# Hello World Lab

## Resources

Lab code is on GitHub

<https://github.com/WatsonDevCloud/HelloWorld>

Bluemix web console

<https://console.ng.bluemix.net/>

## Introduction

**FILE: manifest.yml (should I discuss this here????)**

applications:

- services:

- Relationship Extraction-os

disk\_quota: 1024M

name: HelloWorld

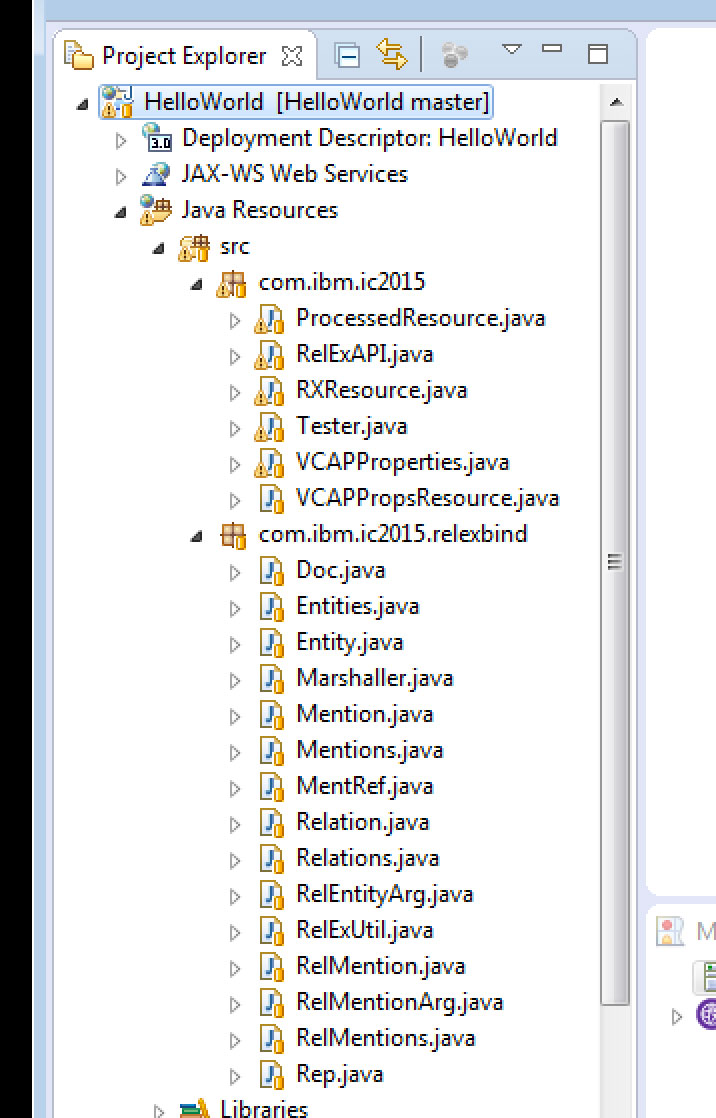
host: helloworldapp

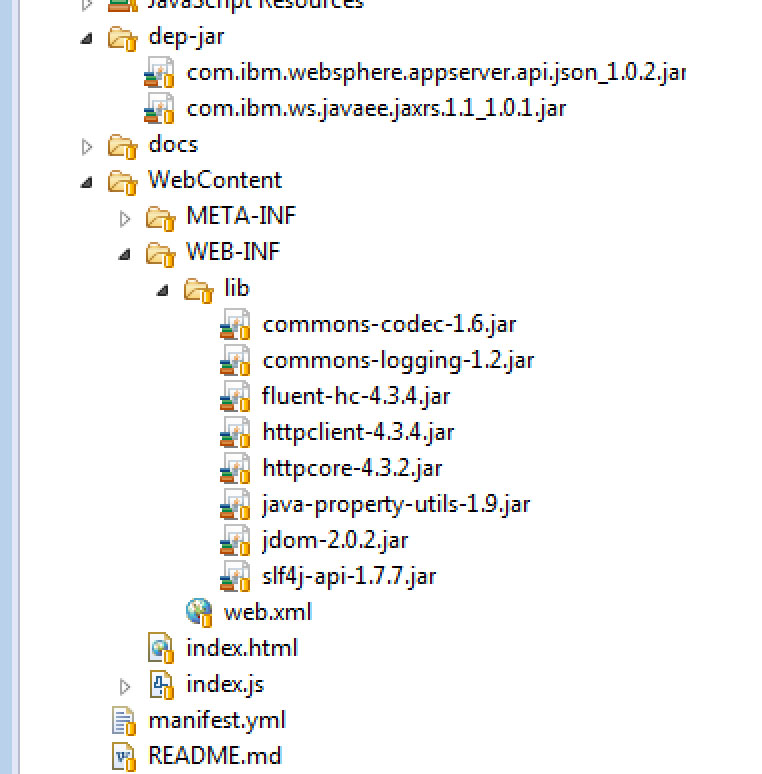
domain: mybluemix.net

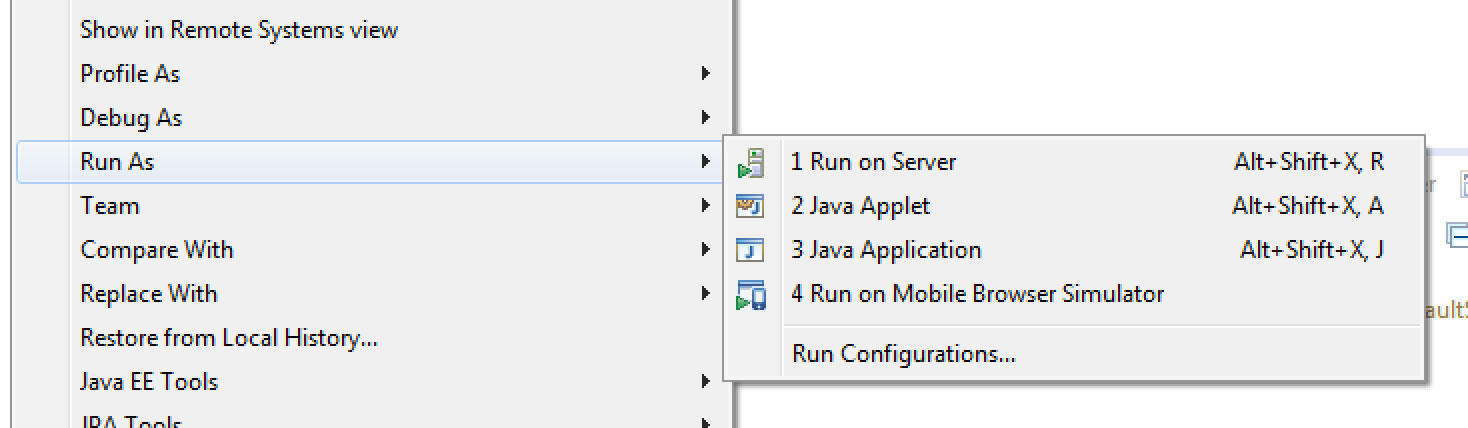
path: webStarterApp.war

