RESTful Web Services

Introduction

- The `RESTful` web services are highly popular in software architecture for distributed systems on the web.
- The Representational State Transfer (REST) web services set a new style fro SOA systems.
- The term Representational State Transfer was introduced in 2000 by Roy Fielding in his doctoral dissertation.

The REStful Services Architecture

- REST stands for Representational State Transfer. It is a web service architecture that focuses on the system resources.
- Each resource is identified by a URI (Unified Resource Identifier).
- Accessing the resources is done using HTTP. The server reply is the representation of the resource we try to access. This representation is usually an XML document.

The RESTful Services Architecture

- Web Services that use the REST architecture are called RESTful Services.
- Systems that use or provide RESTful services are usually referred to as RESTafarians.

  One of the most popular RESTful web service is the RSS feed many blogs usually provide.
- Most RESTful services are being used via a simple HTTP GET request for which they reply with an XML document.

REStful Services Sample

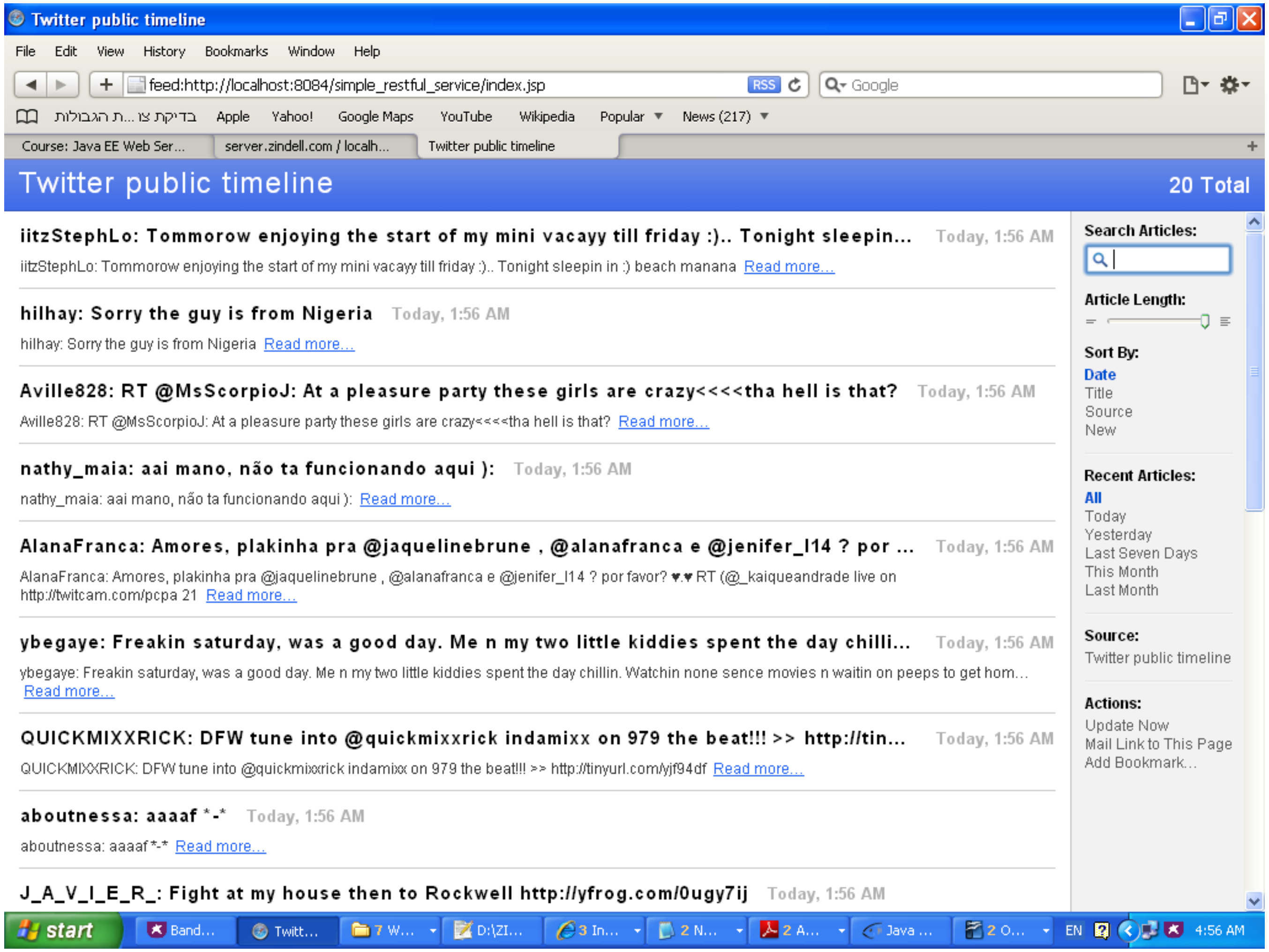
```
<%@ page contentType="text/xml; charset=UTF-8" %>
<%@ page import="java.io.BufferedReader,
java.io.IOException,java.io.InputStreamReader,
java.net.MalformedURLException,java.net.URL, java.net.URLConnection" %>

<%
try
{
        URL twitter = new
        URL("http://twitter.com/statuses/public_timeline.rss");
        URLConnection connection = twitter.openConnection();
        BufferedReader in = new BufferedReader(new
        InputStreamReader(connection.getInputStream()));
        String str;
        while ((str = in.readLine()) != null)
        {
            out.println(str);
        }
        in.close();
```

RESTful Services Sample

```
catch (MalformedURLException e)
{
    e.printStackTrace();
}
catch (IOException e)
{
    e.printStackTrace();
}
%>
```

RESTful Services Sample


Twitter public timeline
File Edit View History Bookmarks Window Help
feed:http://localhost:8084/simple_restful_service/index.jsp
Course: Java EE Web Ser... server.zindell.com / localh... Twitter public timeline

Twitter public timeline 20 Total

iitzStephLo: Tommorow enjoying the start of my mini vacayy till friday :).. Tonight sleepin... Today, 1:56 AM
iitzStephLo: Tommorow enjoying the start of my mini vacayy till friday :).. Tonight sleepin in :) beach manana Read more...

hilhay: Sorry the guy is from Nigeria Today, 1:56 AM
hilhay: Sorry the guy is from Nigeria Read more...

Aville828: RT @MsScorpioJ: At a pleasure party these girls are crazy<<<<tha hell is that? Today, 1:56 AM
Aville828: RT @MsScorpioJ: At a pleasure party these girls are crazy<<<<tha hell is that? Read more...

nathy_maia: aai mano, não ta funcionando aqui): Today, 1:56 AM
nathy_maia: aai mano, não ta funcionando aqui): Read more...

AlanaFranca: Amores, plakinha pra @jaquelinebrune , @alanafranca e @jenifer_l14 ? por ... Today, 1:56 AM
AlanaFranca: Amores, plakinha pra @jaquelinebrune , @alanafranca e @jenifer_l14 ? por favor? ♥.♥ RT (@_kaiqueandrade live on http://twitcam.com/pcpa 21 Read more...

ybegaye: Freakin saturday, was a good day. Me n my two little kiddies spent the day chilli... Today, 1:56 AM
ybegaye: Freakin saturday, was a good day. Me n my two little kiddies spent the day chillin. Watchin none sence movies n waitin on peeps to get hom... Read more...

QUICKMIXXRICK: DFW tune into @quickmixxrick indamixx on 979 the beat!!! >> http://tin... Today, 1:56 AM
QUICKMIXXRICK: DFW tune into @quickmixxrick indamixx on 979 the beat!!! >> http://tinyurl.com/yjf94df Read more...

aboutnessa: aaaaf *-* Today, 1:56 AM
aboutnessa: aaaaf *-* Read more...

J_A_V_I_E_R_: Fight at my house then to Rockwell http://yfrog.com/0ugy7ij Today, 1:56 AM

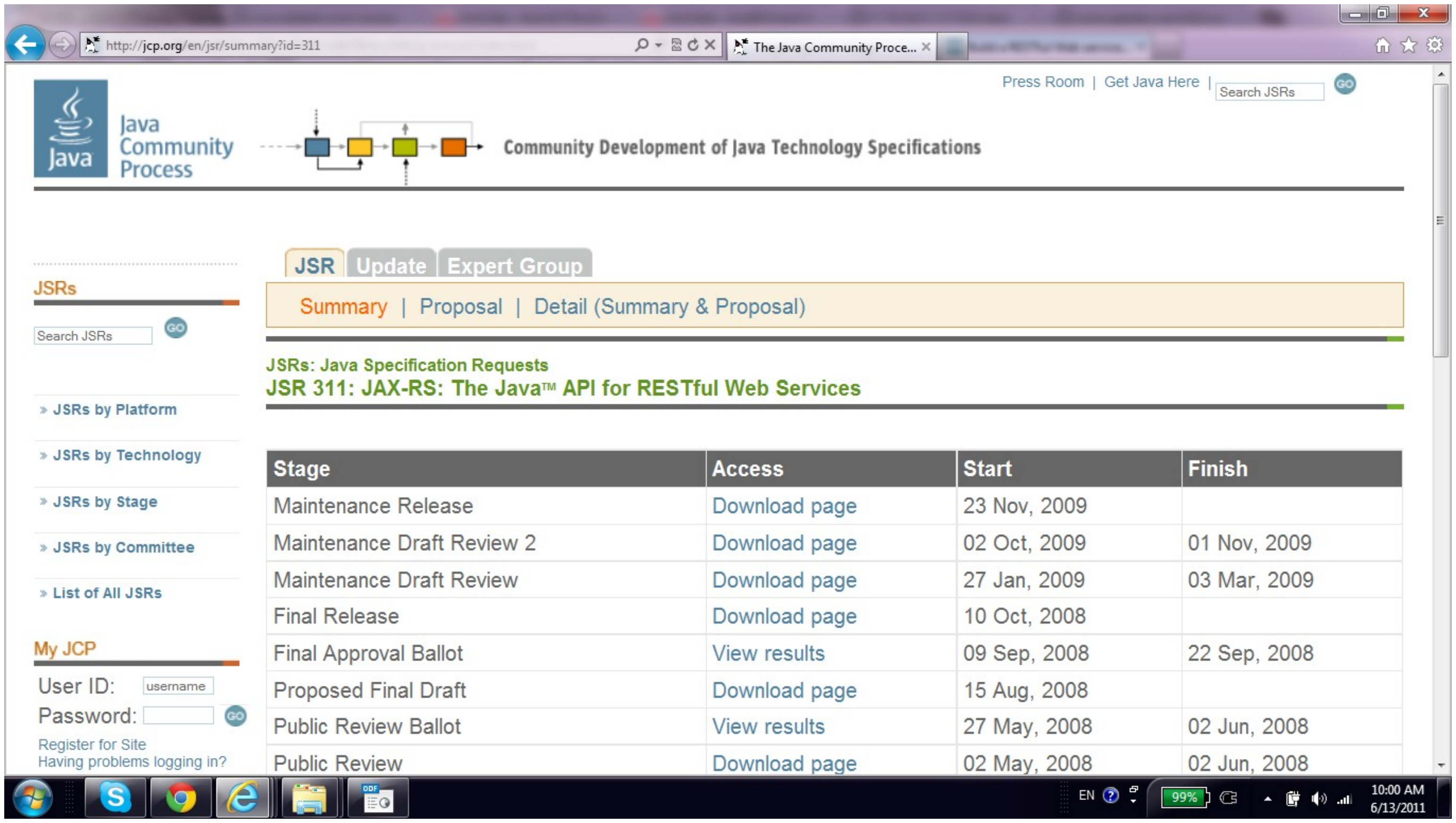
Search Articles:
Article Length:
Sort By: Date Title Source New
Recent Articles: All Today Yesterday Last Seven Days This Month Last Month
Source: Twitter public timeline
Actions: Update Now Mail Link to This Page Add Bookmark...
start
4:56 AM

© 2008 Haim Michael

Java API for RESTful Web Services

- JSR 311 specifies Java API for RESTful web services. It aims at allowing us to develop RESTful web services in a standard way.

