

Taller de Bases de Datos

# **Diseño de servicios REST**

---

# Contenidos

---

Introducción a REST

Recursos

Asociaciones

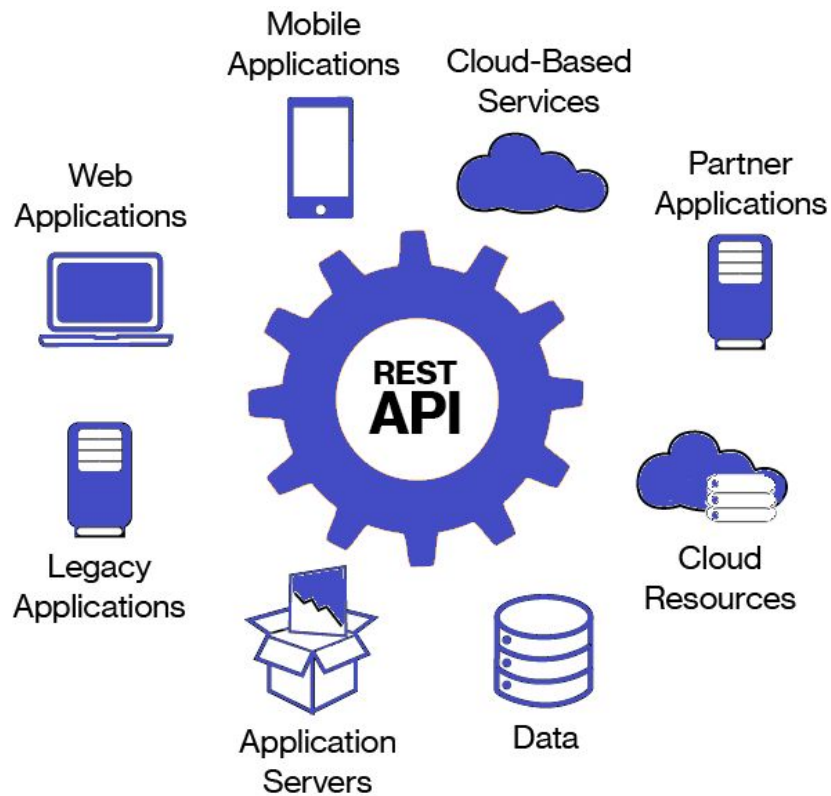
Variaciones más complejas

Paginación y otras configuraciones

Formato de mensaje

# ¿Qué es REST?

---



# ¿Qué es REST?

---

Es una arquitectura para crear servicios web.

Se caracteriza principalmente por ser **cliente/servidor sin estado** (*stateless*)

Una API que se adhiere a esta arquitectura tiene lo siguiente:

- URI: Identificador universal de recursos
- Métodos HTTP estándar (GET, POST, PUT, DELETE)
- Media type que define a los datos (ej: Json)

# Recursos

# Pensemos en perros

---



# Cuántas cosas podemos hacer



/traerTodosLosPerros

/necesitaComida

/traerTodosLosPerrosNuevos

/hacerQueTodosLosPerrosSeSienten

...

# Solamente necesitamos dos URL para un recurso

---

Para una colección:

**/dogs**

Para un elemento:

**/dogs/1**



# Pensando en bases de datos

---

POST

GET

PUT

DELETE

CREATE

READ

UPDATE

DELETE

# Nos resulta



Recurso	POST create	GET read	UPDATE update	DELETE delete
/dogs	Crea un nuevo perro	Lista los perros	¡Actualiza todos los perros!	¡Borra todos los perros!
/dogs/1	Error	Muestra a Chamuyo	Actualiza a Chamuyo si existe, si no; error	Borra a Chamuyo

# Nombrando recursos

---

- Preferir sustantivos a verbos
- Preferir plurales a singulares
- Más concreto que abstracto  
**/dogs** en vez de **/animals**

# **Asociaciones**

# Obtener todos los perros de un dueño

---

GET /owners/123/dogs



# Variaciones complejas

# Perros cafés corriendo en el parque

---

/dogs?**color**=brown&**state**=running&**location**=park

# **Paginación y otras configuraciones**





## Paginación

Si no se especifica, se debe mantener una paginación por defecto (ej:10)

```
/dogs?limit=10&offset=3
```