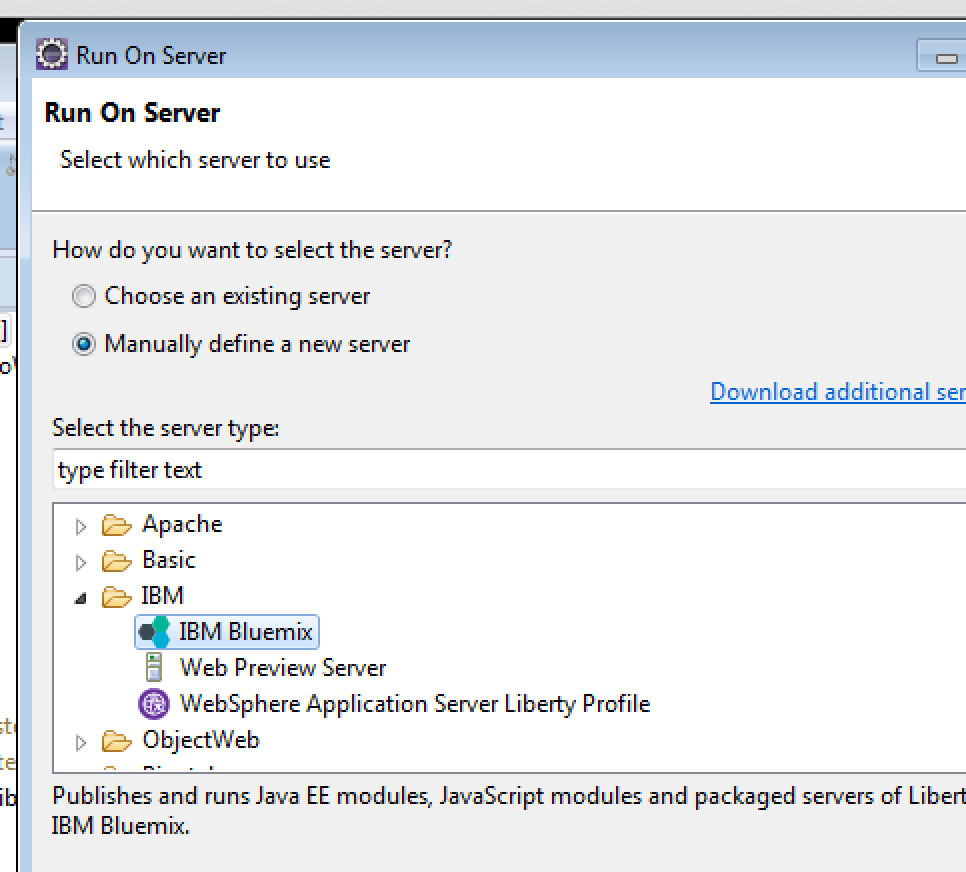
instances: 1

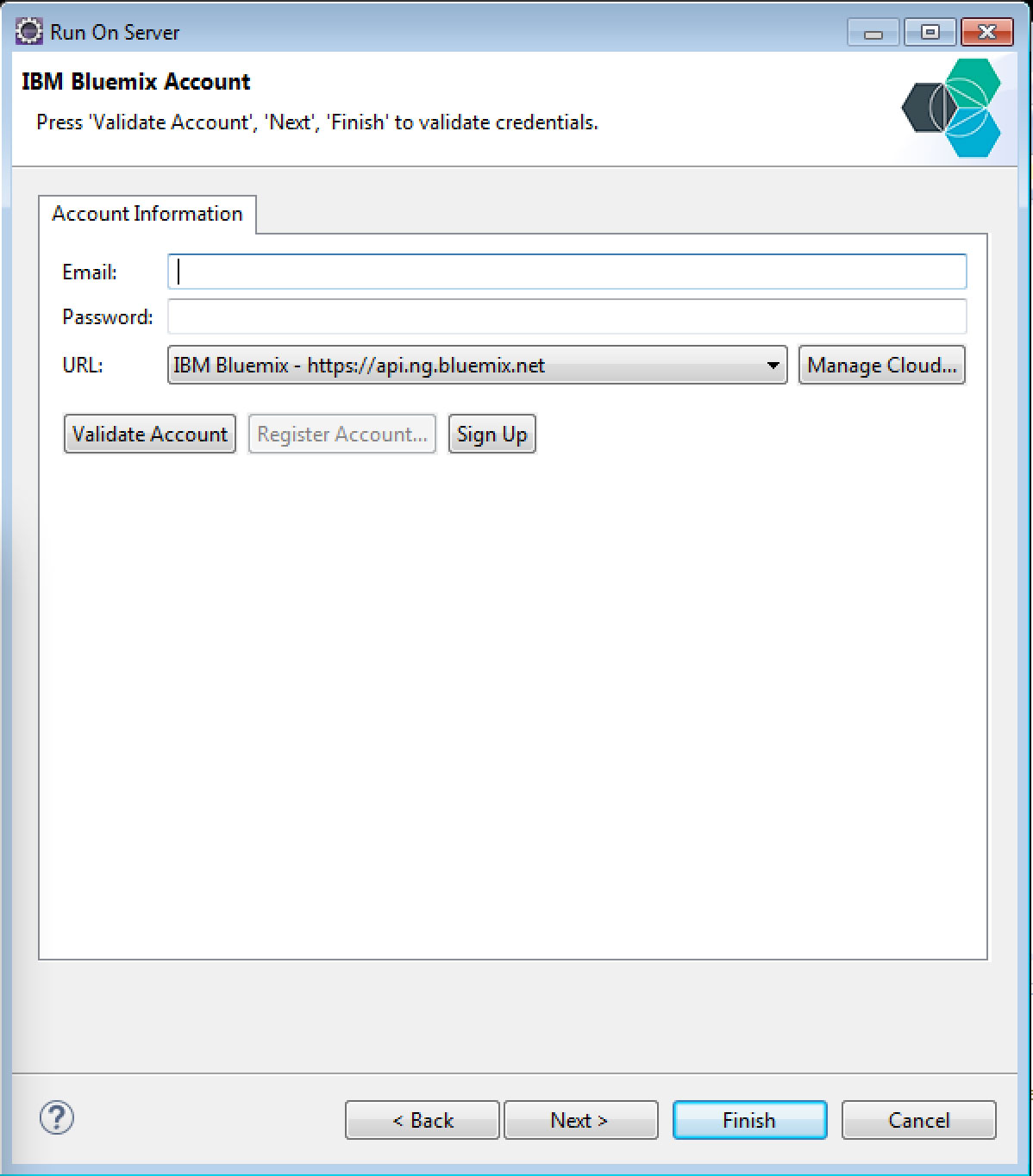
memory: 512M



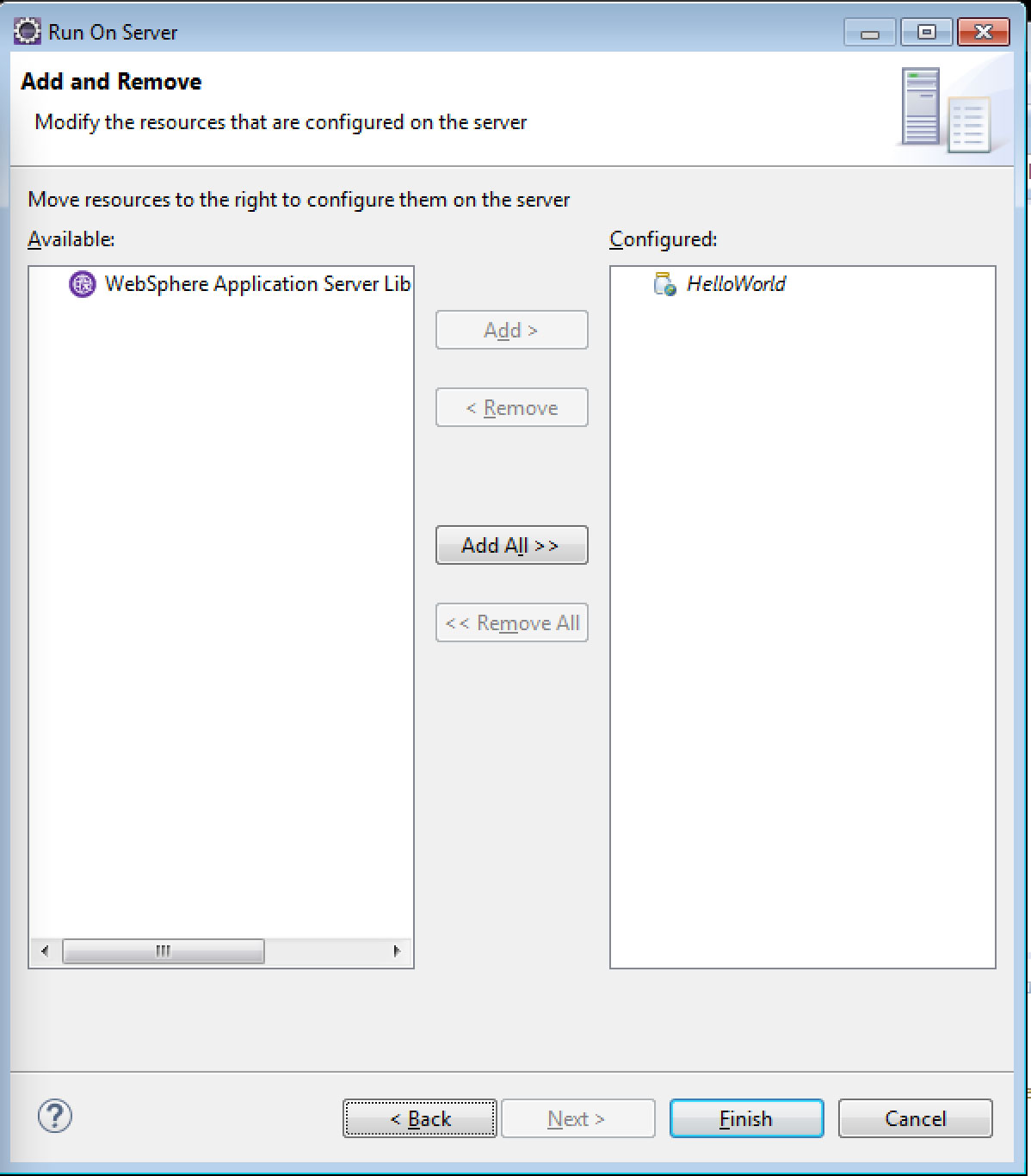