Java API for RESTful Web Services



http://jcp.org/en/jsr/summary?id=311

Press Room | Get Java Here | Search JSRs

Java Community Process

Community Development of Java Technology Specifications

JSRs

Search JSRs

- JSRs by Platform
- JSRs by Technology
- JSRs by Stage
- JSRs by Committee
- List of All JSRs

My JCP

User ID: username

Password:

Register for Site

Having problems logging in?

JSR | Update | Expert Group

Summary | Proposal | Detail (Summary & Proposal)

JSRs: Java Specification Requests

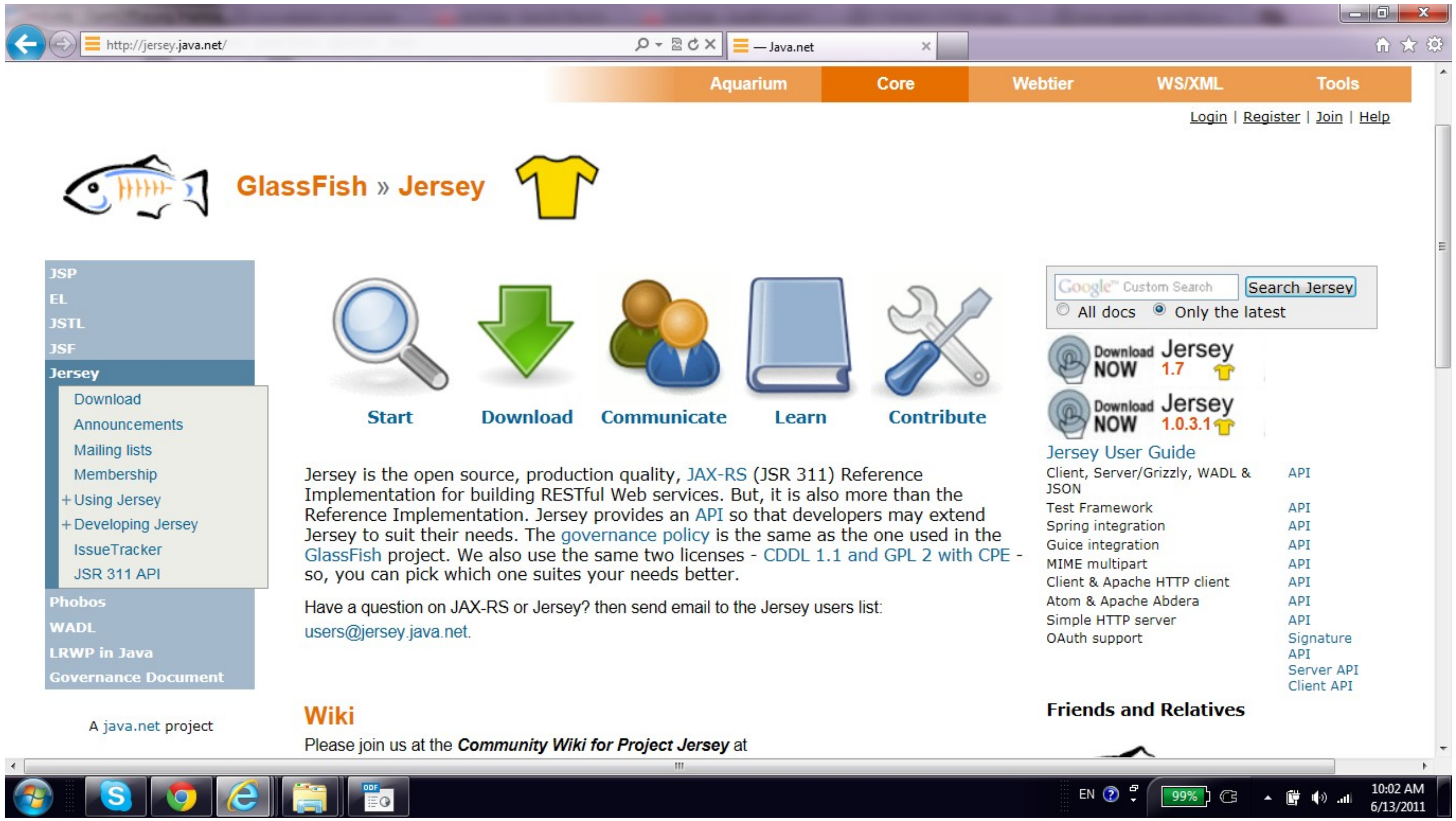
JSR 311: JAX-RS: The Java™ API for RESTful Web Services

Stage	Access	Start	Finish
Maintenance Release	Download page	23 Nov, 2009	
Maintenance Draft Review 2	Download page	02 Oct, 2009	01 Nov, 2009
Maintenance Draft Review	Download page	27 Jan, 2009	03 Mar, 2009
Final Release	Download page	10 Oct, 2008	
Final Approval Ballot	View results	09 Sep, 2008	22 Sep, 2008
Proposed Final Draft	Download page	15 Aug, 2008	
Public Review Ballot	View results	27 May, 2008	02 Jun, 2008
Public Review	Download page	02 May, 2008	02 Jun, 2008

Jersey Project

- Jersey is the reference implementation for JSR 311. We can easily integrate Jersey with Tomcat in order to develop a RESTful web service.

Jersey Project



Jersey is the open source, production quality, JAX-RS (JSR 311) Reference Implementation for building RESTful Web services. But, it is also more than the Reference Implementation. Jersey provides an API so that developers may extend Jersey to suit their needs. The governance policy is the same as the one used in the GlassFish project. We also use the same two licenses - CDDL 1.1 and GPL 2 with CPE - so, you can pick which one suites your needs better.

Have a question on JAX-RS or Jersey? then send email to the Jersey users list: users@jersey.java.net.

Wiki

Please join us at the ***Community Wiki for Project Jersey*** at

Jersey User Guide

Client, Server/Grizzly, WADL & JSON	API
Test Framework	API
Spring integration	API
Guice integration	API
MIME multipart	API
Client & Apache HTTP client	API
Atom & Apache Abdera	API
Simple HTTP server	API
OAuth support	Signature API Server API Client API

Friends and Relatives

Download Jersey

- You can easily download Jersey latest version browsing at

  http://download.java.net/maven/2/com/sun/jersey/jersey-archive/1.7/jersey-archive-1.7.zip

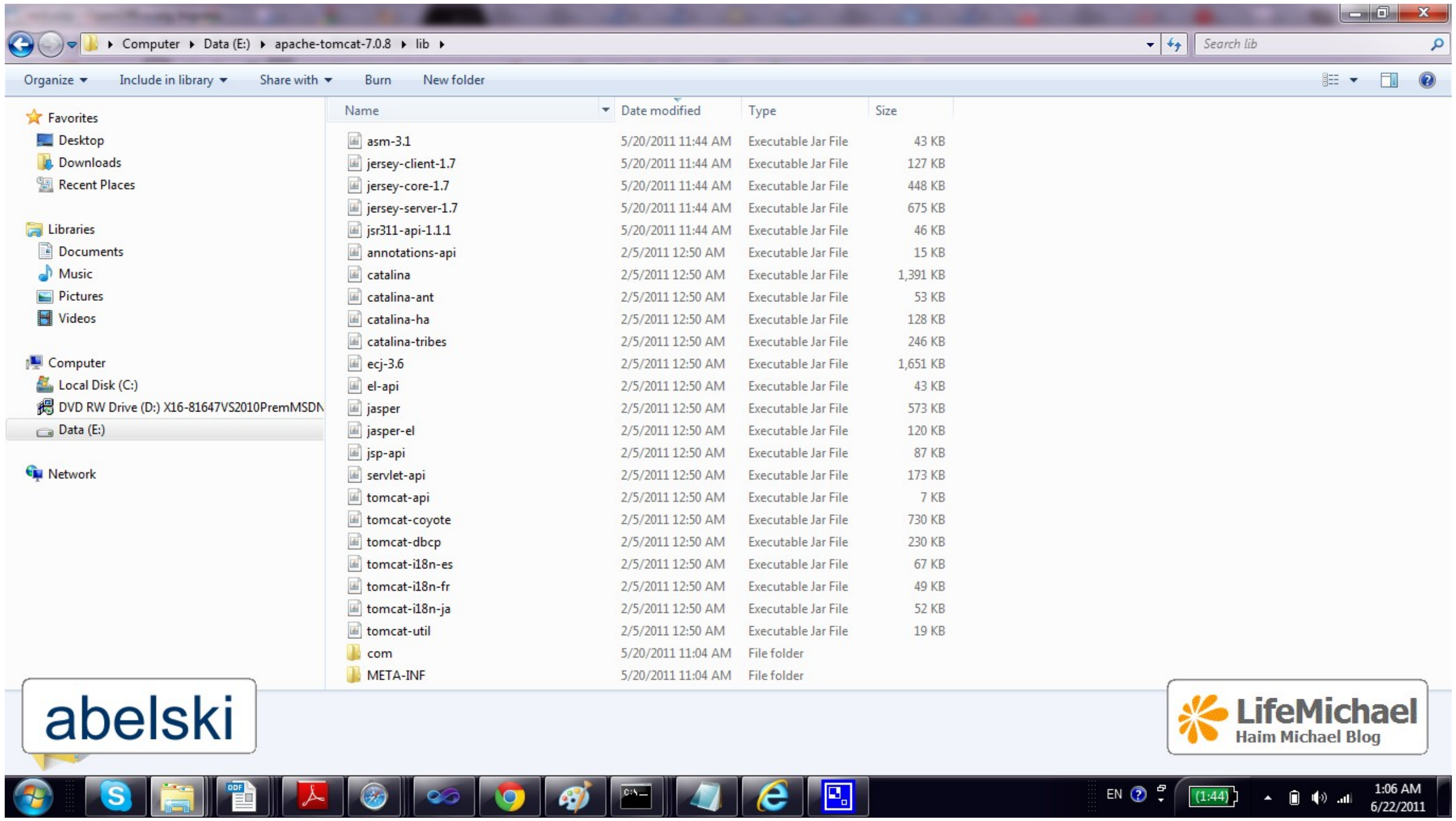
- In order to deploy a simple Jersey based RESTful web service we will need the following files:

  ```
  jersey-core.jar
  jersey-server.jar
  jsr311-api.jar
  asm.jar
  ```

Install Jersey

- We should first copy these jar files into the `lib` folder of our Tomcat installation.

