# Documentación API REST Parking

## Como arrancar el servicio

### Servidor

Cambiamos el nombre de Parking.zip a Parking.jar

Ejecutamos el comando siguiente en el cmd:

Java –jar <pathAbsolutoAlJar>\Parking.jar

### Cliente

Vía telnet, nos conectaremos al servidor para hacer las pruebas:

> telnet

> open localhost 80

## Peticiones GET

Hacen una búsqueda de parkings en el sistema por cada uno de los atributos informados en la petición, además se puede añadir como condición si están llenos o no.

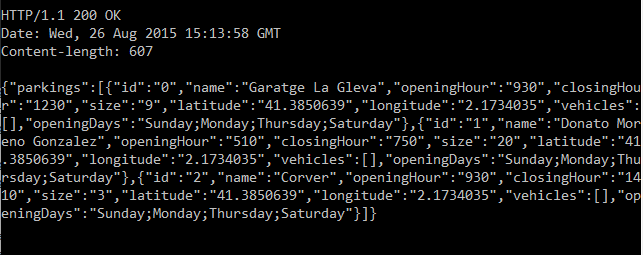
|  |  |  |
| --- | --- | --- |
| **Campo** | **Tipo** | **Descripción** |
| Id | Long | Identificador del parking |
| Name | String | Nombre del parking |
| openingHour | Integer | Hora de apertura |
| closingHour | Integer | Hora de cierre |
| size | Integer | Tamaño del parking |
| latitude | Double | Latitud en º |
| longitude | Double | Longitud en º |
| openingDays | String | Dias que abre |
| completo | Boolean | Si está completo o no |

### Ejemplos de llamadas:

* Obtener todos los parkings:

GET /parking HTTP/1.1 [enter]

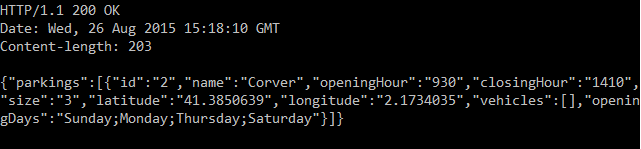
[enter]



* Obtener el parking con el identificador 2

GET /parking?id=2 HTTP/1.1 [enter]

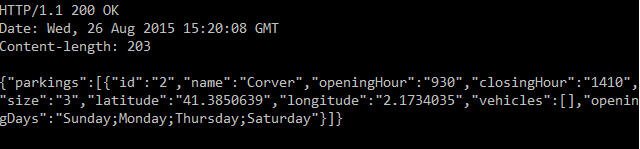
[enter]



* Obtener el parking con el identificador 2 y el nombre “Corver”

GET /parking?id=2&name=Corver HTTP/1.1 [enter]

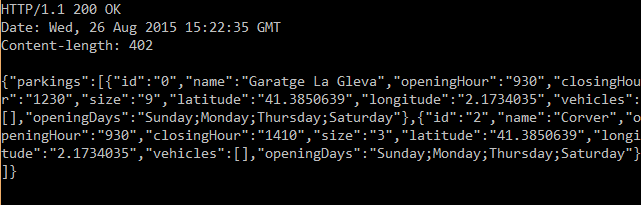
[enter]



* Obtener los parkings que abren a las 15:30 (930)

GET /parking?openingHour=930 HTTP/1.1 [enter]

[enter]



## Peticiones POST

Hacen un insert en el sistema de un elemento en concreto pasado por parámetros, utiliza los siguientes elementos:

|  |  |  |
| --- | --- | --- |
| **Campo** | **Tipo** | **Descripción** |
| Id | Long | Identificador del parking |
| Name\* | String | Nombre del parking |
| openingHour | Integer | Hora de apertura |
| closingHour | Integer | Hora de cierre |
| Size\* | Integer | Tamaño del parking |
| latitude | Double | Latitud en º |
| longitude | Double | Longitud en º |
| openingDays | String | Dias que abre |

* Campos primarios que son NOT NULL, no es posible realizar un insert sin informar la clave primaria.

Nos devolverá un Json con los datos que han sido persistidos.