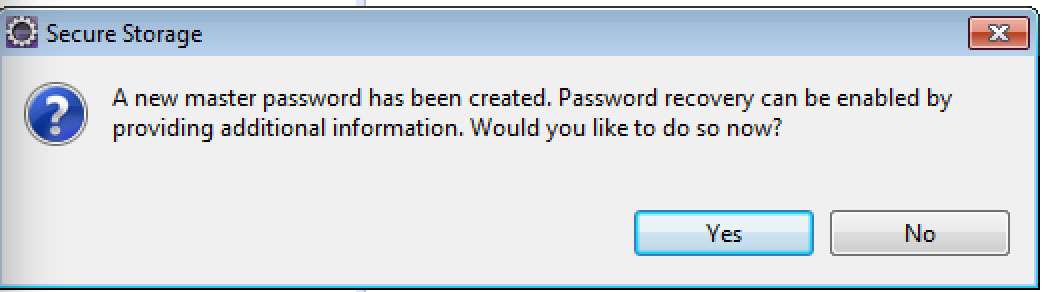




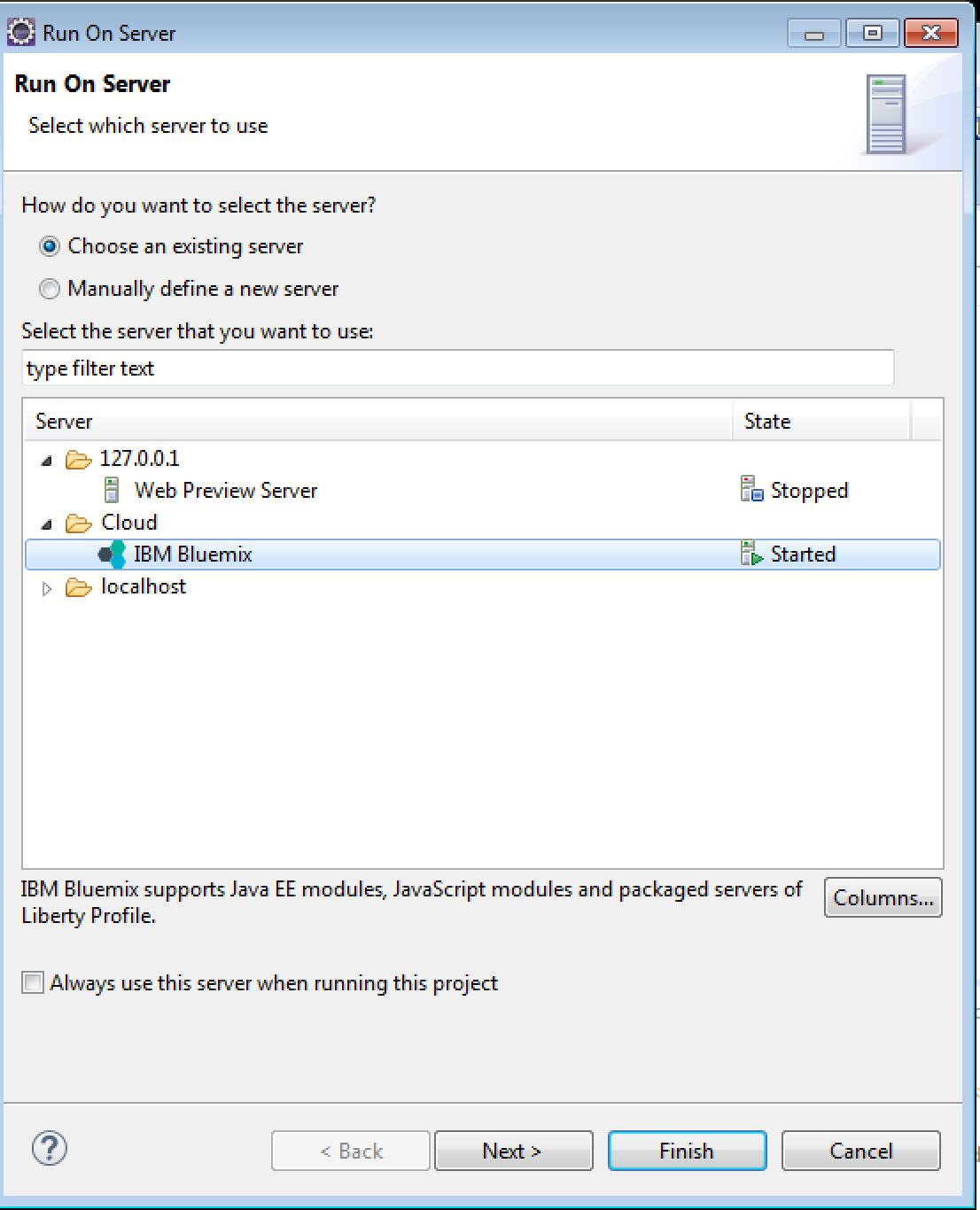


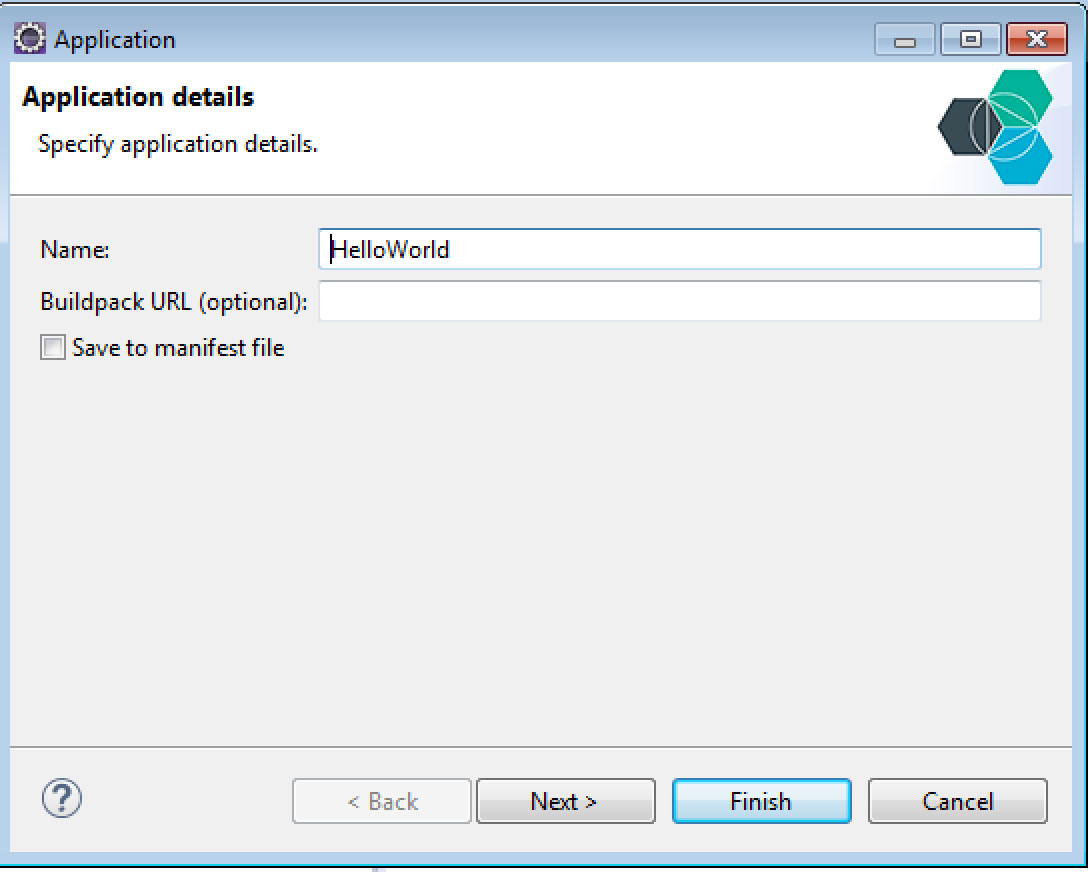


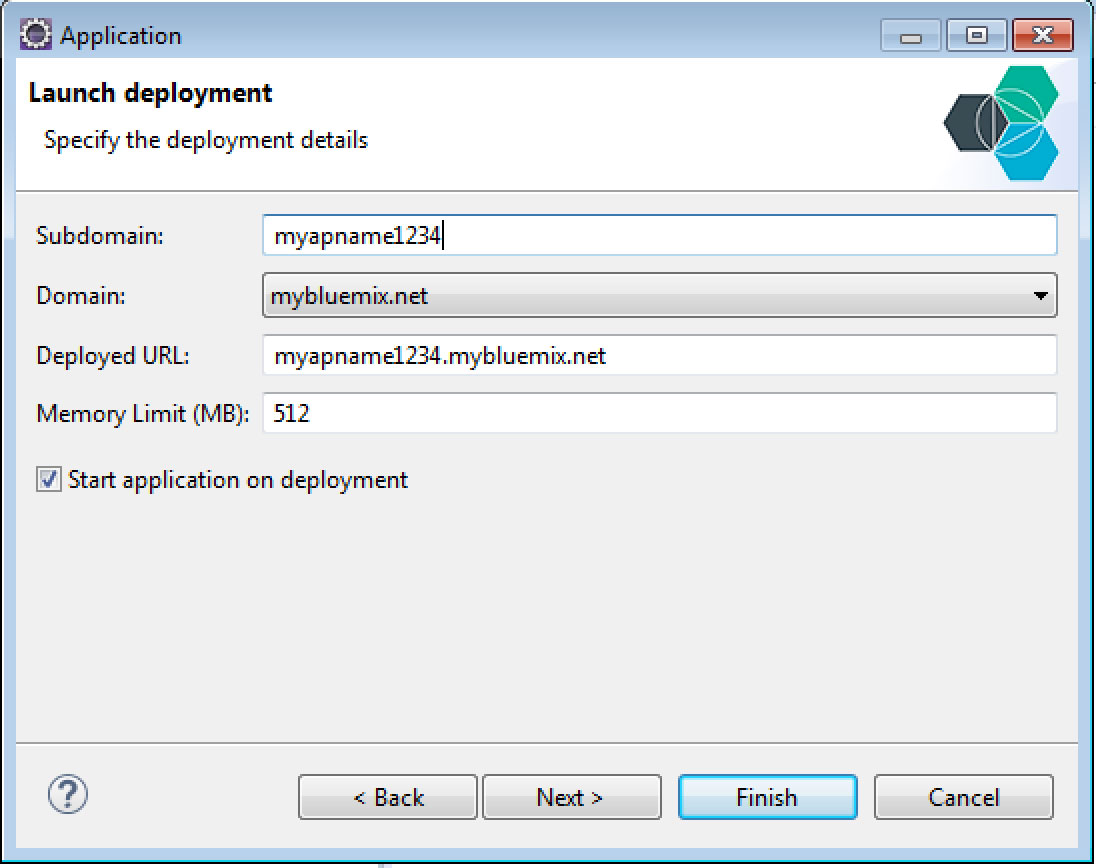