Install Jersey



Name	Date modified	Type	Size
asm-3.1	5/20/2011 11:44 AM	Executable Jar File	43 KB
jersey-client-1.7	5/20/2011 11:44 AM	Executable Jar File	127 KB
jersey-core-1.7	5/20/2011 11:44 AM	Executable Jar File	448 KB
jersey-server-1.7	5/20/2011 11:44 AM	Executable Jar File	675 KB
jsr311-api-1.1.1	5/20/2011 11:44 AM	Executable Jar File	46 KB
annotations-api	2/5/2011 12:50 AM	Executable Jar File	15 KB
catalina	2/5/2011 12:50 AM	Executable Jar File	1,391 KB
catalina-ant	2/5/2011 12:50 AM	Executable Jar File	53 KB
catalina-ha	2/5/2011 12:50 AM	Executable Jar File	128 KB
catalina-tribes	2/5/2011 12:50 AM	Executable Jar File	246 KB
ecj-3.6	2/5/2011 12:50 AM	Executable Jar File	1,651 KB
el-api	2/5/2011 12:50 AM	Executable Jar File	43 KB
jasper	2/5/2011 12:50 AM	Executable Jar File	573 KB
jasper-el	2/5/2011 12:50 AM	Executable Jar File	120 KB
jsp-api	2/5/2011 12:50 AM	Executable Jar File	87 KB
servlet-api	2/5/2011 12:50 AM	Executable Jar File	173 KB
tomcat-api	2/5/2011 12:50 AM	Executable Jar File	7 KB
tomcat-coyote	2/5/2011 12:50 AM	Executable Jar File	730 KB
tomcat-dbcp	2/5/2011 12:50 AM	Executable Jar File	230 KB
tomcat-i18n-es	2/5/2011 12:50 AM	Executable Jar File	67 KB
tomcat-i18n-fr	2/5/2011 12:50 AM	Executable Jar File	49 KB
tomcat-i18n-ja	2/5/2011 12:50 AM	Executable Jar File	52 KB
tomcat-util	2/5/2011 12:50 AM	Executable Jar File	19 KB
com	5/20/2011 11:04 AM	File folder	
META-INF	5/20/2011 11:04 AM	File folder	

Deployment

❖ We should define the Jersey servlet dispatcher in our `web.xml` file.

Deployment

```
<servlet>
    <servlet-name>My Jersey REST Service</servlet-name>
    <servlet-class>
        com.sun.jersey.spi.container.servlet.ServletContainer
    </servlet-class>
    <init-param>
        <param-name>
            com.sun.jersey.config.property.packages
        </param-name>
        <param-value>
            com.abelski.samples
        </param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>My Jersey REST Service</servlet-name>
    <url-pattern>/rest/*</url-pattern>
</servlet-mapping>
```

this param defines the package where jersey should look for the classes we defined

we should add it into our web.xml configuration file

Deployment

- We can develop the class that will be instantiated and function as a REStful web service.

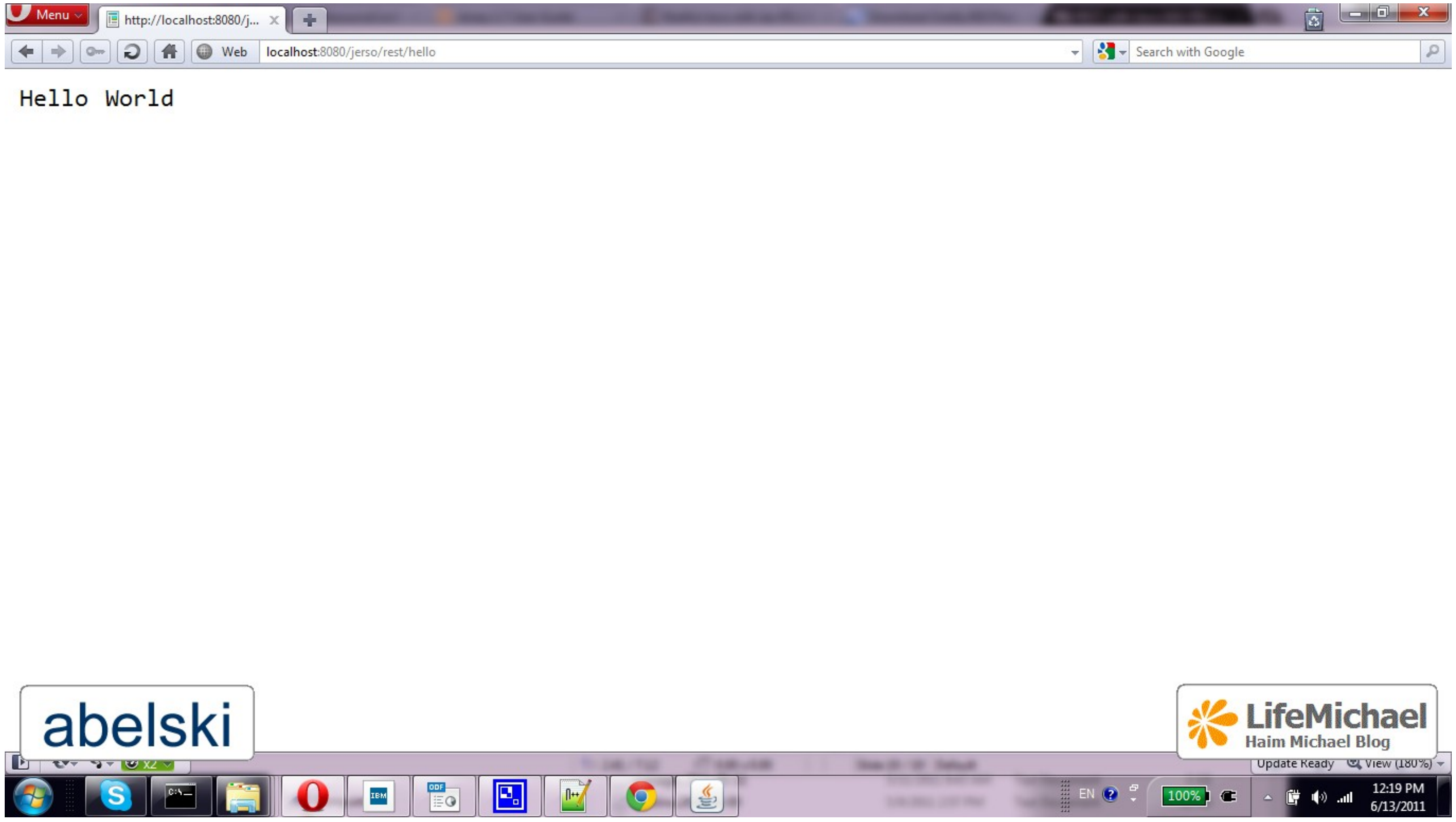
Deployment

```
package com.abelski.samples;

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

@Path("/hello")
public class HelloWorld
{
    @GET
    @Produces(MediaType.TEXT_PLAIN)
    public String sayHello()
    {
        return "Hello World";
    }
}
```

Deployment



```
Hello World
```

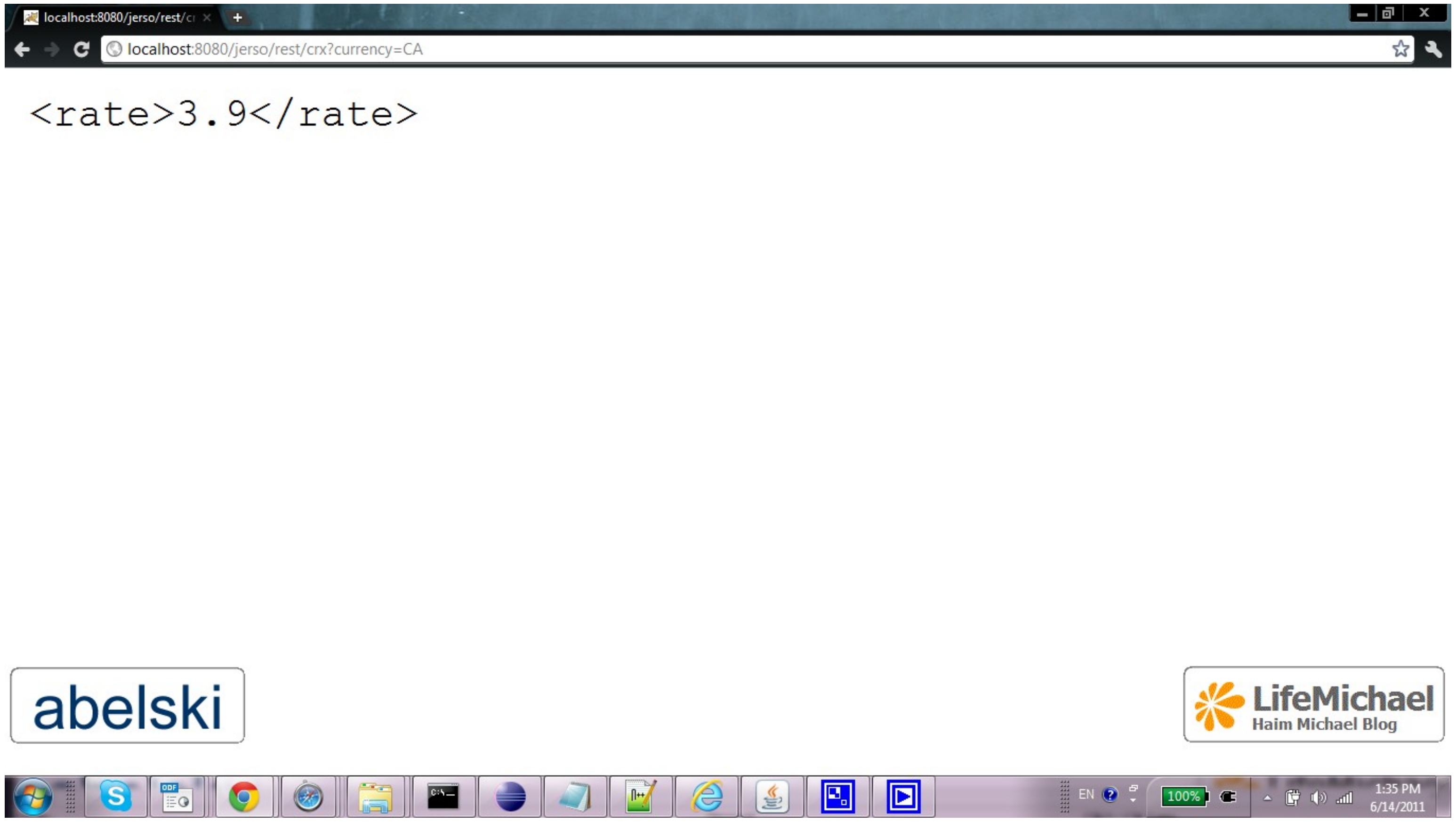
Query String Parameters

❖ We can easily extract the value of a specific query string parameter by using the `@QueryParam` annotation.

