




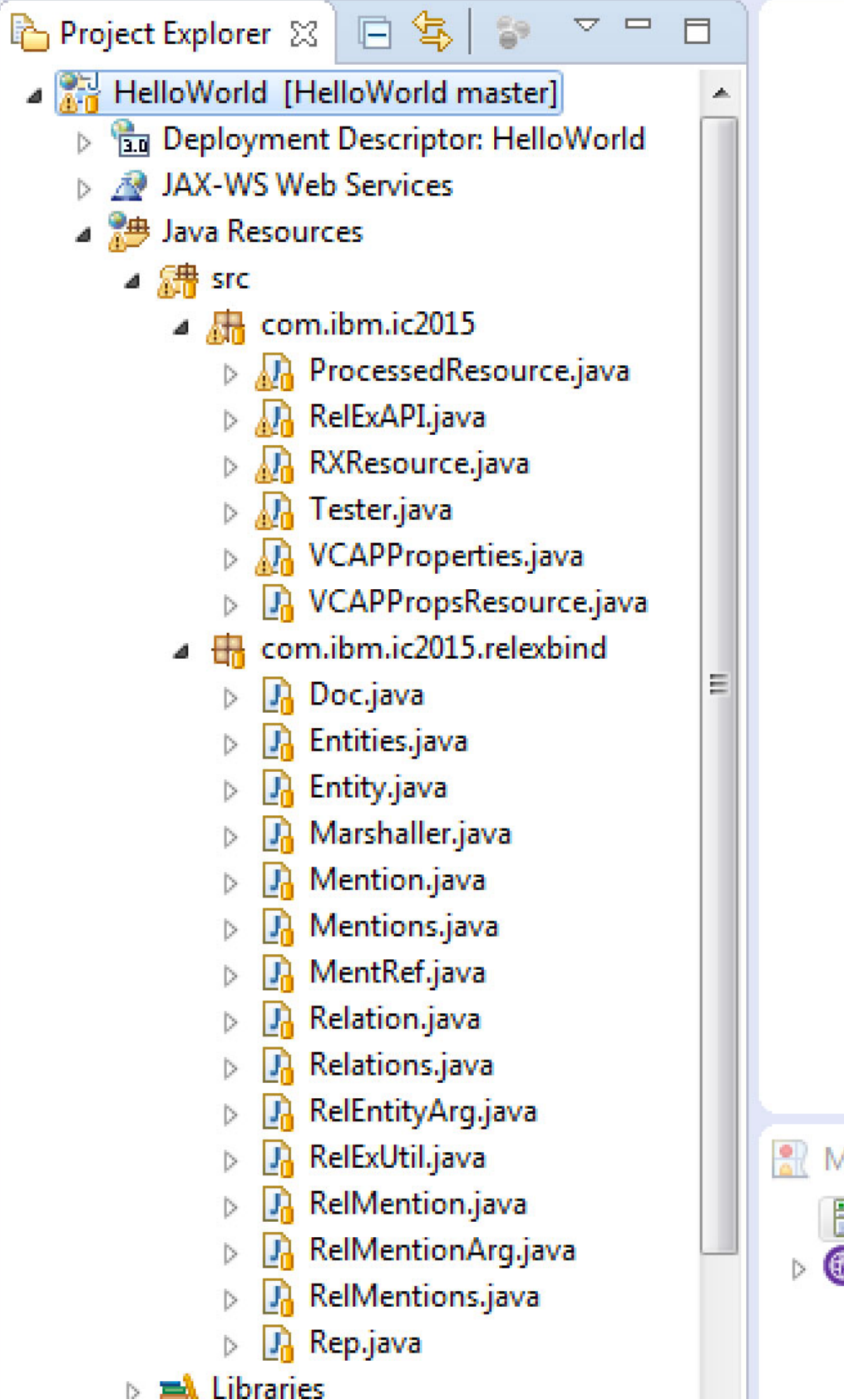
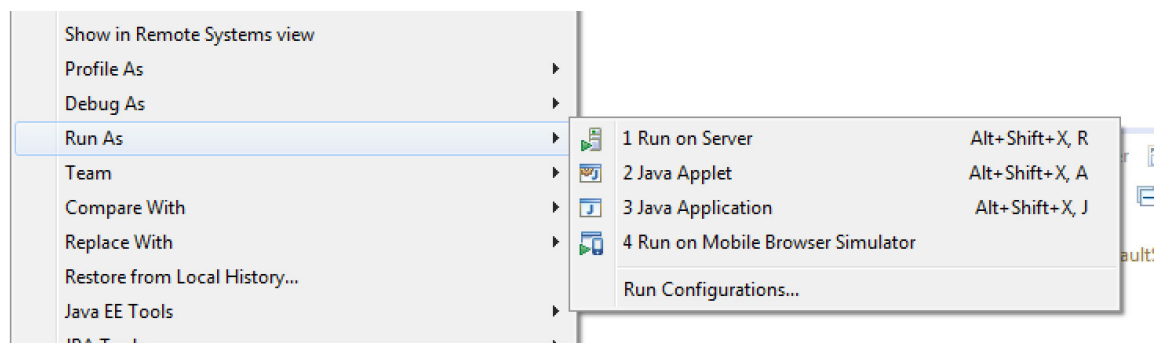
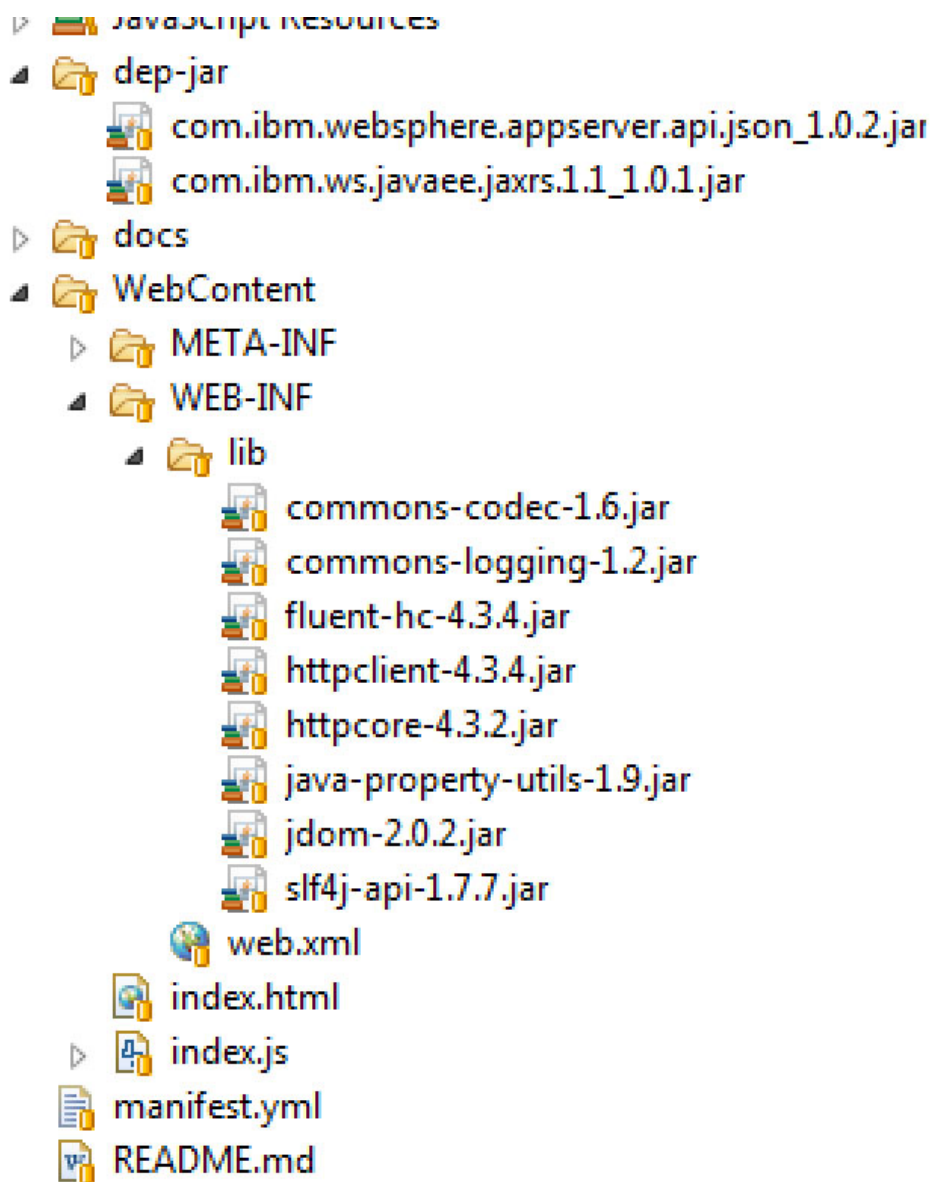


	Remote	▶
	Switch To	▶
	Advanced	▶
	Pull	
	Synchronize Workspace	
	Merge Tool	
	Merge...	







## Run On Server

### Run On Server

Select which server to use

How do you want to select the server?

- ☐ Choose an existing server
- ☒ Manually define a new server

[Download additional ser](#)

Select the server type:

type filter text

- ▶ Apache
- ▶ Basic
- ▶ IBM
  - ▶ IBM Bluemix
  - ▶ Web Preview Server
  - ▶ WebSphere Application Server Liberty Profile
- ▶ ObjectWeb

Publishes and runs Java EE modules, JavaScript modules and packaged servers of Liberty IBM Bluemix.

## IBM Bluemix Account

Press 'Validate Account', 'Next', 'Finish' to validate credentials.



### Account Information

Email:

Password:

URL:  [Manage Cloud...](#)

[Validate Account](#) [Register Account...](#) [Sign Up](#)



< Back

Next >

Finish

Cancel

## Organizations and Spaces

Press 'Validate Account', 'Next', 'Finish' to validate credentials.



### Organizations and Spaces:

▲ interconnecttest01@gmail.com

dev

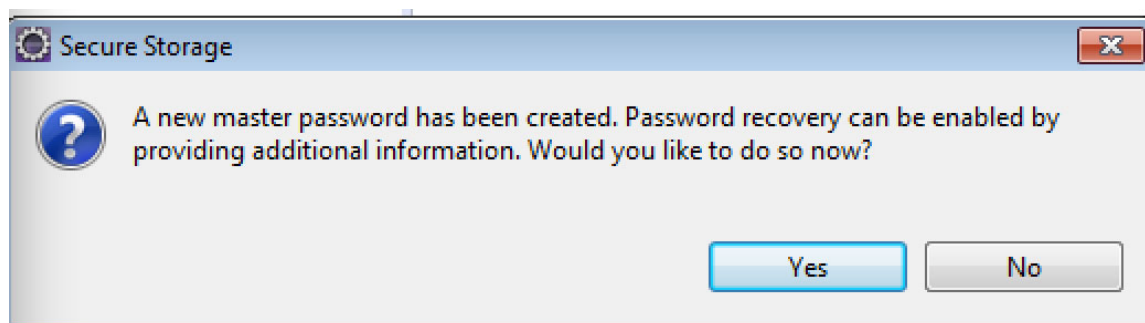
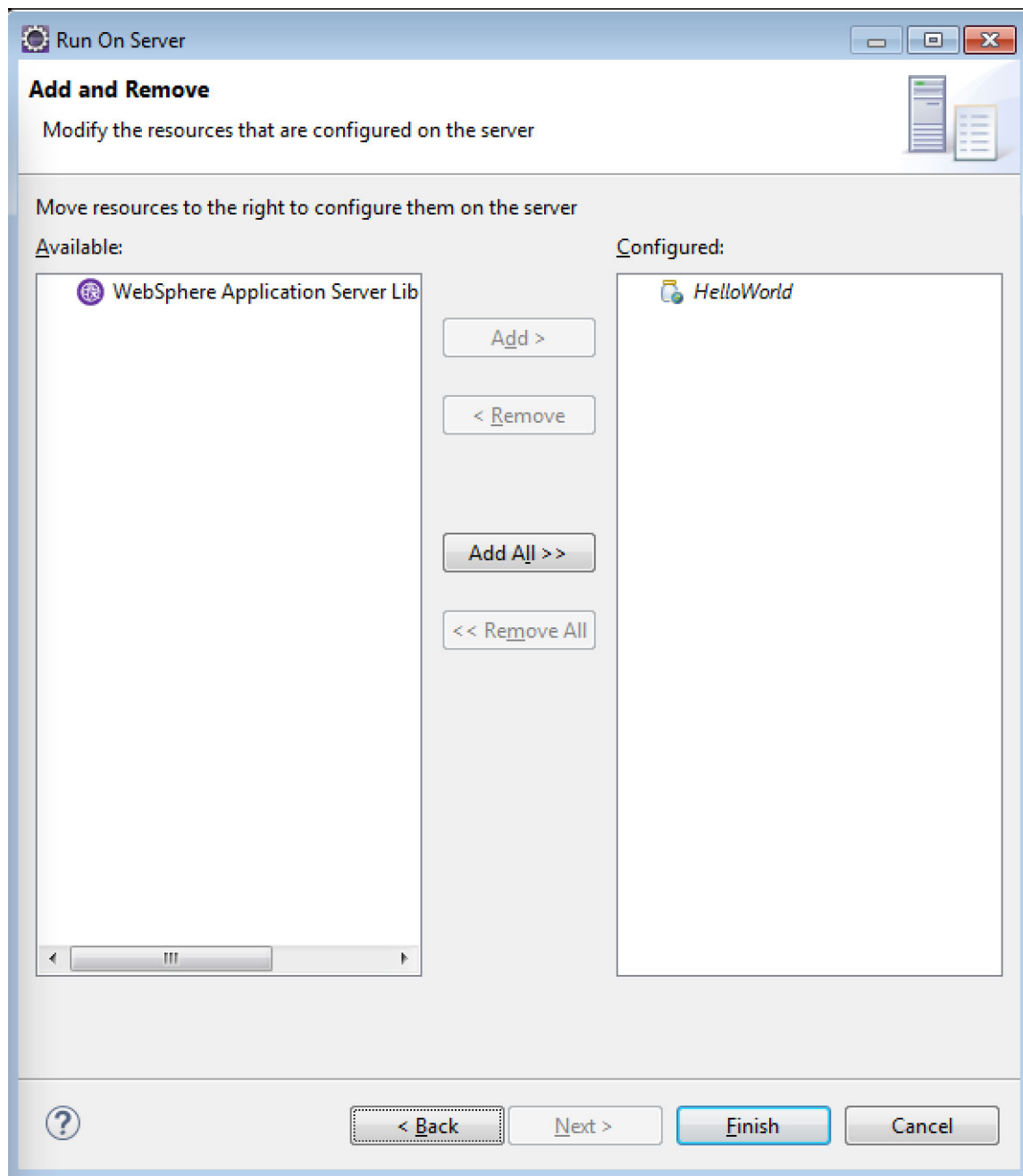


