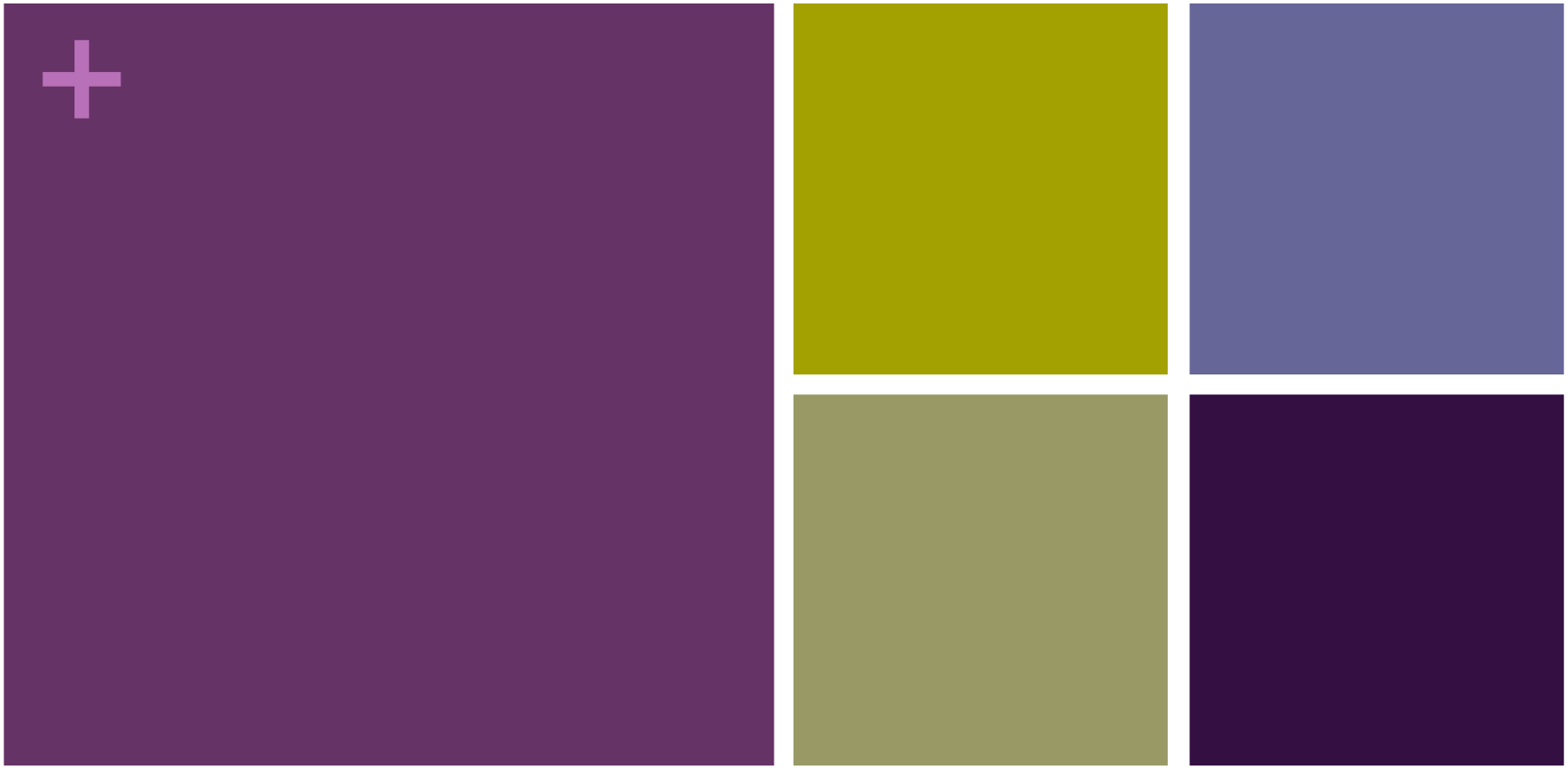


Sistemas Distribuidos
REST FAQ

Rodrigo Santamaría

No hay salida en el navegador

REST

- El código no funciona o no aparece nada al visitar la URL del servicio (o Estado HTTP 404).
- En `web.xml`, `display-name` debe contener el nombre del proyecto, y el valor `param-value` del `param-name` `'com.sun.jersey.config.property.packages'` debe contener el nombre del paquete donde están las clases.
- Observa si, en la pestaña de Consola de Eclipse, Tomcat ha reportado algún error
  - Si ha arrancado correctamente la última línea será "`Server startup in xxx ms`" o similar

No hay salida en el navegador

Otros posibles errores

- Si tienes un error 404, recuerda que la URL incluye secciones que dependen del web.xml y de las anotaciones @Path (ver diap. 19 del seminario de REST)
- La anotación `@Path` debe declararse siempre después de la anotación `@GET`

Puerto en uso

- Sobre todo en vuestros ordenadores personales, puede que tengáis alguna aplicación usando el puerto 8080
  - Buscadlas y cerradlas
    - En GNU/Linux se pueden ver los puertos en activo con `nestat` o `lsof`
  - O bien cambiad el puerto en que arranca Tomcat
    - En Eclipse, en la pestaña Servers, haciendo doble clic en el servidor se puede cambiar su configuración.

ClassNotFoundException

Jersey

- `java.lang.ClassNotFoundException: com.sun.jersey.spi.container.servlet.ServletContainer`
- Los jar de Jersey no pueden meterse en cualquier carpeta, deben ir directamente en `WebContent/WEB-INF/lib`
  - Sin subcarpetas!

ContainerException – no root

Jersey

- `com.sun.jersey.api.container.ContainerException: The ResourceConfig instance does not contain any root resource classes.`
- El parámetro `com.sun.jersey.config.property.packages` en *web.xml* debe tener como valor el paquete que contiene los servicios en el proyecto

No injection source

REST

- `[[FATAL] No injection source found for a parameter of type XXX at index 0.;`
  - Este problema suele darse por tener como parámetro de la interfaz un tipo no básico (p. ej. un List, HashMap, etc.)

Unsupported major.minor version

REST

- `Caused by: java.lang.UnsupportedClassVersionError: org/glassfish/jersey/servlet/ServletContainer : Unsupported major.minor version 51.0 (no puedo cargar clase org.glassfish.jersey.servlet.ServletContainer)`
  - Suele indicar que estamos usando una versión de jersey compilada en una versión de Java más moderna que la versión de Java en la que ejecutamos nuestro proyecto.
  - Solución: recompilar con la misma versión java con la que se ejecuta.
  - Posiblemente se deba a el desarrollo del proyecto en tu propio ordenador, al migrarlo al laboratorio
  - Recuerda que en el laboratorio tenemos java 1.8, cambia la configuración de la ruta a la JRE y la versión de compilación si usas un proyecto importado de Eclipse.

ServletContainer cannot be cast

REST

- `Caused by: class org.glassfish.jersey.servlet.ServletContainer cannot be cast to class jakarta.servlet.Servlet`
  - Hemos detectado que este fallo ocurre cuando la versión de Tomcat es más moderna que la versión de jersey.
  - En particular, sabemos que ocurre si tenemos Tomcat 10 y jersey 2.3 (Eclipse EE 2020)
  - Solución: Usar Tomcat 9.

startup.sh no responde en ssh

Despliegue: tomcat

- Cuando ejecuto startup desde ssh con una orden como:
  - `ssh 127.0.0.1 "/ruta/a/tomcat/bin/startup.sh"`
- Me indica que tomcat se ha arrancado, pero luego si accedo a http://localhost:8080 veo que tomcat no está arrancado
- Solución:
  - Si te fijas en la salida de tomcat, es distinta a cuando la ejecutas sin ssh

```
Using CATALINA_BASE: /ruta/a/tomcat
Using CATALINA_HOME: /ruta/a/tomcat
Using CATALINA_TMPDIR: /ruta/a/tomcat/tmp
Using JRE_HOME: /opt/jdk1.6.0
```

en local

```
Using CATALINA_BASE: /ruta/a/tomcat
Using CATALINA_HOME: /ruta/a/tomcat
Using CATALINA_TMPDIR: /ruta/a/tomcat/tmp
Using JRE_HOME: /usr
```

con ssh 127.0.0.1

startup.sh no responde en ssh (ii)

Despliegue: tomcat

- ¿Por qué, si las dos son llamadas ‘locales’, aunque sea a través de ssh?
  - ssh no ejecuta archivos .profile o .bashrc que puedan definir variables de entorno
  - En concreto, no definiremos la versión de java a ejecutar
- En el laboratorio suele haber varias versiones de Java
  - Basta ejecutar `whereis java` para comprobarlo
- Solución: exportar la versión correcta de java en el ssh:
  - `ssh 127.0.0.1 ‘export PATH=$PATH:/opt/jdk1.6.0/bin; /ruta/a/tomcat/bin/startup.sh’`

startup.sh no responde en ssh (iii)

Despliegue: tomcat

- Otra solución: crear variables de entorno en Tomcat:
  - Crear un archivo `setenv.sh` dentro de la carpeta `bin` de Tomcat, con las siguientes líneas*:

```
export JAVA_HOME=/opt/jdk1.8.0_60/

export JRE_HOME=/opt/jdk1.8.0_60/jre
```

* La ruta exacta puede cambiar, hay que asegurarse primero de cuál es la ruta en la que está instalado java, por ejemplo con `whereis java`

No consigo compartir claves

Despliegue: ssh

- Consigo ejecutar el script `shareKeys.sh` disponible en http://vis.usal.es/rodrigo/documentos/sisdis/scripts
  - Y comparte bien las claves
  - Pero al ejecutar otros script me las vuelve a pedir
- Solución: las sentencias al final del script anterior (`ssh-agent` y `ssh-add`) deben añadirse al inicio de cualquier otro script que quiera utilizar el sistema de claves exportadas (por ejemplo, si queremos ejecutar un ssh o scp desde una orden ssh)

No consigo acceder a variables

Despliegue: ssh

- Cuando ejecuto una orden ssh que contiene alguna variable del servidor destinto, esta aparece vacía:
  - `ssh “echo $variable”`
- Las comillas dobles (“) resuelven los caracteres de escape (\) y los nombres de variables ($) *antes* de interpretar el texto. Es decir, `$variable` se resuelve como una variable local antes de ejecutar ssh
- Solución: utilizar comillas simples (‘), que toman el texto literalmente:
  - `ssh ‘echo $variable’`

Tomcat en distintos puertos

- El puerto en el que escucha Tomcat es configurable en `server.xml`, en el tag `Connector`:

```
<Connector port="8080" protocol="HTTP/1.1"
connectionTimeout="20000" RedirectPort="8443" />
```

- También es necesario cambiar los otros dos puertos que aparecen en el `server.xml`:

```
<Server port="8005" shutdown="SHUTDOWN"> [...]

<Connector port="8009" protocol="AJP/1.3"
redirectPort="8443" />
```

- En resumen, tenemos que duplicar la carpeta Tomcat y en una de ellas cambiarle a otros puertos, y luego ejecutar ambos Tomcats

NOTA: esta solución es más compleja que utilizar un despachador de procesos en un solo Tomcat, pero puede resultar más elegante

Tomcat en distintos puertos

Instancias de Jersey

- ¿Por qué no un sólo Tomcat escuchando en dos puertos distintos, mediante la creación de otro `Connector`?
  - Porque Jersey, si tiene clases `Singleton`, **no** va a crear dos instancias de ellas, una por puerto
  - Si queremos dos instancias, tendríamos que usar dos Tomcats, cada uno escuchando en su puerto