Query String Parameters

```
@Path("/crx")
public class CurrenciesExchangeRate
{
    @GET
    @Produces(MediaType.TEXT_PLAIN)
    public String getCurrencyExchangeRate(
        @DefaultValue("ILS") @QueryParam("currency") String currencyName)
    {
        double reply = 1;
        if(currencyName.equals("ILS")) reply = 1;
        if(currencyName.equals("USD")) reply = 3.4;
        if(currencyName.equals("EUR")) reply = 4.9;
        if(currencyName.equals("GBP")) reply = 6.5;
        if(currencyName.equals("CA")) reply = 3.9;
        return "<rate>"+reply+"</rate>";
    }
}
```

Query String Parameters



localhost:8080/jerso/rest/crx?currency=CA

```
<rate>3.9</rate>
```

abelski

LifeMichael
Haim Michael Blog

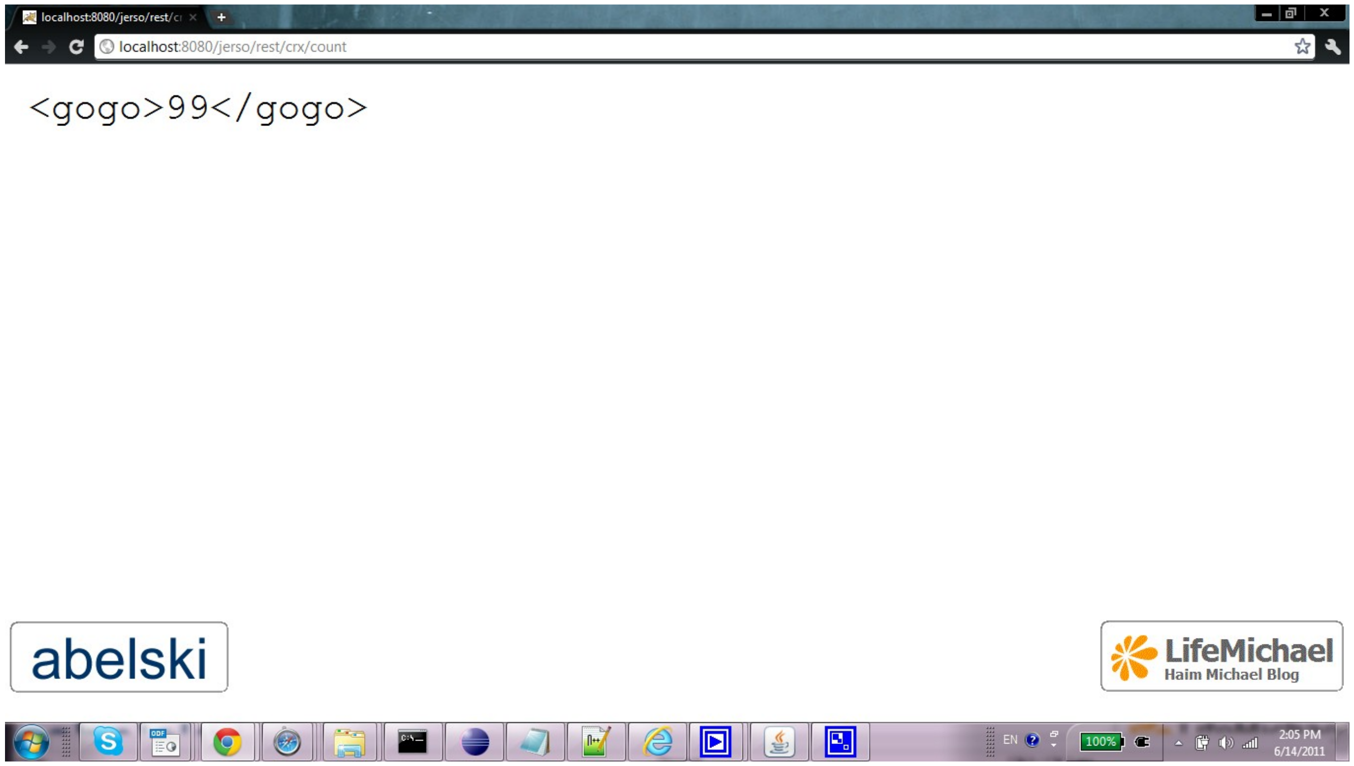
Path Segments

- We can easily associate different methods with the various URL possibilities. In other words, we can specify different methods to be invoked in according to the URL path that was used for using the web service.

Path Segments

```
@Path("/crx")
public class CurrenciesExchangeRate
{
    @GET
    @Produces(MediaType.TEXT_PLAIN)
    public String getCurrencyExchangeRate(
        @DefaultValue("ILS") @QueryParam("currency") String currencyName)
    {
        double reply = 1;
        if(currencyName.equals("ILS")) reply = 1;
        if(currencyName.equals("USD")) reply = 3.4;
        if(currencyName.equals("CA")) reply = 3.9;
        return "<rate>"+reply+"</rate>";
    }
    @GET
    @Path("/count")
    @Produces(MediaType.TEXT_PLAIN)
    public String getCount() {
        return "<gogo>99</gogo>";
    }
}
```

Path Segments

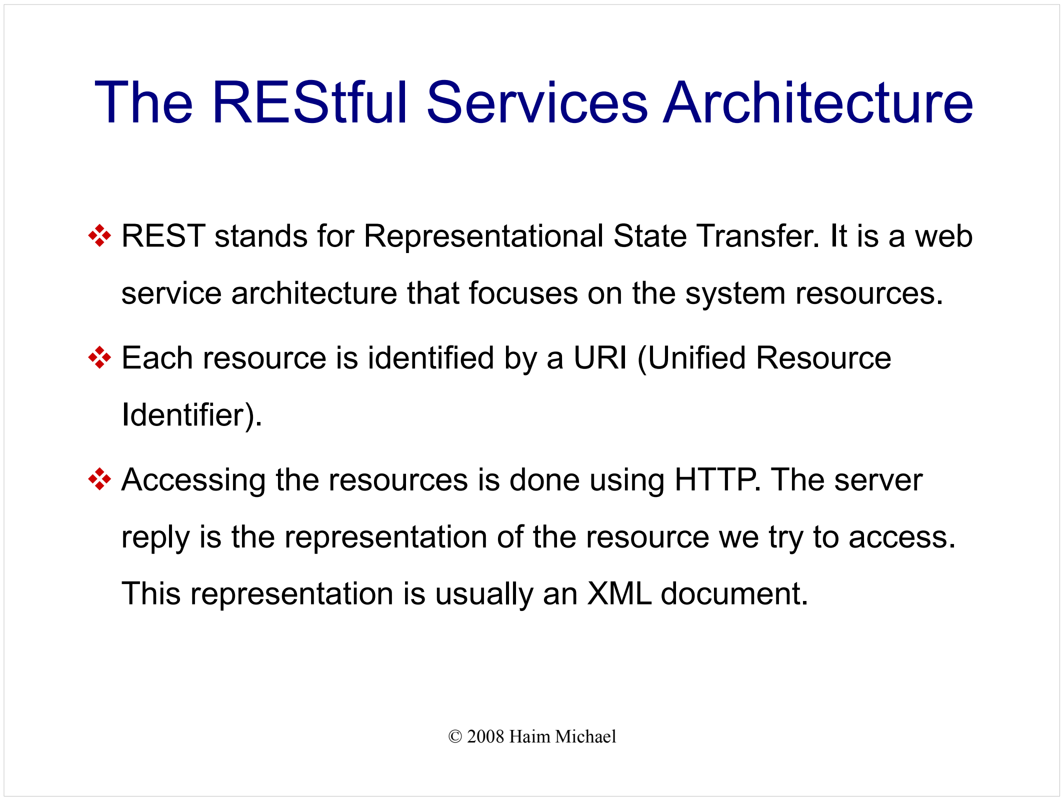


```
<gogo>99</gogo>
```

RESTful Web Services

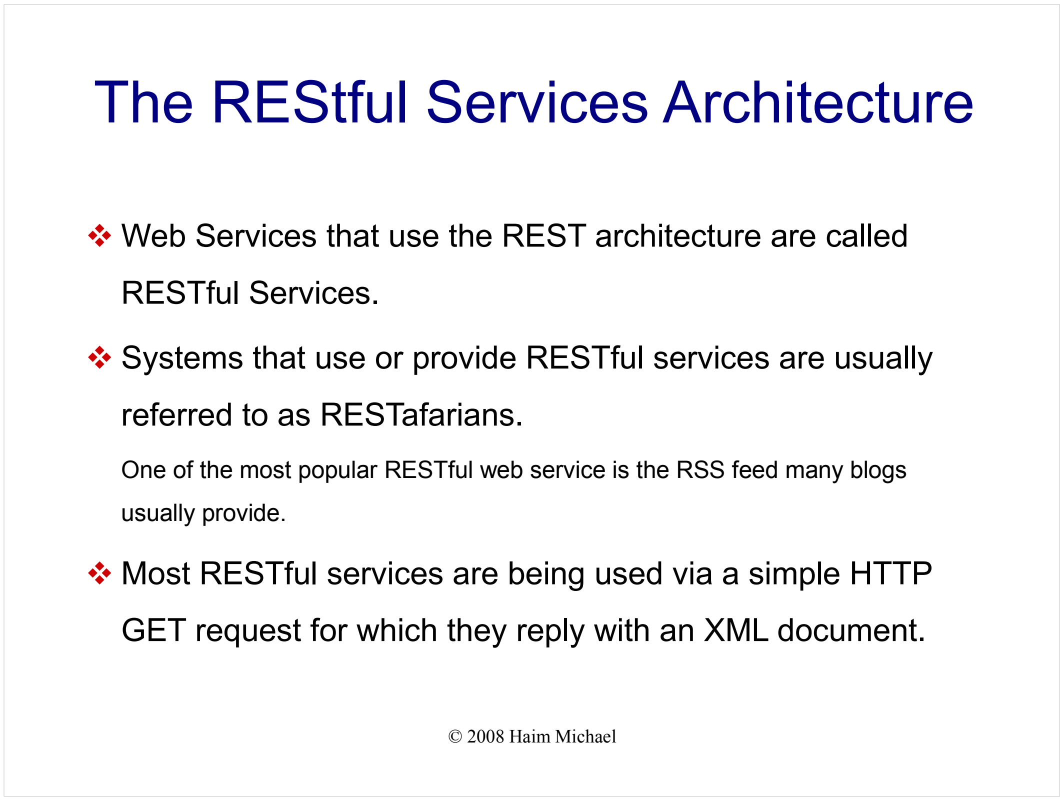
Introduction

- The `REStful` web services are highly popular in software architecture for distributed systems on the web.
- The Representational State Transfer (REST) web services set a new style fro SOA systems.
- The term Representational State Transfer was introduced in 2000 by Roy Fielding in his doctoral dissertation.