## Versionamiento

```
/v1/dogs
```



Casos en que no sea recurso sí se pueden usar verbos

`/convert?from=EUR&to=CLP&amount=100`

`/search?q=fluffy+fur`

`/dogs/count`

# **Formato de mensaje**

# Estructura de mensaje

---

Headers



```
HTTP/1.1 200  
Content-Type: application/json
```

Body



```
{  
  "id": 1,  
  "name": "Chamuyo",  
  "color": "red"  
}
```

# Formatos de mensaje

---

HTTP/1.1 200

Content-Type: application/json

```
{  
  "id": 1,  
  "name": "Chamuyo",  
  "color": "red"  
}
```

HTTP/1.1 200

Pagination-Count: 100

Pagination-Page: 5

Pagination-Limit: 20

Content-Type: application/json

```
[  
  {  
    "id": 1,  
    "name": "Chamuyo",  
    "color": "red"  
  },  
  {  
    "id": 2,  
    "name": "Moneda",  
    "color": "black"  
  }  
]
```

# Formatos de mensaje - errores

HTTP/1.1 404

Content-Type: application/json

```
{
  "message": "El elemento no existe"
}
```

HTTP/1.1 400

Content-Type: application/json

```
{
  "message": "Ocurrieron errores en su solicitud",
  "errors": [
    {
      "message": "El campo es inválido",
      "code": 34,
      "field": "email"
    },
    {
      "message": "El formato es incorrecto",
      "code": 35,
      "field": "phoneNumber"
    }
  ]
}
```

