Programación Distribuida

JAX-RS

JAIME SALVADOR

AGENDA

- **Introducción**
- Content Type
- Anotaciones
- Clientes

Introducción

Java API for RESTful Web Services (JAX-RS)

- Es un API del lenguaje Java el cual proporciona soporte para la creación de servicios web según la arquitectura REST.
- Es una colección de interfaces y anotaciones

Introducción

Java API for RESTful Web Services (JAX-RS)

- POJO-based
- Basado en HTTP
- Independiente del contenedor
- Parte de JavaEE
- Permite la construcción y el consumo de servicios REST
- Facilita el marshaling/unmarshaling de los datos
- Client API (basado en anotaciones)

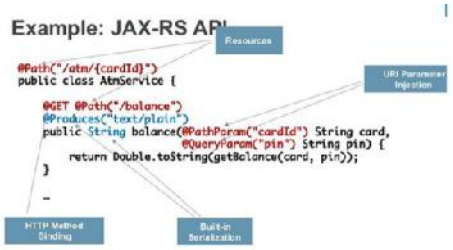
Introducción

Java API for RESTful Web Services (JAX-RS)

- Filtros e Interceptores: request-response invocation chain
- Valicación: @NotNull
- Procesamiento asincrónico (@Suspended)
- @Context: acceso al ambiente de ejecución

Introducción

Java API for RESTful Web Services (JAX-RS)



AGENDA

- Introducción
- **Content Type**
- Anotaciones
- Clientes

Content Type

Formatos (media type) soportados por JAX-RS:

- `APPLICATION_JSON: application/json`
- `APPLICATION_XML: application/xml`
- `TEXT_HTML: text/html`
- `TEXT_PLAIN: text/plain`
- `TEXT_XML: text/xml`

Content Type

Formatos (media type) soportados por JAX-RS:

```
@Produces("application/json")
@Consumes("text/xml)
```

AGENDA

- Introducción
- Content Type
- **Anotaciones**
- Clientes

Anotaciones

JAX-RS Injection --> DI?

Extraer información de una petición HTTP e "inyectar" los valores en los parámetros de un método

- URI matching pattern: `@Path`
  - Es posible utilizarlo en una clase o en un método
  - La clase anotada con `@Path("/")` se denomina ***JAX-RS root resource***
- La porción del URL es relativo al contexto de la aplicación JAX-RS:

  `@Path`

Anotaciones

- `@PathParam` permite extraer valores desde un URI (template)
- `@QueryParam` permite extraer valores desde la cadena de consulta asociada a un URI
- `@FormParam` permite extraer valores enviados desde un formulario (POST)
- `@HeaderParam` permite extraer valores desde el encabezado HTTP
- `@CookieParam` permite extraer valores desde los cookies enviados desde el cliente
- `@MatrixParam` permite extraer valores desde un URI
- `@Context` permite inyectar objetos asociados con el runtime

Anotaciones

@PathParam

```
@GET
@Path(value="/hola1/{nombre}")
public String hola1( @PathParam("nombre") String nombre );
```

http://127.0.0.1:8080/rest-01/services/hola1/mundo

Anotaciones

@QueryParam

```
@GET
@Path(value="/hola2")
public String hola2( @QueryParam("nombre") String nombre );
```

http://127.0.0.1:8080/rest-01/services/hola2?nombre=mundo

Anotaciones

@QueryParam

```
public String hola2( HttpServletRequest request ) {
    String nombre = request.getParameter( "nombre" );
    return "Hola " + nombre;
}
```

```
@GET
@Path(value="/hola2")
public String hola2( @Context HttpServletRequest request );
```

http://127.0.0.1:8080/rest-01/services/hola2?nombre=mundo

Anotaciones

@HeaderParam


```
@GET
@Path(value="/hola3")
public String hola3( @HeaderParam("Accept") String accept );
```

http://127.0.0.1:8080/rest-01/services/hola3

Anotaciones

@HeaderParam

```
@GET
@Path(value="/hola4")
public String hola4(
    @MatrizParam("nombre") String nombre,
    @MatrizParam("apellido") String apellido );
```

http://127.0.0.1:8080/rest-01/services/hola4;nombre=a1;apellido=ap1

AGENDA

- Introducción
- Content Type
- Anotaciones
- **Clientes**

Cliente

```
Client client = ClientBuilder.newClient();
WebTarget target =
    client.target( "http://127.0.0.1:8080/rest-01/services" );

target = target.path("/hola1/mundo" );

Invocation.Builder builder = target.request();

String respuesta = builder.get( String.class );

System.out.println( "RESPUESTA: " + respuesta );
```

Cliente

```
Client client = ClientBuilder.newClient();

String response
   =  client.target( "http://127.0.0.1:8080/rest-01/services" )
      .path("/hola1/{nombre}")
      .resolveTemplate("nombre","mundo")
      .request( MediaType.APPLICATION_JSON )
      .get( String.class );

System.out.println( "RESPUESTA: " + response );
```

Cliente

```
RestTemplate restTemplate = new RestTemplate( );

String respuesta =
   restTemplate.getForObject(
   "http://127.0.0.1:8080/rest-01/services/hola1/{nombre}",
   String.class, "mundo" );

System.out.println( respuesta );
```

Implementaciones?

Implementaciones

JAX-RS + Spring Framework

PREGUNTAS?