The RESTful Services Architecture

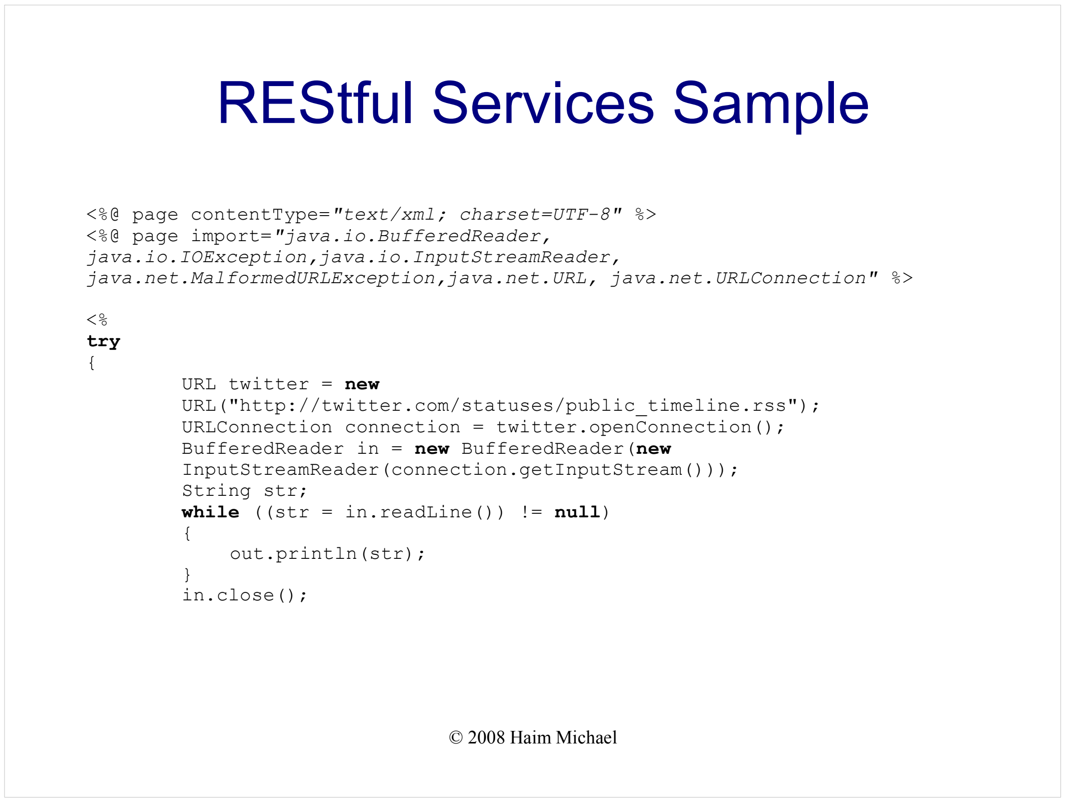
- ❖ REST stands for Representational State Transfer. It is a web service architecture that focuses on the system resources.
- ❖ Each resource is identified by a URI (Unified Resource Identifier).
- ❖ Accessing the resources is done using HTTP. The server reply is the representation of the resource we try to access. This representation is usually an XML document.



The REStful Services Architecture

- Web Services that use the REST architecture are called RESTful Services.
- Systems that use or provide RESTful services are usually referred to as RESTafarians.

  One of the most popular RESTful web service is the RSS feed many blogs usually provide.
- Most RESTful services are being used via a simple HTTP GET request for which they reply with an XML document.



RESTful Services Sample

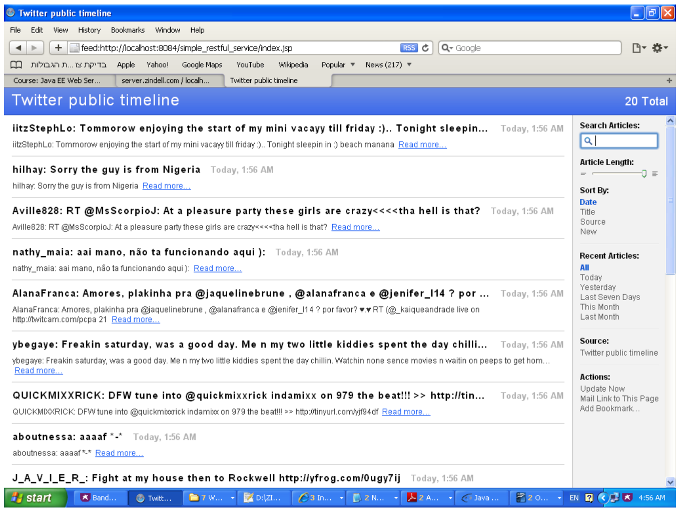
```
<%@ page contentType="text/xml; charset=UTF-8" %>
<%@ page import="java.io.BufferedReader,
java.io.IOException,java.io.InputStreamReader,
java.net.MalformedURLException,java.net.URL, java.net.URLConnection" %>

<%
try
{
        URL twitter = new
        URL("http://twitter.com/statuses/public_timeline.rss");
        URLConnection connection = twitter.openConnection();
        BufferedReader in = new BufferedReader(new
        InputStreamReader(connection.getInputStream()));
        String str;
        while ((str = in.readLine()) != null)
        {
            out.println(str);
        }
        in.close();
```

REStful Services Sample

```
catch (MalformedURLException e)
{
    e.printStackTrace();
}
catch (IOException e)
{
    e.printStackTrace();
}
%>
```

REStful Services Sample


Twitter public timeline
File Edit View History Bookmarks Window Help
feed:http://localhost:8084/simple_restful_service/index.jsp
RSS
Google
בדיקת צו... ת הגבולות Apple Yahoo! Google Maps YouTube Wikipedia Popular News (217)
Course: Java EE Web Ser...
server.zindell.com / localh...
Twitter public timeline
start Band... Twitt... 7 W... D:\ZI... 3 In... 2 N... 2 A... Java ... 2 O... EN 4:56 AM

Twitter public timeline 20 Total

iitzStephLo: Tommorow enjoying the start of my mini vacayy till friday :).. Tonight sleepin... Today, 1:56 AM
iitzStephLo: Tommorow enjoying the start of my mini vacayy till friday :).. Tonight sleepin in :) beach manana Read more...

hilhay: Sorry the guy is from Nigeria Today, 1:56 AM
hilhay: Sorry the guy is from Nigeria Read more...

Aville828: RT @MsScorpioJ: At a pleasure party these girls are crazy<<<<tha hell is that? Today, 1:56 AM
Aville828: RT @MsScorpioJ: At a pleasure party these girls are crazy<<<<tha hell is that? Read more...

nathy_maia: aai mano, não ta funcionando aqui): Today, 1:56 AM
nathy_maia: aai mano, não ta funcionando aqui): Read more...

AlanaFranca: Amores, plakinha pra @jaquelinebrune , @alanafranca e @jenifer_l14 ? por ... Today, 1:56 AM
AlanaFranca: Amores, plakinha pra @jaquelinebrune , @alanafranca e @jenifer_l14 ? por favor? ♥.♥ RT (@_kaiqueandrade live on http://twitcam.com/pcpa 21 Read more...

ybegaye: Freakin saturday, was a good day. Me n my two little kiddies spent the day chilli... Today, 1:56 AM
ybegaye: Freakin saturday, was a good day. Me n my two little kiddies spent the day chillin. Watchin none sence movies n waitin on peeps to get hom... Read more...

QUICKMIXXRICK: DFW tune into @quickmixxrick indamixx on 979 the beat!!! >> http://tin... Today, 1:56 AM
QUICKMIXXRICK: DFW tune into @quickmixxrick indamixx on 979 the beat!!! >> http://tinyurl.com/yjf94df Read more...

aboutnessa: aaaaf *-* Today, 1:56 AM
aboutnessa: aaaaf *-* Read more...

J_A_V_I_E_R_: Fight at my house then to Rockwell http://yfrog.com/0ugy7ij Today, 1:56 AM

Search Articles:

Article Length:

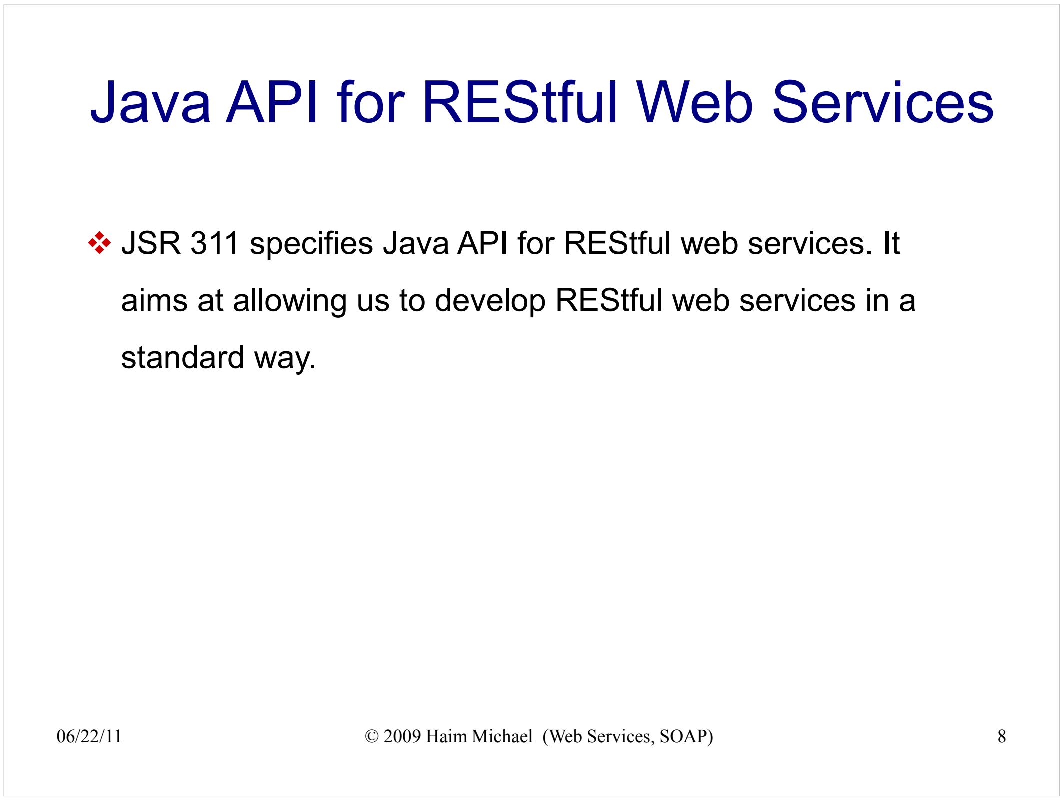
Sort By:
Date
Title
Source
New

Recent Articles:
All
Today
Yesterday
Last Seven Days
This Month
Last Month

Source:
Twitter public timeline

Actions:
Update Now
Mail Link to This Page
Add Bookmark...

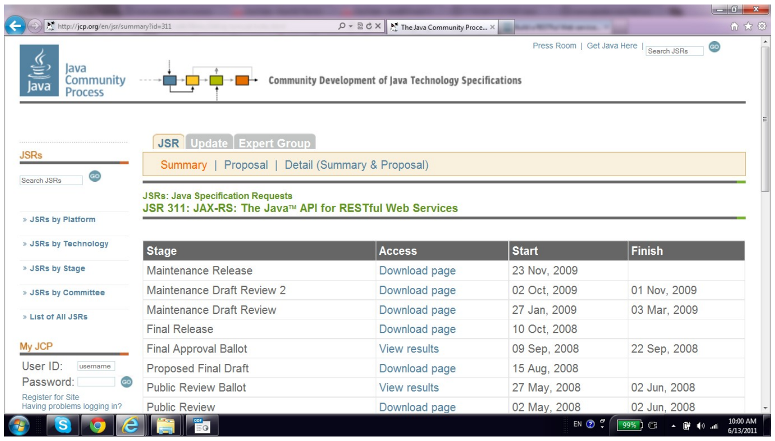
© 2008 Haim Michael



Java API for RESTful Web Services

- JSR 311 specifies Java API for RESTful web services. It aims at allowing us to develop RESTful web services in a standard way.

Java API for REStful Web Services



http://jcp.org/en/jsr/summary?id=311

Press Room | Get Java Here | Search JSRs

Java Community Process

Community Development of Java Technology Specifications

JSR | Update | Expert Group

Summary | Proposal | Detail (Summary & Proposal)

JSRs: Java Specification Requests

JSR 311: JAX-RS: The Java™ API for RESTful Web Services

JSRs

» JSRs by Platform

» JSRs by Technology

» JSRs by Stage

» JSRs by Committee

» List of All JSRs

My JCP

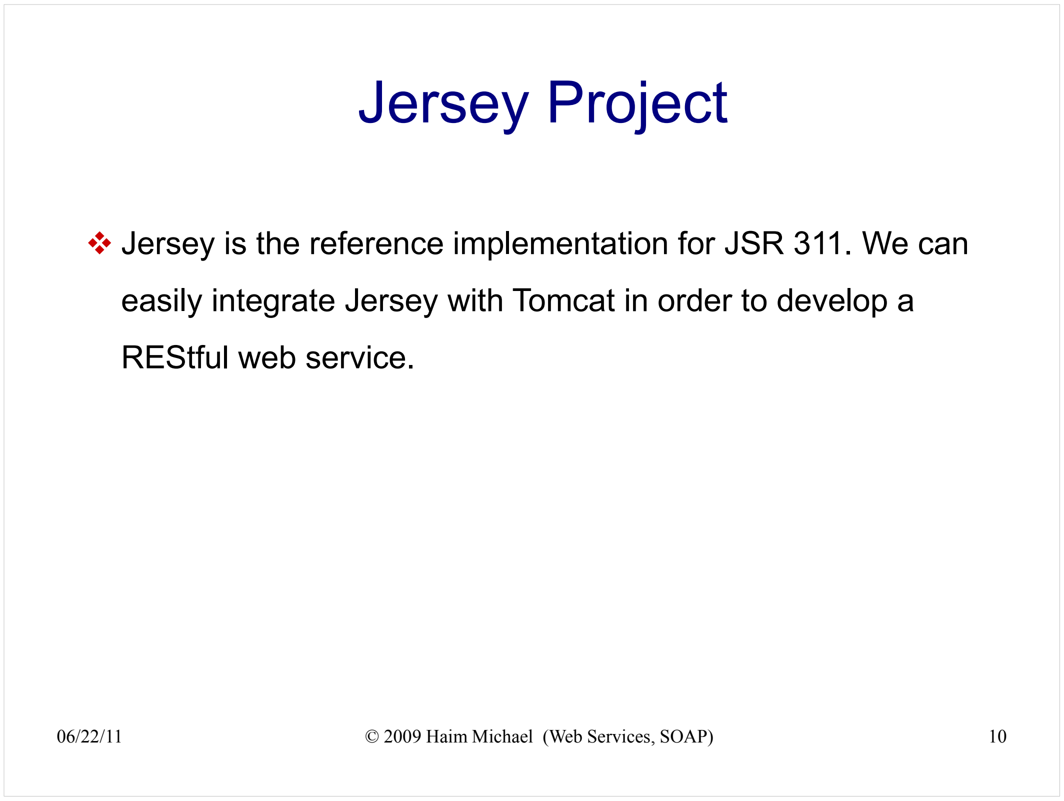
User ID:

Password:

Register for Site

Having problems logging in?

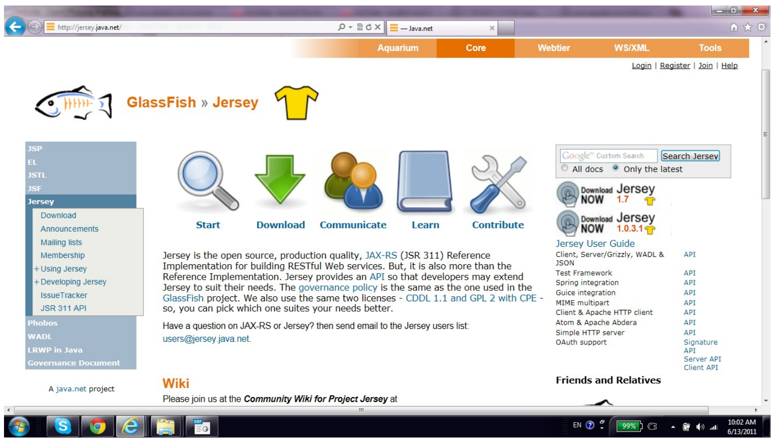
Stage	Access	Start	Finish
Maintenance Release	Download page	23 Nov, 2009	
Maintenance Draft Review 2	Download page	02 Oct, 2009	01 Nov, 2009
Maintenance Draft Review	Download page	27 Jan, 2009	03 Mar, 2009
Final Release	Download page	10 Oct, 2008	
Final Approval Ballot	View results	09 Sep, 2008	22 Sep, 2008
Proposed Final Draft	Download page	15 Aug, 2008	
Public Review Ballot	View results	27 May, 2008	02 Jun, 2008
Public Review	Download page	02 May, 2008	02 Jun, 2008



Jersey Project

❖ Jersey is the reference implementation for JSR 311. We can easily integrate Jersey with Tomcat in order to develop a REStful web service.

Jersey Project



Jersey is the open source, production quality, JAX-RS (JSR 311) Reference Implementation for building RESTful Web services. But, it is also more than the Reference Implementation. Jersey provides an API so that developers may extend Jersey to suit their needs. The governance policy is the same as the one used in the GlassFish project. We also use the same two licenses - CDDL 1.1 and GPL 2 with CPE - so, you can pick which one suites your needs better.

Have a question on JAX-RS or Jersey? then send email to the Jersey users list: users@jersey.java.net.

Wiki

Please join us at the ***Community Wiki for Project Jersey*** at

Jersey User Guide

Client, Server/Grizzly, WADL & JSON	API
Test Framework	API
Spring integration	API
Guice integration	API
MIME multipart	API
Client & Apache HTTP client	API
Atom & Apache Abdera	API
Simple HTTP server	API
OAuth support	Signature API Server API Client API

Friends and Relatives

Download Jersey

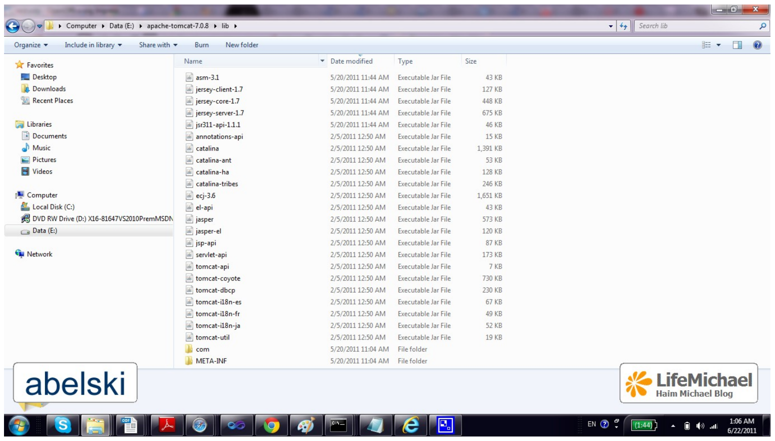
- You can easily download Jersey latest version browsing at
  http://download.java.net/maven/2/com/sun/jersey/jersey-archive/1.7/jersey-archive-1.7.zip
- In order to deploy a simple Jersey based RESTful web service we will need the following files:

```
jersey-core.jar
jersey-server.jar
jsr311-api.jar
asm.jar
```

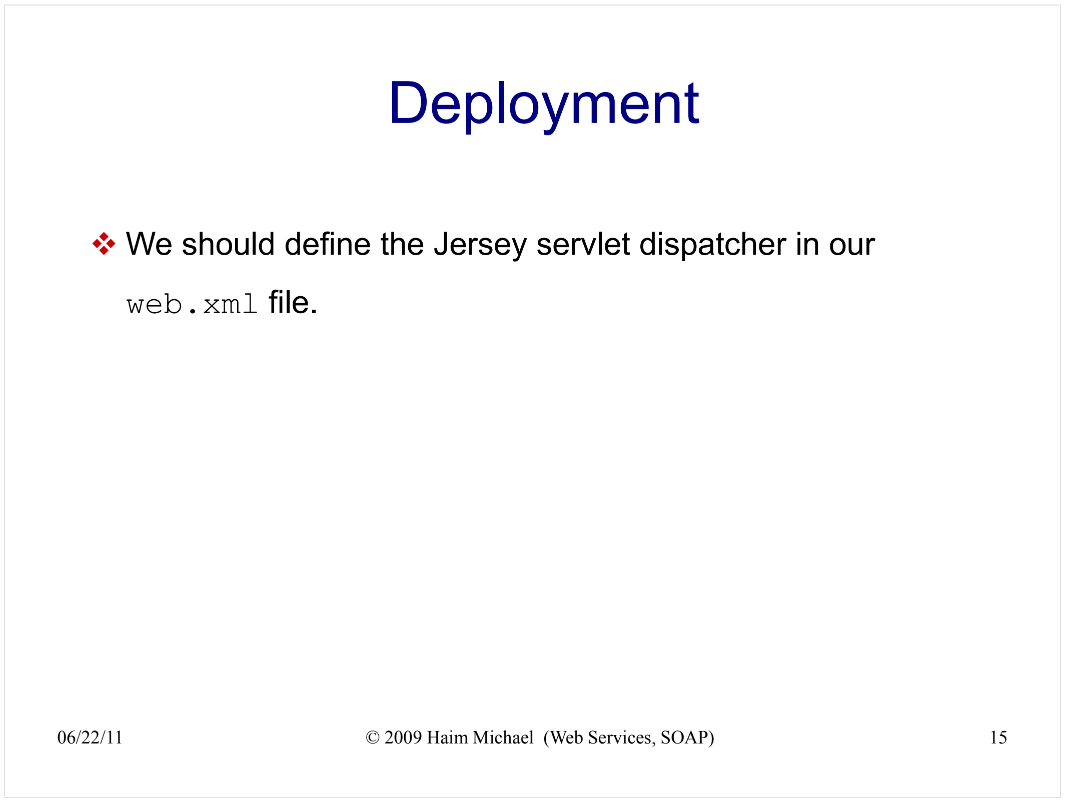
Install Jersey

❖ We should first copy these jar files into the `lib` folder of our Tomcat installation.

Install Jersey

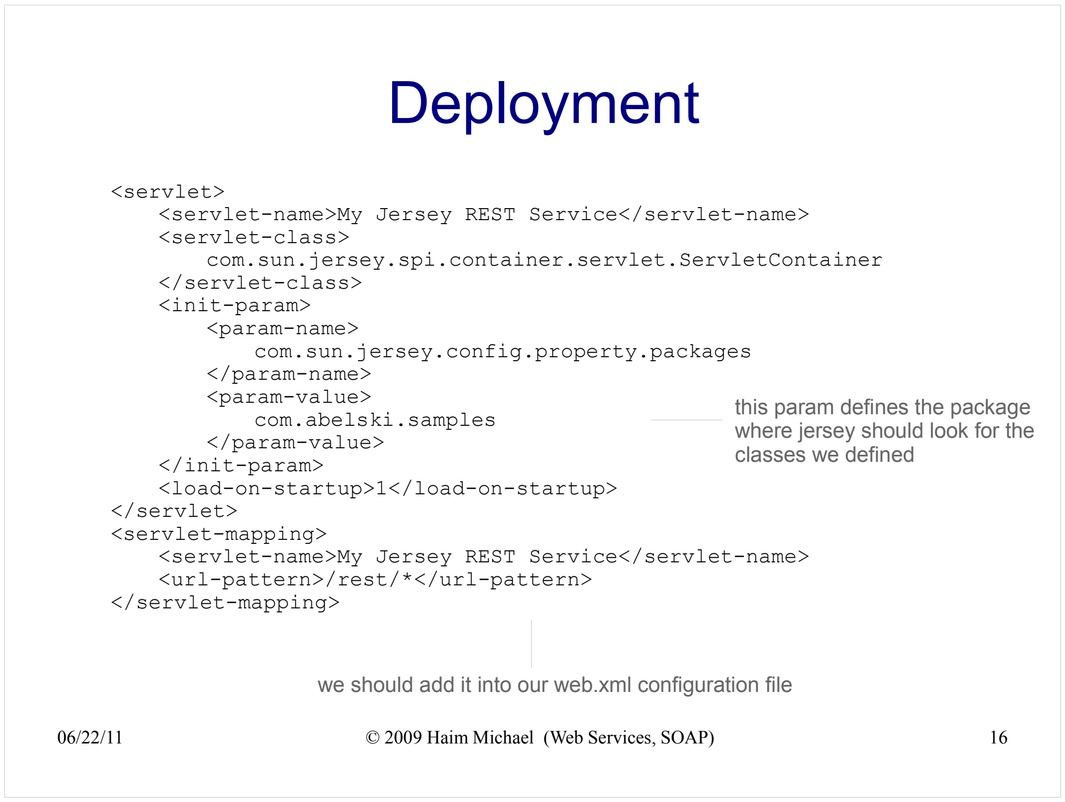


Name	Date modified	Type	Size
asm-3.1	5/20/2011 11:44 AM	Executable Jar File	43 KB
jersey-client-1.7	5/20/2011 11:44 AM	Executable Jar File	127 KB
jersey-core-1.7	5/20/2011 11:44 AM	Executable Jar File	448 KB
jersey-server-1.7	5/20/2011 11:44 AM	Executable Jar File	675 KB
jsr311-api-1.1.1	5/20/2011 11:44 AM	Executable Jar File	46 KB
annotations-api	2/5/2011 12:50 AM	Executable Jar File	15 KB
catalina	2/5/2011 12:50 AM	Executable Jar File	1,391 KB
catalina-ant	2/5/2011 12:50 AM	Executable Jar File	53 KB
catalina-ha	2/5/2011 12:50 AM	Executable Jar File	128 KB
catalina-tribes	2/5/2011 12:50 AM	Executable Jar File	246 KB
ecj-3.6	2/5/2011 12:50 AM	Executable Jar File	1,651 KB
el-api	2/5/2011 12:50 AM	Executable Jar File	43 KB
jasper	2/5/2011 12:50 AM	Executable Jar File	573 KB
jasper-el	2/5/2011 12:50 AM	Executable Jar File	120 KB
jsp-api	2/5/2011 12:50 AM	Executable Jar File	87 KB
servlet-api	2/5/2011 12:50 AM	Executable Jar File	173 KB
tomcat-api	2/5/2011 12:50 AM	Executable Jar File	7 KB
tomcat-coyote	2/5/2011 12:50 AM	Executable Jar File	730 KB
tomcat-dbcp	2/5/2011 12:50 AM	Executable Jar File	230 KB
tomcat-i18n-es	2/5/2011 12:50 AM	Executable Jar File	67 KB
tomcat-i18n-fr	2/5/2011 12:50 AM	Executable Jar File	49 KB
tomcat-i18n-ja	2/5/2011 12:50 AM	Executable Jar File	52 KB
tomcat-util	2/5/2011 12:50 AM	Executable Jar File	19 KB
com	5/20/2011 11:04 AM	File folder	
META-INF	5/20/2011 11:04 AM	File folder	



Deployment

❖ We should define the Jersey servlet dispatcher in our `web.xml` file.



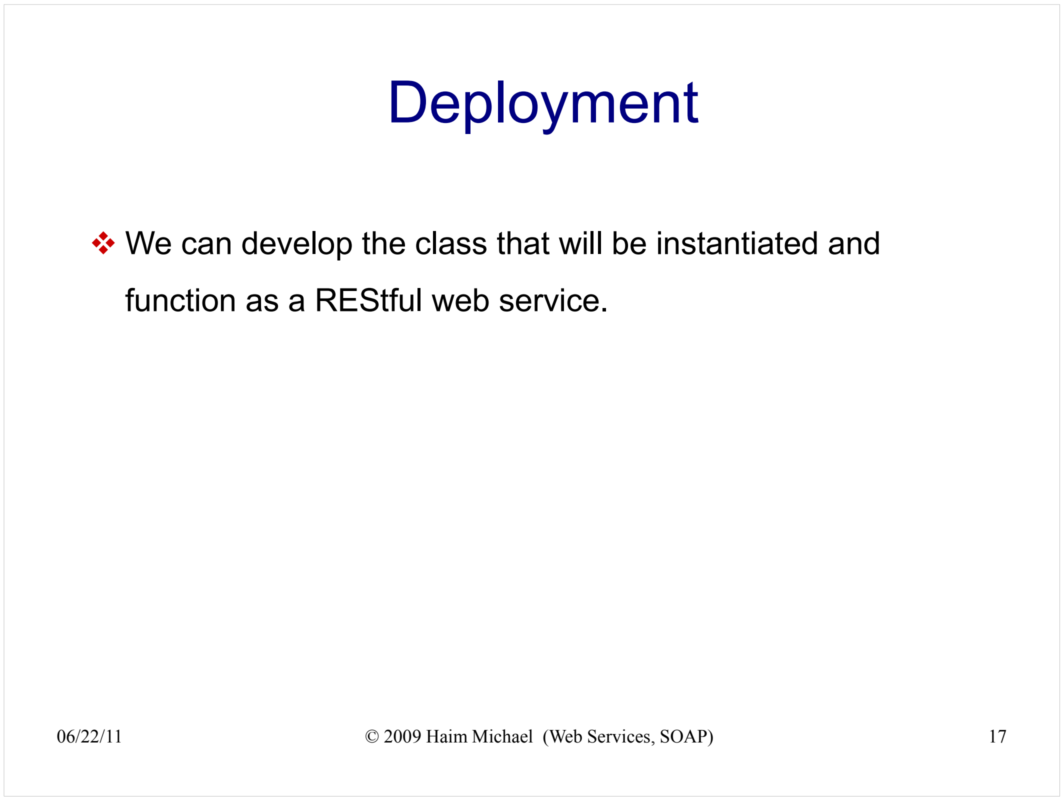
Deployment

```
<servlet>
    <servlet-name>My Jersey REST Service</servlet-name>
    <servlet-class>
        com.sun.jersey.spi.container.servlet.ServletContainer
    </servlet-class>
    <init-param>
        <param-name>
            com.sun.jersey.config.property.packages
        </param-name>
        <param-value>
            com.abelski.samples
        </param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>My Jersey REST Service</servlet-name>
    <url-pattern>/rest/*</url-pattern>
</servlet-mapping>
```

this param defines the package where jersey should look for the classes we defined

