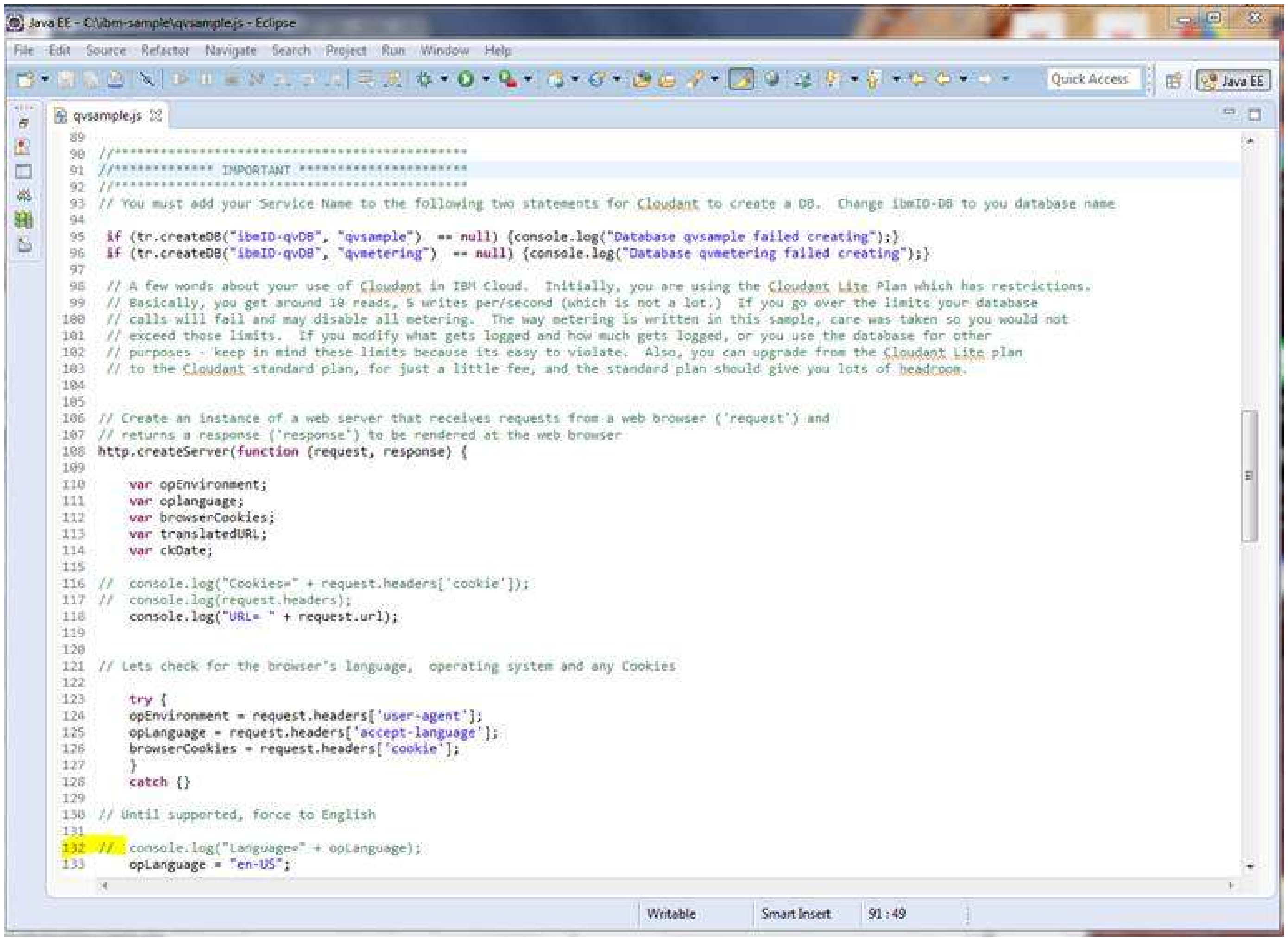
Your *Quick2Cloud* Sample Web Application has been pushed up to the IBM Cloud and should now be running. Once again, the IBM Cloud generated a URL shown in the last few lines of your shell display. Use this URL for now when you want access to the application from a web browser.

