



ESPECIFICACIÓN SERVICIO REST PARA LA ENTIDAD PERSONAJE

Pacto de Honor

Facultad de Ingeniería y Ciencias básicas.
Politécnico Grancolombiano.

Contenido

Control de versiones	3
Objetivo	4
URL	4
Métodos	5
find	5
Satisfactorio:	6
No satisfactorio:	6
create	7
Satisfactorio:	8
No satisfactorio:	8
edit	9
Satisfactorio:	9
No satisfactorio:	9
getData	10
Satisfactorio.....	10
No satisfactorio	10
count	11

Control de versiones

Versión	Fecha	Autor	Revisado por	Cambios realizados
0.1	11/04/2017	Nicolás Rubiano		Versión inicial del documento.
0.2	19/04/2017	Nicolás Rubiano		Se agrega el caso no satisfactorio para el método getData

Objetivo

El objetivo del presente documento, es brindar herramientas de conocimiento los desarrolladores que desean consumir los servicios REST que proporcionará el área de desarrollo de Backend, específicamente sobre la entidad Personaje.

Los siguientes, son las especificaciones que se proporcionarán para tener acceso a todos los datos referentes sobre la entidad:

URL

<http://IP:Puerto/Polifight/webresources/personaje>

Al consumir la URL directa del servicio, por método GET, para la entidad Personaje, se listarán todos los datos que se encuentren en la DB:

```
1  [
2  {
3    "costo": 500,
4    "descripcionPersonaje": "personaje 01 cargue ejemplo",
5    "idCategoria": 1,
6    "idImagen": 1,
7    "idPersonaje": 1,
8    "nivelDano": 50,
9    "nombrePersonaje": "juan rata"
10  },
11  {
12    "costo": 1000,
13    "descripcionPersonaje": "personaje 02 cargue ejemplo",
14    "idCategoria": 2,
15    "idImagen": 2,
16    "idPersonaje": 2,
17    "nivelDano": 100,
18    "nombrePersonaje": "pepe grillo"
19  },
20  {
21    "costo": 100,
22    "descripcionPersonaje": "Descripción del nuevo personaje",
23    "idCategoria": 1,
24    "idImagen": 1,
25    "idPersonaje": 7,
26    "nivelDano": 3,
27    "nombrePersonaje": "Pedro Conejo"
28  },
29  ]
```

Métodos

find

EndPoint: /Polifight/webresources/personaje/find

Este método se deberá consumir por método GET. Se recibirá el id (Parámetro "id") del Personaje y retornará todos los atributos asociados a dicho personaje:



The image shows a code editor window with a sidebar on the left containing tabs for 'Raw', 'HTML', 'JSON', and 'XML'. The 'JSON' tab is selected. The main editing area displays a JSON object with the following properties: 'idPersonaje' (1), 'idCategoria' (1), 'idImagen' (1), 'nombrePersonaje' (juan rata), 'descripcionPersonaje' (personaje 01 cargue ejemplo), 'costo' (500), and 'nivelDano' (50). The first line of the JSON object is highlighted in yellow.

```
1 {  
2   "idPersonaje": "1",  
3   "idCategoria": "1",  
4   "idImagen": "1",  
5   "nombrePersonaje": "juan rata",  
6   "descripcionPersonaje": "personaje 01 cargue ejemplo",  
7   "costo": "500",  
8   "nivelDano": "50"  
9 }
```

