

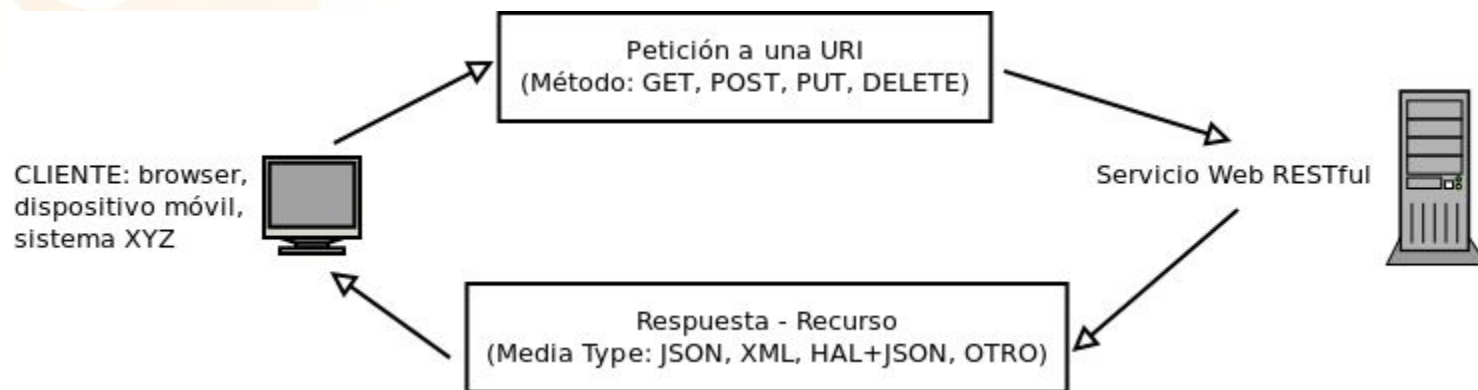


RESTful en Drupal 8

Creando Servicios Web desde el Core

RESTful - Comunicación

- Cuando hablamos de RESTful denotamos comunicación entre sistemas.
- Podemos tener, por ejemplo, un *Servicio Web RESTful* que nos entregue información de tickets



RESTful - Recursos (1)

- Un sistema que implementa un Servicio Web RESTful nos da acceso a recursos. A través de una URI: [Identificador Uniforme de Recurso](#)

`http://www.example.com/ticket/1`

- Los recursos son los “sustantivos”

RESTful - Recursos (2)

- Ejemplos de recursos:
 - Tickets
 - Noticias
 - Eventos
 - Platos de un restaurante
 - Mensajes
- Ser cualquier cosa que nos imaginemos, siempre y cuando la podamos *representar* de alguna manera.

RESTful - Formatos Base y Media Type (1)

- Los formatos son el “tipo” estructura en el que se envían los datos (petición y respuesta): json, xml. Un sistema RESTful puede usar uno o ambos.
- Cuando se envía, en la petición se expresa como Media Type: application/json o application/xml.
- También hay otros Media Type como:
 - application/hal+json
 - application/vnd.github+json
- Son más preparados para consumo de otras “máquinas” y no tanto para ser humano.

RESTful - Formato Base y Media Type (2)

- El media type debe indicarse en la cabecera, y no en el URI.

<http://www.example.com/ticket/809.json> (no recomendado)

<http://www.example.com/ticket/809> (recomendado)

En la cabecera lleva: Accept: application/json

Esto está en relación con los niveles de madurez de un sistema RESTful de acuerdo al modelo de Richardson

RESTful - Métodos (1)

- Los Servicios Web RESTful exponen métodos para realizar operaciones sobre los recursos. Son los “verbos” que indican acciones.
- Para ello estos Servicios Web usan explícitamente métodos de HTTP: GET, POST, PUT, DELETE. La forma usual es:
 - GET para obtener recursos, sin modificarlos. Pueden ser tipo listado, a través de parámetros de consulta.
 - POST usualmente usado para grabar nuevo(s) recurso(s)
 - PUT para cambiar el estado o actualizar datos
 - DELETE para borrar

RESTful - Métodos (2)

- Esto se parece mucho a un CRUD
- Drupal hace uso de estos métodos.
 - GET, para obtener recursos, sin modificarlos
 - POST, para enviar información nueva
 - PATCH, para actualizar
 - DELETE, para borrar
- Los utiliza de esta manera, aunque a muchos les parece un poco extraña esta elección.

RESTful - Métodos (3)

- No todos los sistemas son tan ordenados y algunos usan los métodos de forma indiscriminada.
- Por ejemplo hay muchos sistemas que usan GET para grabar nuevos datos.