### Ejemplos de llamadas:

* Añadir un parking:

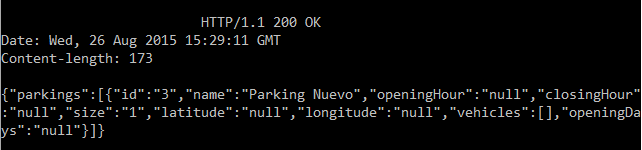
POST /parking HTTP/1.1 [enter]

Content-type: plain/text [enter]

Content-length: 25[enter]

[enter]

name=Parking nuevo&size=1 [al llegar aquí, por el context-length ya debería enviar sola la petición]



## Peticiones PUT

Con estas peticiones realizaremos un update de un parking que esté definido en el sistema, la petición devolverá en todo caso un JSON con el parking que ha sido modificado o uno vacío si no ha encontrado ninguno.

Los parámetros aceptados son los siguientes:

|  |  |  |
| --- | --- | --- |
| **Campo** | **Tipo** | **Descripción** |
| Id\* | Long | Identificador del parking |
| Name | String | Nombre del parking |
| openingHour | Integer | Hora de apertura |
| closingHour | Integer | Hora de cierre |
| Size º | Integer | Tamaño del parking |
| Latitude | Double | Latitud en º |
| longitude | Double | Longitud en º |
| openingDays | String | Dias que abre |
| addCarº | String | La matrícula del coche que acaba de entrar al parking |
| removeCarº | String | La matrícula del coche que acaba de salir del parking |

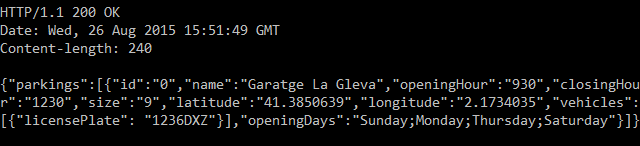
\*Campo obligatorio

º Una de estas opciones/atributos ha de estar activo

* Añadir un vehículo al parking

PUT /parking?id=0&addCar=1236DXZ HTTP/1.1[enter]

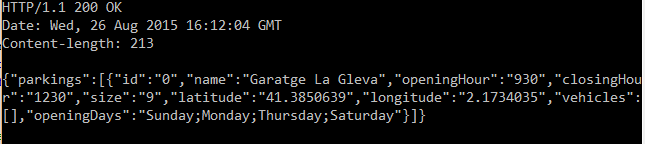
[enter]



* Quitar un vehículo al parking

PUT /parking?id=0&removeCar=1236DXZ HTTP/1.1[enter]

[enter]



#### NOTAS

Si no añades todos los atributos, al actualizar un registro perderás los datos anteriores

## Peticiones DELETE

Con esta petición borramos un parking. Los parámetros aceptados son los siguientes:

|  |  |  |
| --- | --- | --- |
| **Campo** | **Tipo** | **Descripción** |
| Id\* | Long | Identificador del parking |

\*Campo obligatorio

Como respuesta nos retorna el elemento eliminado o un json vacío si no ha habido elementos.

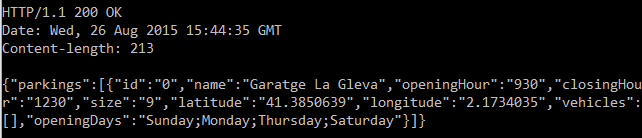
### Ejemplos de llamadas:

* Borrar un parking:

DELETE /parking?id=0 HTTP/1.1 [enter]

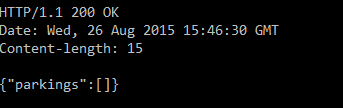
Content-type: plain/text [enter]

[enter]



Podemos realizar un get para el id 0 y podremos observar como no encontrará resultados:

GET /parking?id=0 HTTP/1.1[enter]



## NOTAS

No he realizado el desarrollo de los días, tan solo habría que encontrar el número de día que es esa fecha y encontrarlo en el enumSet de Day en Parking para saber si es apto o no.

No he utilizado la latitud ni longitud, no he querido alargar la práctica

He decidido centrarme en la búsqueda antes que en los otros métodos

El sistema de errores no es el de HTTP puesto que no quería cortar la respuesta, he realizado un código 200 para las peticiones con error para poder devolver el json de errores

En lugar de aumentar o disminuir plazas, he decidido que en el parking entran o salen coches, de esta forma es más natural el reducir o ampliar el número.