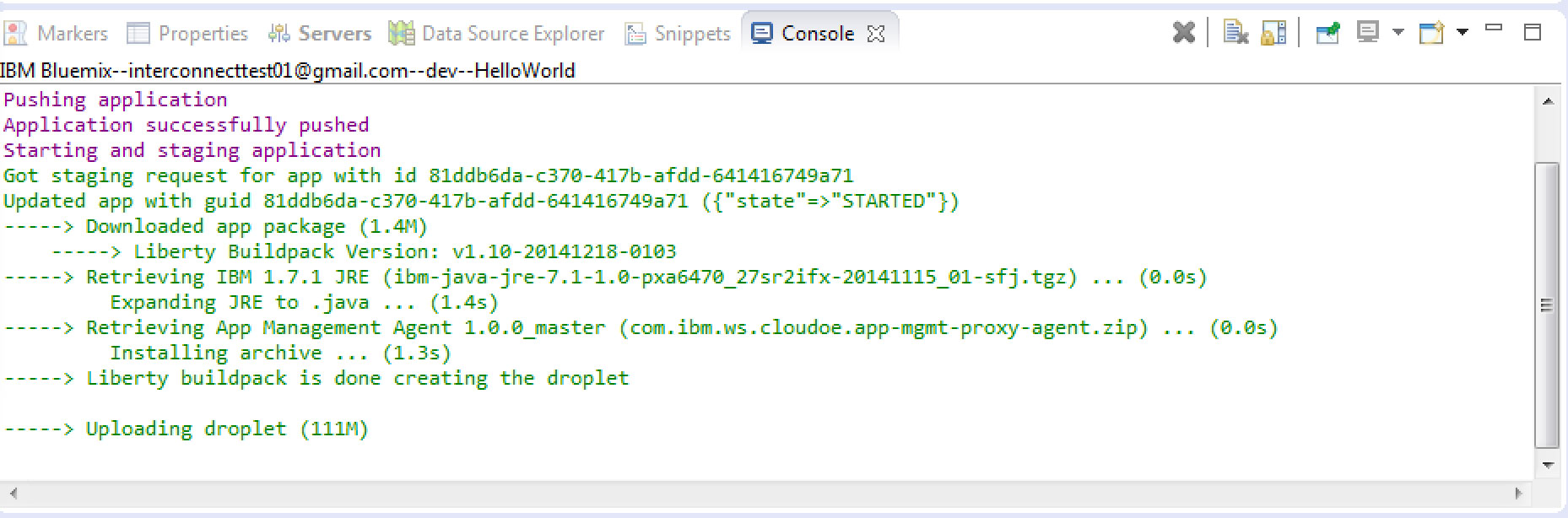


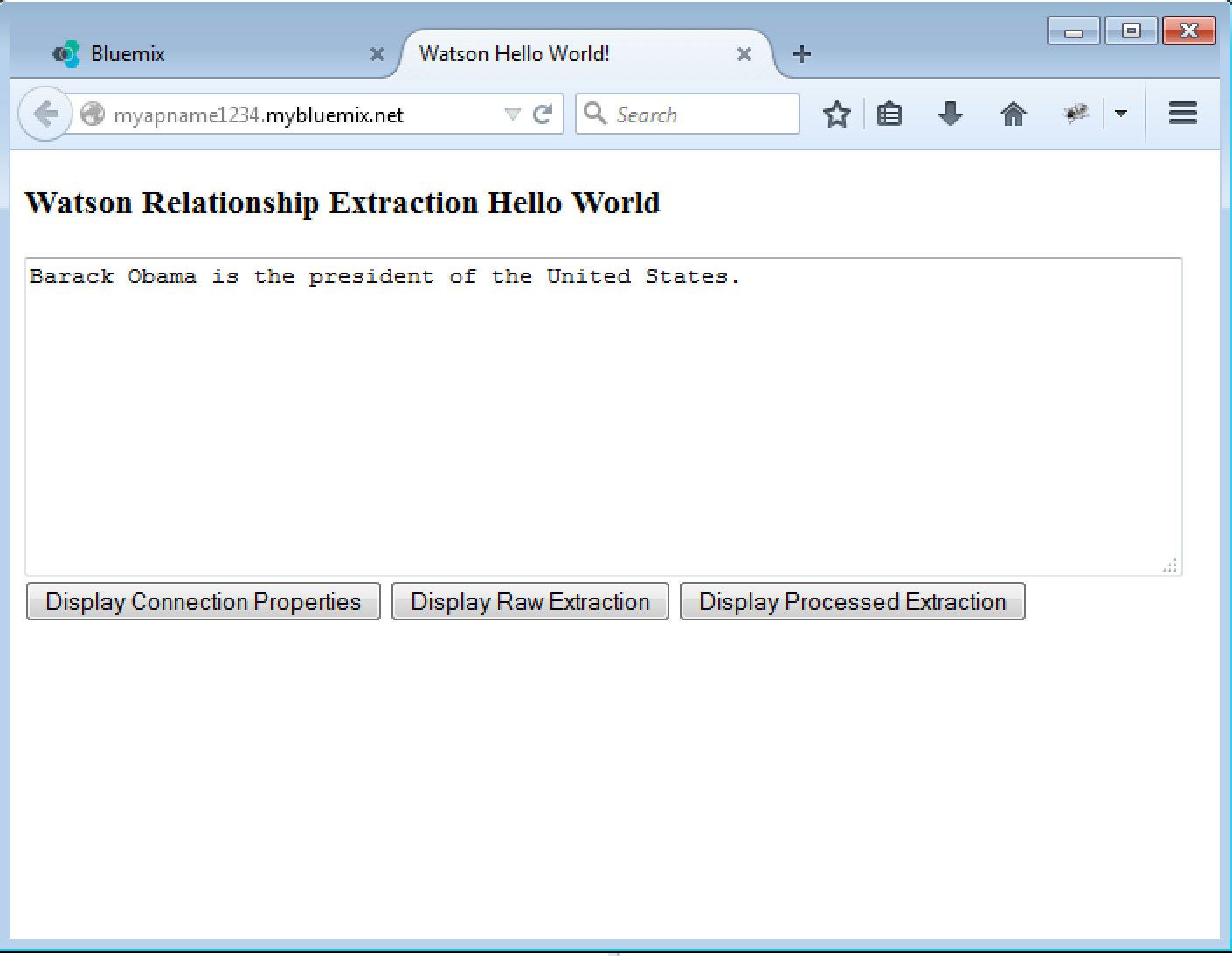
This image may be out of order. I think you see this only after you deploy again.

















**FILE: com/ibm/ic2015/VCAPProperties.java**

package com.ibm.ic2015;

import java.io.IOException;

import com.ibm.json.java.JSONArray;

import com.ibm.json.java.JSONObject;

public class VCAPProperties {

private String serviceName = "relationship\_extraction";

private String baseURL = "<service url>";

private String username = "<service username>";

private String password = "<service password>";

public VCAPProperties() {

processVCAP\_Services();

}

/\*\*

\* If exists, process the VCAP\_SERVICES environment variable in order to get the

\* username, password and baseURL

\*/

private void processVCAP\_Services() {

System.out.println("Processing VCAP\_SERVICES");

JSONObject sysEnv = getVcapServices();

if (sysEnv == null) return;

System.out.println("Looking for: "+ serviceName );

if (sysEnv.containsKey(serviceName)) {

JSONArray services = (JSONArray)sysEnv.get(serviceName);

JSONObject service = (JSONObject)services.get(0);

JSONObject credentials = (JSONObject)service.get("credentials");

baseURL = (String)credentials.get("url");

username = (String)credentials.get("username");

password = (String)credentials.get("password");

System.out.println("baseURL = "+baseURL);

System.out.println("username = "+username);

System.out.println("password = "+password);

} else {

System.out.println(serviceName + " is not available in VCAP\_SERVICES, "

+ "please bind the service to your application");

}

}

/\*\*

\* Gets the <b>VCAP\_SERVICES</b> environment variable and return it

\* as a JSONObject.

\*

\* @return the VCAP\_SERVICES as Json

\*/

private JSONObject getVcapServices() {

String envServices = System.getenv("VCAP\_SERVICES");

if (envServices == null) return null;

JSONObject sysEnv = null;

try {

sysEnv = JSONObject.parse(envServices);

} catch (IOException e) {

// Do nothing, fall through to defaults

System.out.println("Error parsing VCAP\_SERVICES: " + e.getMessage());

}

return sysEnv;

}

public String getBaseURL() {

return baseURL;

}

public String getUsername() {

return username;

}

public String getPassword() {

return password;

}

}

**FILE: com/ibm/ic2015/ VCAPPropsResource.java**

**package** com.ibm.ic2015;

**import** javax.ws.rs.GET;

**import** javax.ws.rs.Path;

**import** javax.ws.rs.Produces;

@Path("/vcapprops")

**public** **class** VCAPPropsResource {

@GET

@Produces("text/plain")

**public** String getVcapProperties() {

VCAPProperties vcp = **new** VCAPProperties();

String props = "URL: " + vcp.getBaseURL() +

"\nUsername: " + vcp.getUsername() +

"\nPassword: " + vcp.getPassword();

**return** props;

}

}

****

**FILE: com/ibm/ic2015/ RelExAPI.java**

package com.ibm.ic2015;

import java.io.IOException;

import java.net.URI;

import java.net.URISyntaxException;

import java.util.ArrayList;

import java.util.List;

import org.apache.http.NameValuePair;

import org.apache.http.client.ClientProtocolException;

import org.apache.http.client.fluent.Executor;

import org.apache.http.client.fluent.Request;

import org.apache.http.client.utils.URLEncodedUtils;

import org.apache.http.entity.ContentType;

import org.apache.http.message.BasicNameValuePair;

import com.ibm.ic2015.relexbind.Marshaller;

import com.ibm.ic2015.relexbind.Rep;

public class RelExAPI {

/\*

\* Using Apache HTTPClient library to call the Relationship Extraction REST service.

\* Results are an XML document.