< Back

Next >

Finish

Cancel





Run On Server

Run On Server

Select which server to use

How do you want to select the server?

☒ Choose an existing server

☐ Manually define a new server

Select the server that you want to use:

type filter text

Server	State
127.0.0.1	
Web Preview Server	Stopped
Cloud	
IBM Bluemix	Started
localhost	

IBM Bluemix supports Java EE modules, JavaScript modules and packaged servers of Liberty Profile.

Columns...

☐ Always use this server when running this project

?

< Back

Next >

Finish

Cancel

Application

Application details

Specify application details.

Name:

HelloWorld

Buildpack URL (optional):

☐

 Save to manifest file

?

< Back

Next >

Finish

Cancel

Application

Launch deployment

Specify the deployment details

Subdomain:

myapname1234

Domain:

mybluemix.net

Deployed URL:

myapname1234.mybluemix.net

Memory Limit (MB):

512

☒

 Start application on deployment

?

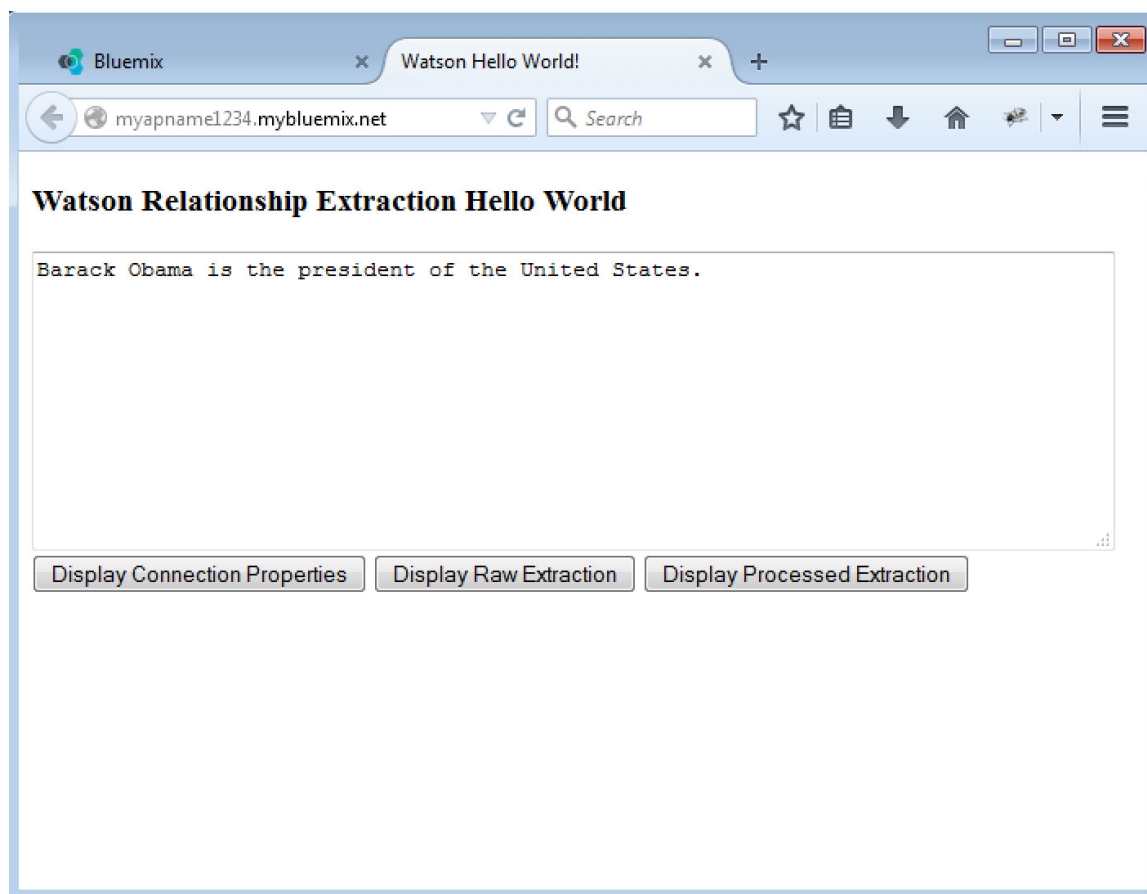
< Back

Next >

Finish

Cancel

```
IBM Bluemix--interconnecttest01@gmail.com--dev--HelloWorld
Pushing application
Application successfully pushed
Starting and staging application
Got staging request for app with id 81ddb6da-c370-417b-afdd-641416749a71
Updated app with guid 81ddb6da-c370-417b-afdd-641416749a71 ({"state"=>"STARTED"})
-----> Downloaded app package (1.4M)
-----> Liberty Buildpack Version: v1.10-20141218-0103
-----> Retrieving IBM 1.7.1 JRE (ibm-java-jre-7.1-1.0-pxa6470_27sr2ifx-20141115_01-sfj.tgz) ... (0.0s)
      Expanding JRE to .java ... (1.4s)
-----> Retrieving App Management Agent 1.0.0_master (com.ibm.ws.cloudoe.app-mgmt-proxy-agent.zip) ... (0.0s)
      Installing archive ... (1.3s)
-----> Liberty buildpack is done creating the droplet
-----> Uploading droplet (111M)
```



## 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");
        }
    }
}
```

## FILE: com/ibm/ic2015/VCAPProperties.java (continued)

```
/**
 * 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/ VCAPPPropsResource.java

```
package com.ibm.ic2015;

import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.Produces;

@Path("/vcapprops")
public class VCAPPPropsResource {

    @GET
    @Produces("text/plain")
    public String getVcapProperties() {

        VCAPPProperties vcp = new VCAPPProperties();

        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 (part 1)

```
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/ Tester.java (part 2)

```
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 (final)

```
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/ 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);

        VCAPPProperties vcp = new VCAPPProperties();

        String xml = RelExAPI.performExtraction(text,
                                                "ie-en-news",
                                                vcp.getBaseURL(),
                                                vcp.getUsername(),
                                                vcp.getPassword());

        return xml;
    }
}
```

## 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);

        VCAPPProperties vcp = new VCAPPProperties();

        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);

    }

}
```