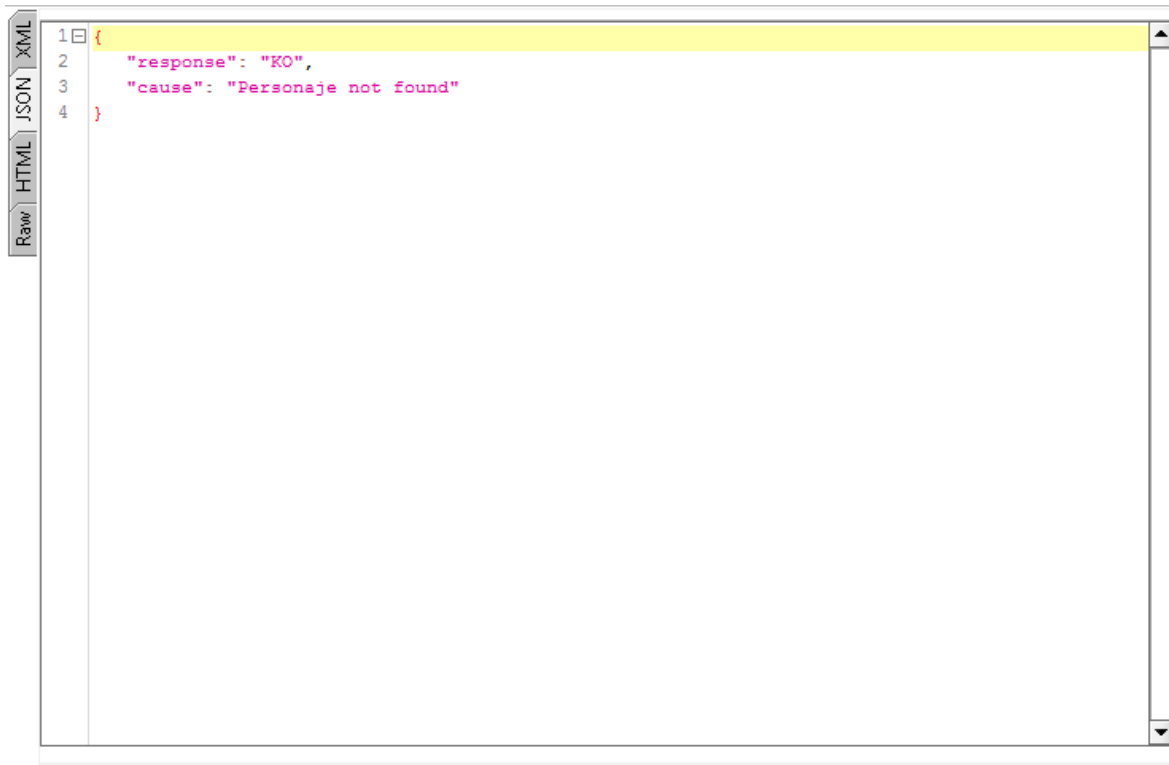
Satisfactorio:

En el caso satisfactorio, se enviarán los atributos de la entidad en formato JSON.

No satisfactorio:

Si el servicio tiene algún problema con la solicitud, se responderán dos campos:

- "response": En caso de algún inconveniente se responderá "KO".
- "cause": Causa del error, para éste método se identifica sólo 1 error, Id del personaje no encontrado en la DB



A screenshot of a code editor with a sidebar on the left containing tabs for 'Raw', 'HTML', 'JSON', and 'XML'. The 'JSON' tab is selected. The editor displays a JSON object with the following content:

```
1 {  
2   "response": "KO",  
3   "cause": "Personaje not found"  
4 }
```

create

EndPoint: /Polifight/webresources/personaje/create

Se debe enviar un JSON por el método POST con los atributos de la entidad Personaje:



A screenshot of a REST client interface. At the top, there is a 'Media Type' dropdown menu set to 'application/json', a 'Post QueryString' checkbox which is unchecked, and a 'Post Body' icon. Below this, the request body is shown in a text area with the following JSON content:

```
{  
  "nombrePersonaje": "Miguel Vaca",  
  "descripcionPersonaje": "Descripción de la vaca",  
  "costo": "230",  
  "nivelDano": "20",  
  "idImagen": "1",  
  "idCategoria": "1"  
}
```

Es importante recalcar que los nombres de los atributos que se envían deben tener los nombres exactos que muestran en la imagen anterior.

Satisfactorio:

El servicio responde OK en el campo “response” en caso de que la creación se haya realizado con éxito:



Result Grid							
Filter Rows:		Edit:		Export/Import:		Wrap Cell Content:	
	id_personaje	id_categoria	id_imagen	nombre_personaje	descripcion_personaje	costo	nivel_dano
	7	1	1	Pedro Conejo	Descripción del nuevo personaje	100	3
	8	1	1	Pedro Pez	Descripción del nuevo personaje	100	3
	9	1	1	Miguel Vaca	Descripción de la vaca	230	20
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

No satisfactorio:

Si el servicio no puede procesar la solicitud se responderá un KO en el campo “response”. En este caso se incluirá el campo “cause” donde se indicará la causa del error

```
{
  response: KO
  cause: Time Out
}
```


edit

EndPoint: /Polifight/webresources/personaje/edit

Método que servirá para modificar una entidad Jugador. Se debe consumir por POST enviando un formato JSON con el parámetro "id", éste es obligatorio, y seguido de los atributos que se deseen modificar.