\*/

public static String performExtraction(String text, String sid, String url, String username, String password) {

System.out.println("text: " + text);

System.out.println("sid: " + sid);

System.out.println("url: " + url);

System.out.println("username: " + username);

System.out.println("password: " + password);

List<NameValuePair> queryParams = new ArrayList<NameValuePair>();

queryParams.add(new BasicNameValuePair("txt", text ));

queryParams.add(new BasicNameValuePair("sid", sid ));

queryParams.add(new BasicNameValuePair("rt", "xml" ));

String response = "ERROR";

try {

Executor executor = Executor.newInstance().auth(username, password);

URI serviceURI = new URI(url).normalize();

byte[] responseBytes = null;

responseBytes = executor.execute(Request.Post(serviceURI)

.bodyString(URLEncodedUtils.format(queryParams, "utf-8"),

ContentType.APPLICATION\_FORM\_URLENCODED)

).returnContent().asBytes();

response = new String(responseBytes, "UTF-8");

} catch (URISyntaxException e) {

e.printStackTrace();

} catch (ClientProtocolException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

return response;

}

}

**FILE: com/ibm/ic2015/ Tester.java (step 1 of 3 )**

package com.ibm.ic2015;

import java.util.List;

import com.ibm.ic2015.relexbind.Marshaller;

import com.ibm.ic2015.relexbind.RelExUtil;

import com.ibm.ic2015.relexbind.Rep;

public class Tester {

/\*

\*

\* Tester for Relationship Extraction API so we don't have to deploy to Bluemix to test.

\*

\*/

public static void main(String[] args) {

String text = "Barack Obama is the president of the United States.";

String url = "https://gateway.watsonplatform.net/laser/service/api/v1/sire/cf997efc-3c5d-440d-9c54-885b04d56c4e";

String username = "ebe462b4-16d8-4ed7-9653-ad70999bb34c";

String password = "htbNjBXFzHpG";

String xml = RelExAPI.performExtraction(text,

"ie-en-news",

url,

username,

password);

System.out.println(xml);

}

}



**FILE: com/ibm/ic2015/ RXResource.java**

package com.ibm.ic2015;

import javax.ws.rs.FormParam;

import javax.ws.rs.POST;

import javax.ws.rs.Path;

import javax.ws.rs.Produces;

@Path("/rx")

public class RXResource {

@POST

@Produces("application/xml")

public String getRX(@FormParam("text") String text) {

System.out.println("Form text: " + text);

VCAPProperties vcp = new VCAPProperties();

String xml = RelExAPI.performExtraction(text,

"ie-en-news",

vcp.getBaseURL(),

vcp.getUsername(),

vcp.getPassword());

return xml;

}

}



**FILE: com/ibm/ic2015/ Tester.java (step 2 of 3)**

package com.ibm.ic2015;

import java.util.List;

import com.ibm.ic2015.relexbind.Marshaller;

import com.ibm.ic2015.relexbind.RelExUtil;

import com.ibm.ic2015.relexbind.Rep;

public class Tester {

/\*

\*

\* Tester for Relationship Extraction API so we don't have to deploy to Bluemix to test.

\*

\*/

public static void main(String[] args) {

String text = "Barack Obama is the president of the United States.";

String url = "https://gateway.watsonplatform.net/laser/service/api/v1/sire/cf997efc-3c5d-440d-9c54-885b04d56c4e";

String username = "ebe462b4-16d8-4ed7-9653-ad70999bb34c";

String password = "htbNjBXFzHpG";

String xml = RelExAPI.performExtraction(text,

"ie-en-news",

url,

username,

password);

System.out.println(xml);

Rep rep = Marshaller.marshallXml(xml);

System.out.println(rep);

}

}



**FILE: com/ibm/ic2015/ Tester.java (step 3 of 3)**

package com.ibm.ic2015;

import java.util.List;

import com.ibm.ic2015.relexbind.Marshaller;

import com.ibm.ic2015.relexbind.RelExUtil;

import com.ibm.ic2015.relexbind.Rep;

public class Tester {

/\*

\*

\* Tester for Relationship Extraction API so we don't have to deploy to Bluemix to test.

\*

\*/

public static void main(String[] args) {

String text = "Barack Obama is the president of the United States.";

String url = "https://gateway.watsonplatform.net/laser/service/api/v1/sire/cf997efc-3c5d-440d-9c54-885b04d56c4e";

String username = "ebe462b4-16d8-4ed7-9653-ad70999bb34c";

String password = "htbNjBXFzHpG";

String xml = RelExAPI.performExtraction(text,

"ie-en-news",

url,

username,

password);

System.out.println(xml);

Rep rep = Marshaller.marshallXml(xml);

System.out.println(rep);

RelExUtil util = new RelExUtil(rep);

List<String> entities = util.getEntitiesList();

List<String> rels = util.getRelationsList();

System.out.println(RelExUtil.stringFromList(entities));

System.out.println(RelExUtil.stringFromList(rels));

}

}



**FILE: com/ibm/ic2015/ ProcessedResource.java**

package com.ibm.ic2015;

import java.util.List;

import javax.ws.rs.FormParam;

import javax.ws.rs.POST;

import javax.ws.rs.Path;

import javax.ws.rs.Produces;

import com.ibm.ic2015.relexbind.Marshaller;

import com.ibm.ic2015.relexbind.RelExUtil;

import com.ibm.ic2015.relexbind.Rep;

@Path("/processed")

public class ProcessedResource {

@POST

@Produces("text/plain")

public String getProcessedRX(@FormParam("text") String text) {

System.out.println("Form text: " + text);

VCAPProperties vcp = new VCAPProperties();

String xml = RelExAPI.performExtraction(text,

"ie-en-news",

vcp.getBaseURL(),

vcp.getUsername(),

vcp.getPassword());

Rep rep = Marshaller.marshallXml(xml);

RelExUtil util = new RelExUtil(rep);

List<String> entities = util.getEntitiesList();

List<String> rels = util.getRelationsList();

return "Entities:\n" +

RelExUtil.stringFromList(entities) +

"\nRelations:\n" +

RelExUtil.stringFromList(rels);

}

}