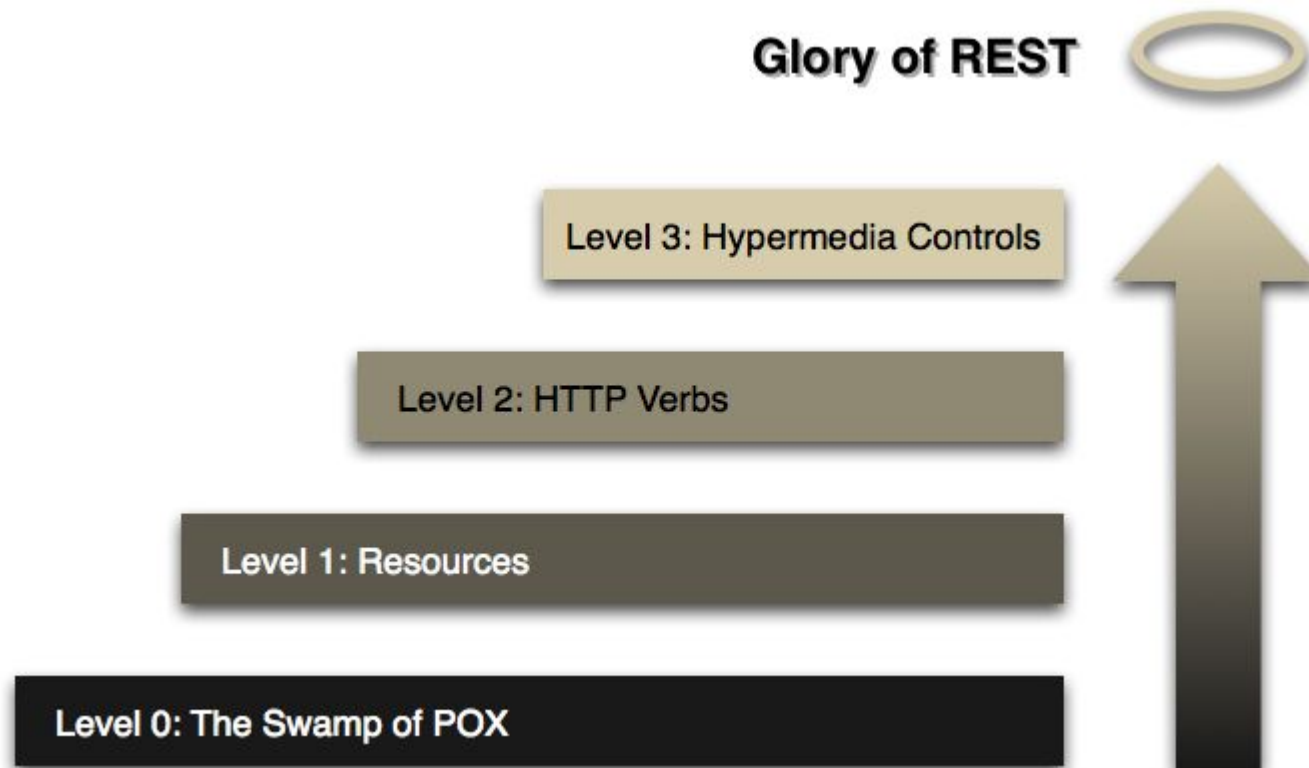
```
GET /agregarusuario?nombre=Jair HTTP/1.1
```

También tiene relación con los niveles de madurez del
modelo de Richardson

RESTful - Arquitectura REST

- RESTful, son sistemas que hacen uso de los principios de arquitectura definidos por REST
- Esta arquitectura plantea (sin ser exhaustivo) diferentes aspectos:
 - Un protocolo cliente/servidor sin estado
 - Un conjunto de operaciones bien definidas - métodos
 - Una sintaxis universal para identificar los recursos - URI
 - Uso de hipermedios
- Mientras más siga estos principios podríamos decir que son más RESTful..

Modelo de Madurez de RESTful - L. Richardson



Para qué puede servirme

- Sistemas que trabajen con dispositivos móviles
- Sitios que se comuniquen entre sí
- Sistemas Front-End que usen otras tecnologías que consuman datos de Drupal como Backend: *Drupal Headless*
- Sistemas que consuman datos de muchos Drupal para presentar un resumen
- Sistemas que actualicen o creen datos en otros Drupal (llenar contenido a través de/hacia otros sistemas u otros Drupal)



Lo que tenemos en Drupal

Módulos RESTful Web Service y Serialization

- Activamos los módulos RESTful Web Services y Serialization.
- El módulo Serialization es requerido por RESTful, por lo que tenemos que activarlo. Su función es convertir los datos en el formato que necesitemos, por ejemplo JSON.
- El módulo RESTful Web Services es el que crea los servicios que exponen los datos.