Para efectos de ejemplo, modificaremos el nombre del dato insertado anteriormente en el ejemplo del método [create](#), por "Lola Vaca" enviando el siguiente JSON:

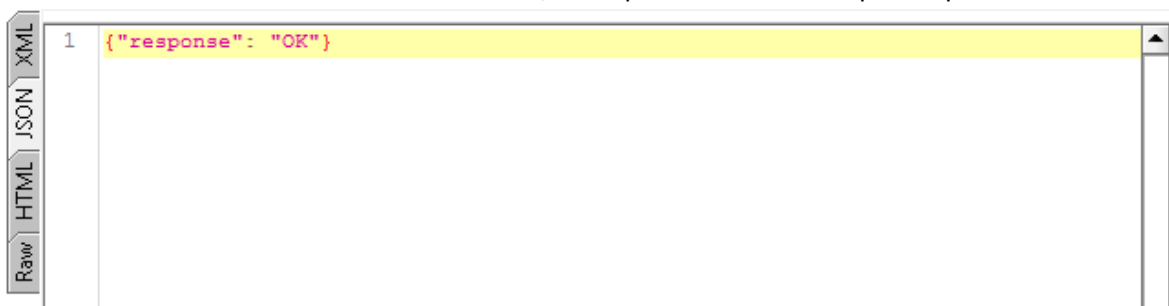


The screenshot shows a REST client interface. At the top, the 'Media Type' is set to 'application/json' and the 'Post' checkbox is checked. The request body is a JSON object with two fields: 'idPersonaje' with the value '9' and 'nombrePersonaje' with the value 'Lola Conejo'.

```
{
  "idPersonaje": "9",
  "nombrePersonaje": "Lola Conejo"
}
```

Satisfactorio:

Si la modificación al atributo fue satisfactorio, se responderá con el campo "response": "OK":



The screenshot shows a REST client interface with the 'Raw' tab selected. The response is a JSON object with a single field: 'response' with the value 'OK'.

```
1 {"response": "OK"}
```

No satisfactorio:

Si el servicio no puede procesar la solicitud se responderá un KO en el campo "response". En este caso se incluirá el campo "cause" donde se indicará la causa del error



A screenshot of a web client interface. On the left, there are tabs for 'Raw', 'HTML', 'JSON', and 'XML'. The 'JSON' tab is selected. The main area displays a JSON object with the following structure:

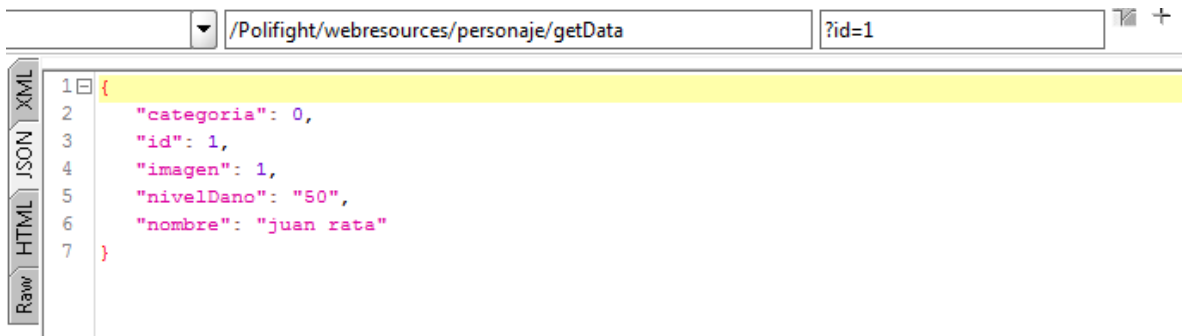
```
1 {
2   "response": "KO",
3   "cause": "Personaje not found"
4 }
```

getData

URL: /Polifight/webresources/personaje/getData

Este método es solicitado con una estructura específica por el equipo de FrontEnd el día Miércoles 5 de Abril. Se consume por método GET y debe enviarse el parámetro "id" y retorna la siguiente estructura:

Satisfactorio




A screenshot of a web client interface. The address bar shows the URL `/Polifight/webresources/personaje/getData` with a query parameter `?id=1`. On the left, there are tabs for 'Raw', 'HTML', 'JSON', and 'XML'. The 'JSON' tab is selected. The main area displays a JSON object with the following structure:

```
1 {
2   "categoria": 0,
3   "id": 1,
4   "imagen": 1,
5   "nivelDano": "50",
6   "nombre": "juan rata"
7 }
```

No satisfactorio

En caso de que no se envíe un id que se encuentre en la base de datos o que se ocasione algún tipo de excepción, el servicio retornará un JSON con dos campos:

`webresources/personaje/getData?id=200]`



A screenshot of a web client interface. The address bar shows the URL `webresources/personaje/getData?id=200]`. On the left, there are tabs for 'Raw', 'HTML', 'JSON', and 'XML'. The 'JSON' tab is selected. The main area displays a JSON object with the following structure:

```
1 {
2   "response": "KO",
3   "cause": "Personaje not found"
4 }
```

count

EndPoint: /Polifight/webresources/personaje/count

Este método retorna la cantidad de datos que se encuentran en la DB. Se debe consumir por método GET directamente sin parámetros:

