n the first [RESTEasy tutorial](http://www.mastertheboss.com/jboss-frameworks/resteasy/resteasy-tutorial) [RESTEasy tutorial](http://www.mastertheboss.com/web-interfaces/273-resteasy-tutorial-.html) we have learnt the basics about REST Web services and we have tested a simple RESTful Web service. In this tutorial we will show how to inject web application elements (form parameters, query parameters and more) into a RESTful Web service.

You can use the following annotations to bind HTTP requests to a RESTful web service:

@FormParam

@PathParam

@QueryParam

@HeaderParam

@CookieParam

@MatrixParam

Let's explore all the possible interactions.

**@FormParam**

The annotation **@FormParam** can be used to inject the parameters of a Web form into a RESTful Web service.

Here's an example:



Here we are submitting a POST request containing two parameters email and password which are translated into the parameters "e" and "p" of the login method.

Here's the full example:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | <form method="POST" action="login">      Email Address: <input type="text" name="email">  <br>  Password: <input type="text" name="password">  <input type="submit">      </form> |
| 1  2  3  4  5  6  7  8  9  10  11  12 | @Path("/")  public class LoginService  {        @Path("login")    @POST    public String login(@FormParam("email") String e, @FormParam("password") String p) {     return "Logged with " + e + " " + p;    }    } | |

As an alternative, you can bind the parameters email and password at class level, which can be useful if you need to re-use the same parameters across different methods of the service.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | public class User {      @FormParam("email")    private String email;    @FormParam("password")    private String password;    } |

You would need to modify the REST method accordingly:

|  |  |
| --- | --- |
| 1  2  3  4  5 | @POST   @Path("login")   public String login(@Form User form) {   return "Logged with " + form.email + " " + form.password;   } |

**@PathParam**

The @PathParam annotation binds the value of a path segment to a resource method parameter. For example, the following method would intercept an HTTP GET like **http://server:port/login/12345** and convert the PathParam "12345" into the String "id"

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | @Path("/")  public class LoginService  {        @GET    @Path("login/{zip}")    public String login(@PathParam("zip") String id) {     return "Id is " +id;    }    } |

As for @FormParam, you can embed the @PathParam declaration at class level, if you prefer.

**@QueryParam**

The @QueryParam annotation binds the value of a path segment to a resource method parameter. For example, the following method would intercept an HTTP GET like [http://server:port/login?zip=12345](http://serverport/) and inject the query parameter "zip" into the method parameter "zip"

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | @Path("/")  public class LoginService  {     @GET   @Path("login/{zip}")    public String login(@QueryParam("zip") String zip) {     return "Id is " +id;    }    } |

QueryParam can be convenientely used with the DefaultValue annotation so that you can avoid a null pointer exception if no query parameter is passed.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | @GET   @Path("login/{zip}")    public String login(@DefaultValue("11111") @QueryParam("zip") String zip) {   return "Id is " +id;   } |

As for @FormParam, you can embed the @PathParam declaration at class level, if you prefer.

**@HeaderParam**

The @HeaderParam annotation extracts information from the HTTP header and binds it to a method parameter. Example:

|  |  |
| --- | --- |
| 1  2  3  4 | @GET  public String callService(@HeaderParam("User-Agent") String whichBrowser) {  ...  } |

**@CookieParam**

The @CookieParam annotation reads an information stored as a cookie and binds it to a method parameter. Example:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | @GET    public String callService(@CookieParam("sessionid") String sessionid) {    ...   } |

**@MatrixParam**

The @MatrixParam annotation can be used to bind an expression containing several property=value to a method parameter. For example, supposing you were to invoke an URL like[http://server:port/login](http://serverport/);name=francesco;surname=marchioni

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | @GET    public String callService(@MatrixParam("name") String name,                                    @MatrixParam("surname") String surname) {    ...   } |

|  |  |  |
| --- | --- | --- |
| How RESTful Web Services Extract Input Parameters  |  |  | | --- | --- | | [Web Services](http://www.java4s.com/web-services/) » On Jul 6, 2014 By [Sivateja](https://plus.google.com/118054670710951892925?rel=author) | [Tweet](http://twitter.com/share) |   In this article i will show you how a RESTful web service will  extract input parameters from the client request.  We have different ways of sending input values to the rest services, and RESTful web service extract those details based upon the client URL pattern. In JAX-RS we can use the following annotations to extract the input values sent by the client.   * @PathParam * @QueryParam * @MatrixParam * @FormParam   @PathParam,@QueryParam,@MatrixParam are parameter annotations which allows us to map variable URI path fragments into your method call. Confused ? :-) In simple words, these three annotations will come into picture in case if we are passing the input values to the restful service through the URL. After that Rest service will extract those values by using these annotations. Regarding @FormParam, restful web service will use this annotation to retrieve the values sent by the client through some HTML/JSP form. @PathParam URL Syntax http://localhost:7001/<Rest Service Name>/rest/customers/100/Java4s  Did you observe the two parameters appear in the end of the above URL [100 & Java4s], which are separated by forward slash(/) are called as path parameters, as of now just remember the syntax, going forward i will give you an example on each annotation. @QueryParam URL Syntax http://localhost:7001/…/rest/customers?custNo=100&custName=Java4s  If the client sends an input in the form of query string in the URL, then those parameters are called as Query Parameters.  If you observe the above syntax, client passing 2 parameters 100 and Java4s  started after question mark (?) symbol and each parameter is separated by & symbol,  those parameters are called as query parameters. @MatrixParam URL Syntax http://localhost:7001/…/rest/customers;custNo=100;custName=Java4s  Matrix parameters are  another way defining the parameters to be added to URL.  If you observe the above syntax, client is passing two parameters each are separated by comma, these parameters are called as matrix parameters.  Remember these parameters may appear any where in the path. @FormParam URL Syntax Finally form parameters,  if we have a HTML form having two input fields and submit button. Lets client enter those details and submit to the RESTful web service. Then the rest service will extract those details by using this @FormParam annotation. For now just remember these consents, going forward i will give you an example on each annotation. |