Módulos RWS y Serialization - captura

▼ WEB SERVICES

- ☐ **HAL** ▶ Serializes entities using Hypertext Application Language.
- ☐ **HTTP Basic Authentication** ▶ Provides the HTTP Basic authentication provider
- ☒ **RESTful Web Services** ▶ Exposes entities and other resources as RESTful web API
- ☒ **Serialization** ▶ Provides a service for (de)serializing data to/from formats such as JSON and XML

Install

Módulo RESTful Web Services - permisos



RESTful Web Services

Access DELETE on *Content* resource

Access GET on *Content* resource

Access PATCH on *Content* resource

Access POST on *Content* resource

Módulo REST UI

Activar el módulo contribuido REST UI:

<https://www.drupal.org/project/restui>

▼ WEB SERVICES

- | | | |
|-------------------------------------|----------------------------------|--|
| ☐ | HAL | ► Serializes entities using Hypertext Application Language. |
| ☐ | HTTP Basic Authentication | ► Provides the HTTP Basic authentication provider |
| ☒ | REST UI | ► Provides a user interface to manage REST resources |
| ☒ | RESTful Web Services | ► Exposes entities and other resources as RESTful web API |
| ☒ | Serialization | ► Provides a service for (de)serializing data to/from formats such as JSON and XML |

RESTful UI - Configuración

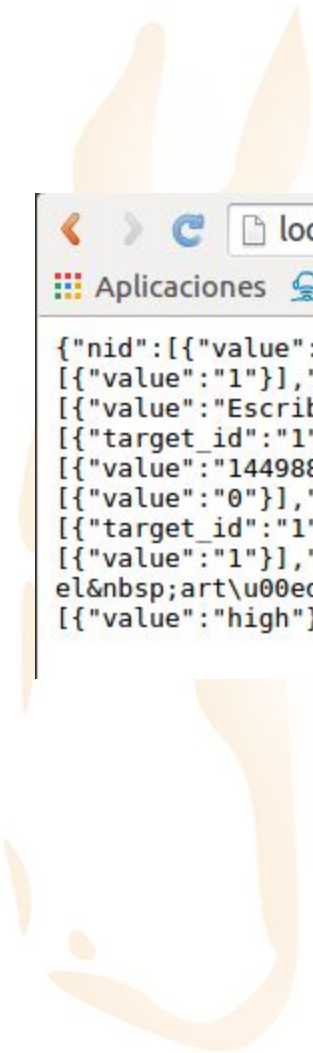
Here you can enable and disable available resources. Once a resource has been enabled, you can restrict its formats and authentication by clicking on its "Edit" link.

Enabled

RESOURCE NAME	PATH	DESCRIPTION	OPERATIONS
Content	//node/{node}	GET authentication: cookie formats: json POST authentication: cookie formats: json DELETE authentication: cookie formats: json PATCH authentication: cookie formats: json	Edit

Disabled

RESOURCE NAME	PATH	DESCRIPTION	OPERATIONS
Action	/entity/action/{action}		Enable
Base field override	/entity/base_field_override/{base_field_override}		Enable
Block	/entity/block/{block}		Enable



Prueba con extensión DHC de Chrome



REQUEST

HTTP `localhost/ejemplorest/node/1?_format=json` GET [? \[1\]](#) [Send](#)

HEADERS [set an authorization](#)

BODY
XHR does not allow an entity-body for GET request.
or change a method definition in [settings](#).

RESPONSE [elapsed time 238ms](#)

200 OK

HEADERS

Cache-Control:	must-revalidate, no-cache, post-check=
Connection:	Keep-Alive
Content-Language:	en
Content-Length:	771 Bytes
Content-Type:	application/json
Date:	2015 Dec 12 06:08:33 -1s
Expires:	1978 Nov 19 00:00:00 -37 years
Keep-Alive:	timeout=5, max=100
Server:	Apache/2.4.7 (Ubuntu)
X-Content-Type-Opti...	nosniff
X-Drupal-Dynamic-C...	HIT
X-Frame-Options:	SAMEORIGIN
X-Generator:	Drupal 8 (https://www.drupal.org)
X-Powered-By:	PHP/5.5.9-1ubuntu4.11
X-UA-Compatible:	IE=edge

[COMPLETE REQUEST HEADERS](#)

BODY [pretty](#)

```
{
  "nid": [
    {
      "value": "1"
    }
  ],
  "uuid": [
    {
      "value": "12985159-da2d-4a19-8c42-66ba45332262"
    }
  ],
  "vid": [
    {
      "value": "1"
    }
  ]
}
```

[Top](#) [Bottom](#) [Collapse](#) [Open](#) [2Request](#) [Copy](#) [Download](#)

Protocolos de autenticación

El Core de Drupal incluye los protocolos de autenticación:

- Cookie
- HTTP Basic

Con la ayuda de módulos contribuidos podemos tener además:

- OAuth
- Simple OAuth
- Otros



Gracias

Ricardo Chang