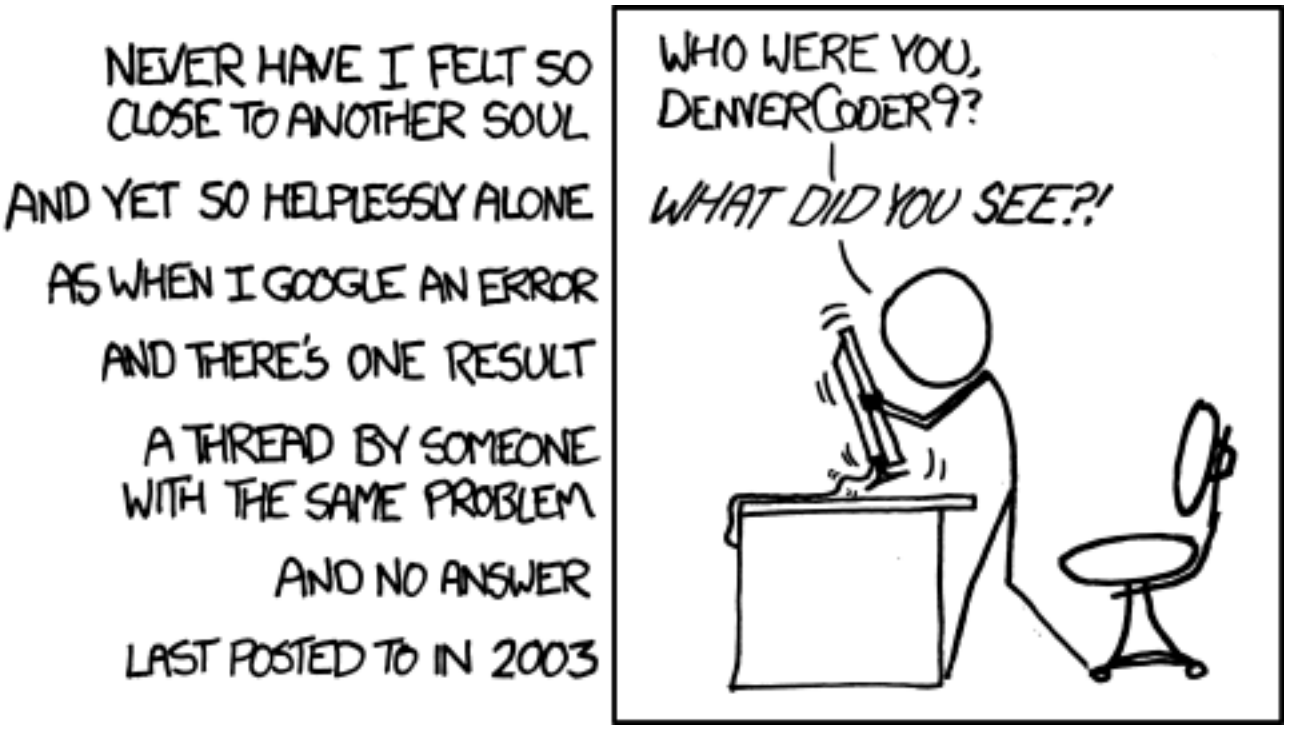
Tomcat en distintos puertos

Distinguiendo el puerto en Jersey

- Cualquier servicio puede tener acceso a la cabecera de la petición HTTP, que contiene el puerto al que se invoca
- En Jersey 1.x, mediante la etiqueta `@Context HttpHeaders` o `@HeaderParam("Host")`
  - https://dzone.com/articles/using-context-jax-rs
- En Jersey 2, la cosa se complica, es necesario implementar la interfaz `ClientResponseFilter`
  - `http://www.hascode.com/2013/12/jax-rs-2-0-rest-client-features-by-example/`

Aportaciones

- Según se vayan identificando errores y su causa, cread un hilo en el foro de Studium con un título indicativo del error y en el cuerpo:
  - Stacktrace (si procede)
  - Causa
  - Solución



https://xkcd.com/979/