Let□s play for a few moments

TIP: You are encouraged to use the *Quick2Cloud□s* Sample Web Application as a starting point for building your own application. Just edit and make it your own or replace it with a new or existing application after graduation.

Step 7

The *Quick2Cloud* sample web server code is located in the q2c-web-app directory under the name qvsample.js. Fire up your local code editor on your desktop and open the qvsample.js file. You will make a change that will log out the language of the user□s operating system as reported to your application by the web browser



Java EE - C:\ibm-sample\qvsample.js - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Java EE
qvsample.js
Writable
Smart Insert
91 : 49

```
//*************************************************
//************* IMPORTANT **********************
//*************************************************
// You must add your Service Name to the following two statements for Cloudant to create a DB.  Change ibmID-DB to you database name

 if (tr.createDB("ibmID-qvDB", "qvsample")  == null) {console.log("Database qvsample failed creating");}
 if (tr.createDB("ibmID-qvDB", "qvmetering")  == null) {console.log("Database qvmetering failed creating");}

 // A few words about your use of Cloudant in IBM Cloud.  Initially, you are using the Cloudant Lite Plan which has restrictions.
 // Basically, you get around 10 reads, 5 writes per/second (which is not a lot.)  If you go over the limits your database
 // calls will fail and may disable all metering.  The way metering is written in this sample, care was taken so you would not
 // exceed those limits.  If you modify what gets logged and how much gets logged, or you use the database for other
 // purposes - keep in mind these limits because its easy to violate.  Also, you can upgrade from the Cloudant Lite plan
 // to the Cloudant standard plan, for just a little fee, and the standard plan should give you lots of headroom.


// Create an instance of a web server that receives requests from a web browser ('request') and
// returns a response ('response') to be rendered at the web browser
http.createServer(function (request, response) {

    var opEnvironment;
    var oplanguage;
    var browserCookies;
    var translatedURL;
    var ckDate;

//  console.log("Cookies=" + request.headers['cookie']);
//  console.log(request.headers);
    console.log("URL= " + request.url);


// Lets check for the browser's language,  operating system and any Cookies

    try {
    opEnvironment = request.headers['user-agent'];
    opLanguage = request.headers['accept-language'];
    browserCookies = request.headers['cookie'];
    }
    catch {}

// Until supported, force to English

//  console.log("Language=" + opLanguage);
    opLanguage = "en-US";
```

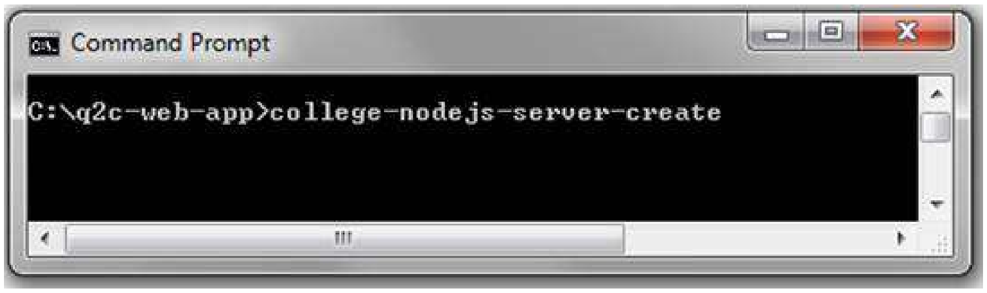
Locate the node.js statement in yellow and **uncomment** the statement.

Save the file

Step 8

In your shell, reissue the script again. This will stop the existing sample application in the Cloud and replace it with your updated version with the OS language from the browser being reported to the Application log.

Note: This illustrates an important point. Once you have generated your script commands you don�t have to regenerate them again. Just edit your sample application and reissue the same script! It is that easy!



```
C:\q2c-web-app>college-nodejs-server-create
```

In this part of the Course, it would be a good time to take a look at this running application from the perspective of the IBM Cloud Dashboard; a dashboard that is provided as part of the IBM user interface to their Cloud.

Continue