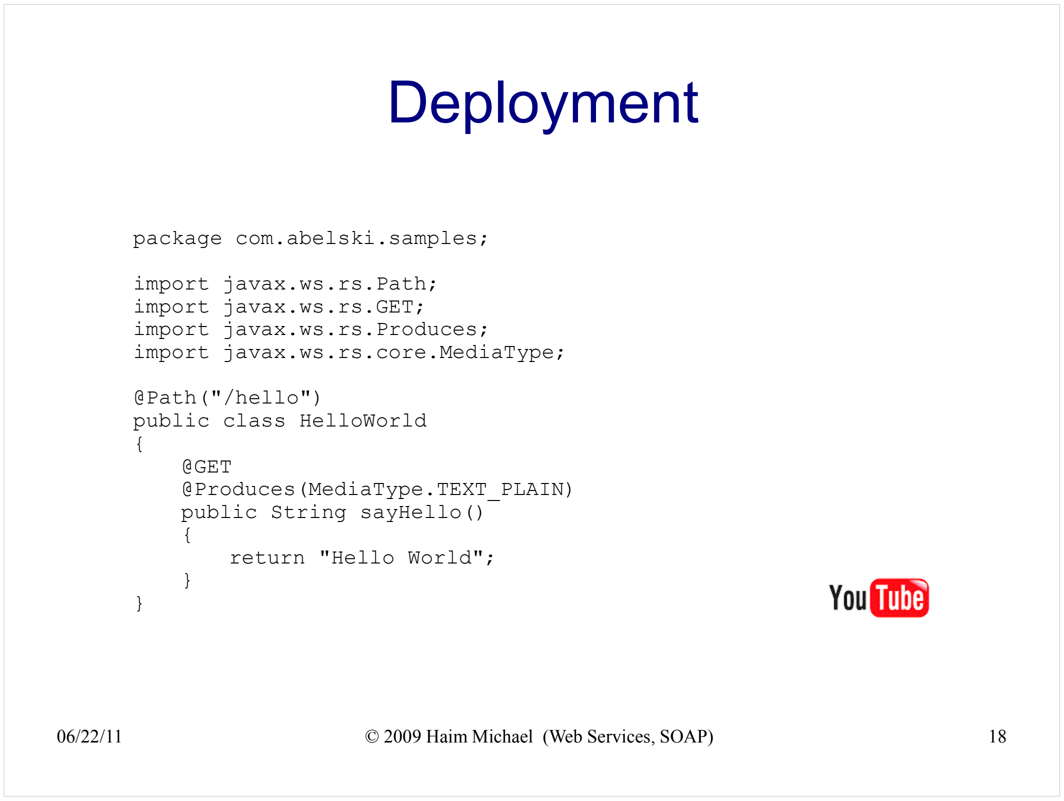
we should add it into our web.xml configuration file

06/22/11 © 2009 Haim Michael (Web Services, SOAP) 16



Deployment

❖ We can develop the class that will be instantiated and function as a RESTful web service.



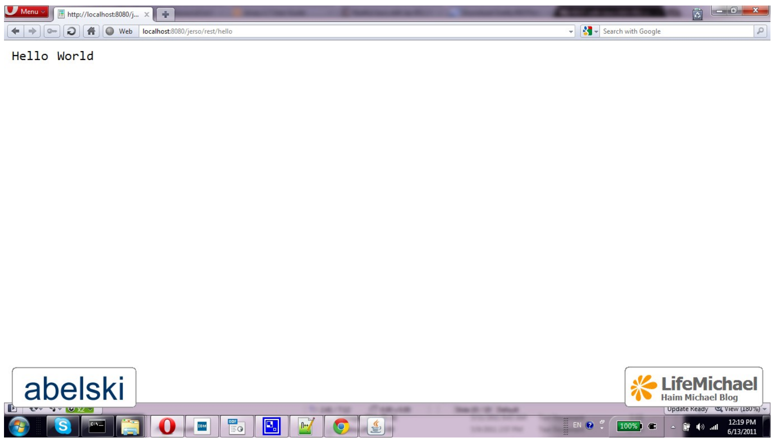
Deployment

```
package com.abelski.samples;

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

@Path("/hello")
public class HelloWorld
{
    @GET
    @Produces(MediaType.TEXT_PLAIN)
    public String sayHello()
    {
        return "Hello World";
    }
}
```

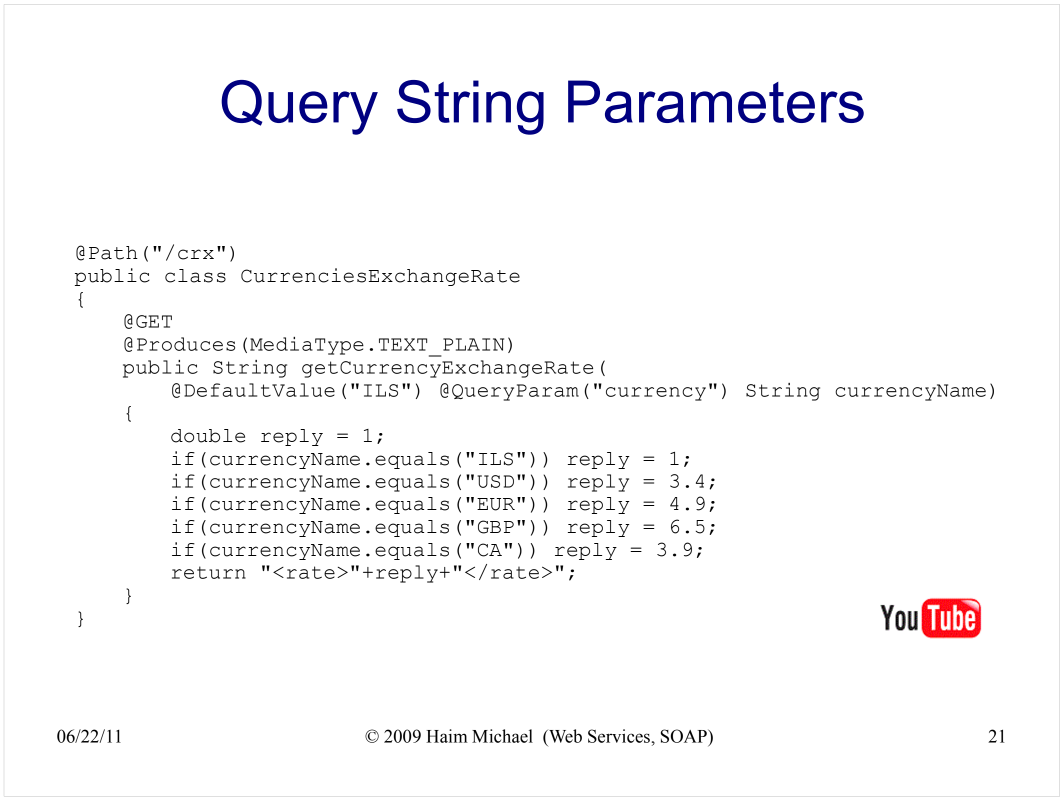
Deployment



```
Hello World
```

Query String Parameters

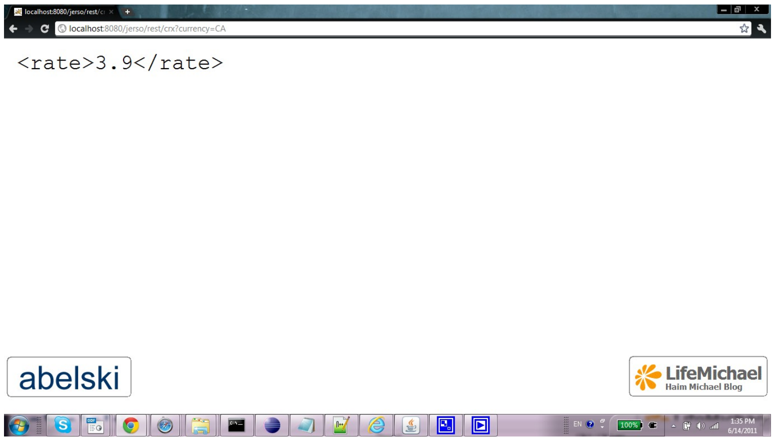
- We can easily extract the value of a specific query string parameter by using the `@QueryParam` annotation.



Query String Parameters

```
@Path("/crx")
public class CurrenciesExchangeRate
{
    @GET
    @Produces(MediaType.TEXT_PLAIN)
    public String getCurrencyExchangeRate(
        @DefaultValue("ILS") @QueryParam("currency") String currencyName)
    {
        double reply = 1;
        if(currencyName.equals("ILS")) reply = 1;
        if(currencyName.equals("USD")) reply = 3.4;
        if(currencyName.equals("EUR")) reply = 4.9;
        if(currencyName.equals("GBP")) reply = 6.5;
        if(currencyName.equals("CA")) reply = 3.9;
        return "<rate>"+reply+"</rate>";
    }
}
```

Query String Parameters



```
<rate>3.9</rate>
```

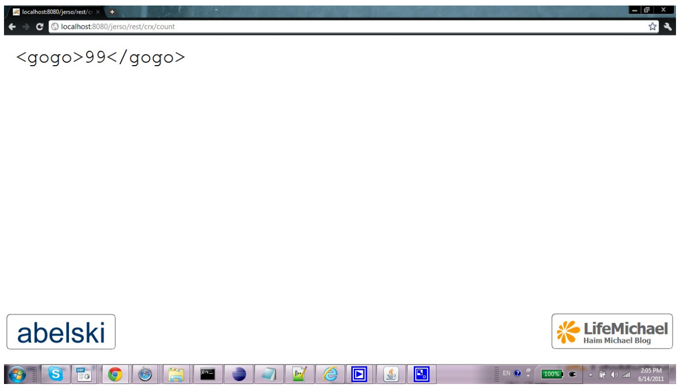
Path Segments

❖ We can easily associate different methods with the various URL possibilities. In other words, we can specify different methods to be invoked in according to the URL path that was used for using the web service.

Path Segments

```
@Path("/crx")
public class CurrenciesExchangeRate
{
    @GET
    @Produces(MediaType.TEXT_PLAIN)
    public String getCurrencyExchangeRate(
        @DefaultValue("ILS") @QueryParam("currency") String currencyName)
    {
        double reply = 1;
        if(currencyName.equals("ILS")) reply = 1;
        if(currencyName.equals("USD")) reply = 3.4;
        if(currencyName.equals("CA")) reply = 3.9;
        return "<rate>"+reply+"</rate>";
    }
    @GET
    @Path("/count")
    @Produces(MediaType.TEXT_PLAIN)
    public String getCount() {
        return "<gogo>99</gogo>";
    }
}
```

Path Segments


localhost:8080/jerso/rest/cr
localhost:8080/jerso/rest/crx/count

```
<gogo>99</gogo>
```

abelski

LifeMichael
Haim Michael Blog

06/22/11 © 2009 Haim Michael (Web Services, SOAP) 25