# Herramientas

# Postman

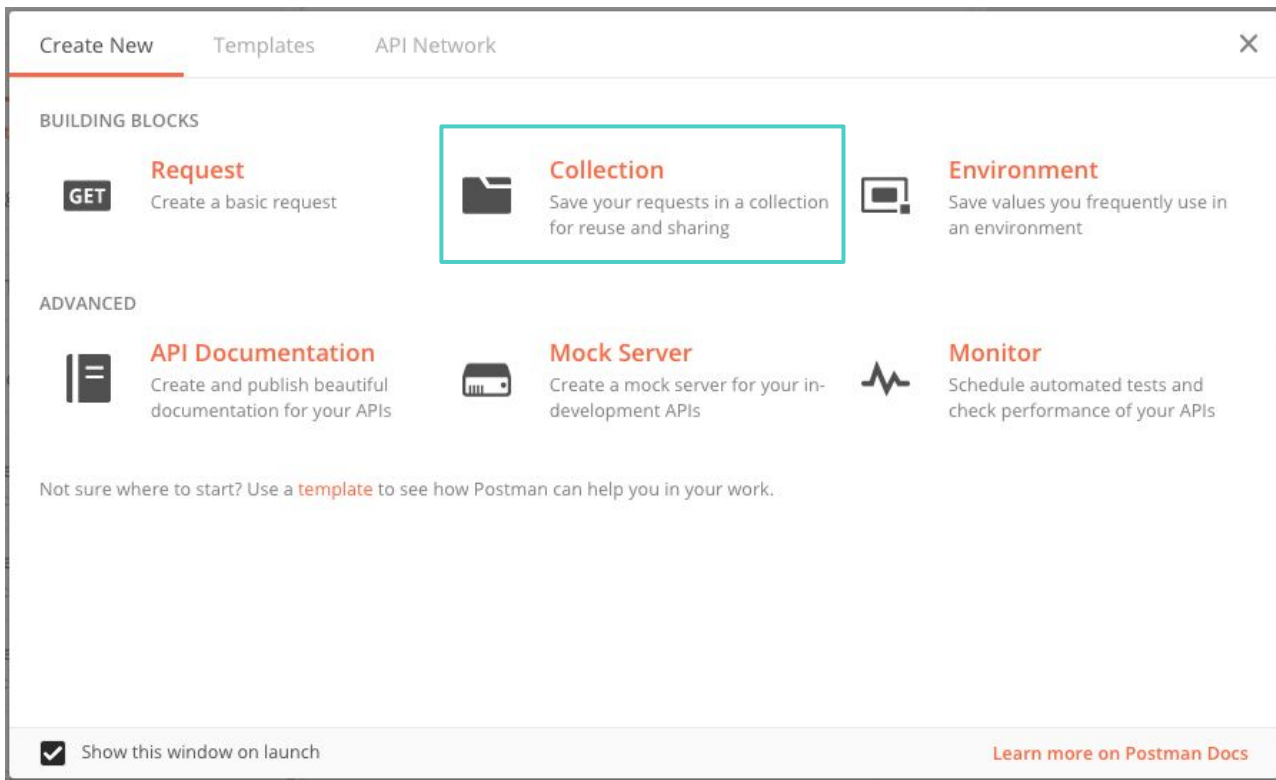
---

- Cliente API
- Pruebas de API
- **Diseño y Mockup de API**
- Documentación



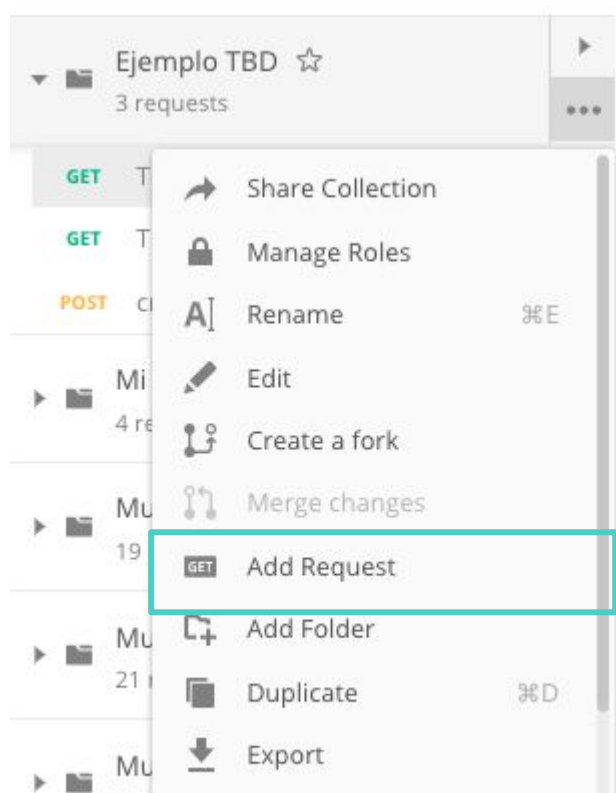


# Creando una colección



# Agregando requests

En la lista izquierda,  
hacer clic en “...” y  
luego en **Add Request**

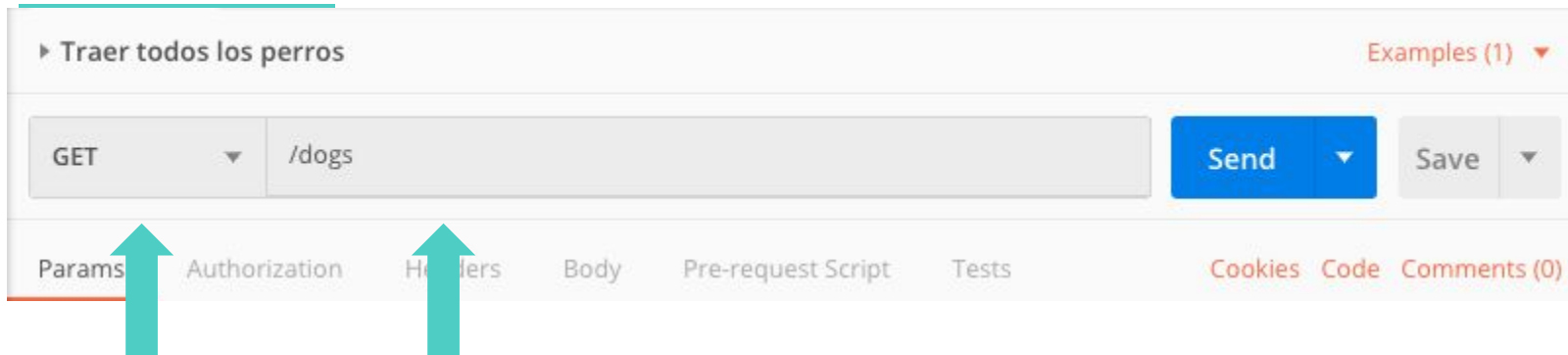


# Creando Request

► Traer todos los perros Examples (1) ▼

GET ▼	/dogs	Send ▼	Save ▼
-------	-------	--------	--------

Params **Authorization** Headers Body Pre-request Script Tests Cookies Code Comments (0)



**Método**

**URI**

# Agregando un ejemplo

## EXAMPLE REQUEST

GET

/dogs

Params

Headers (1)

Body

## EXAMPLE RESPONSE

Body

Headers

Status

200 OK

Pretty

Raw

Preview

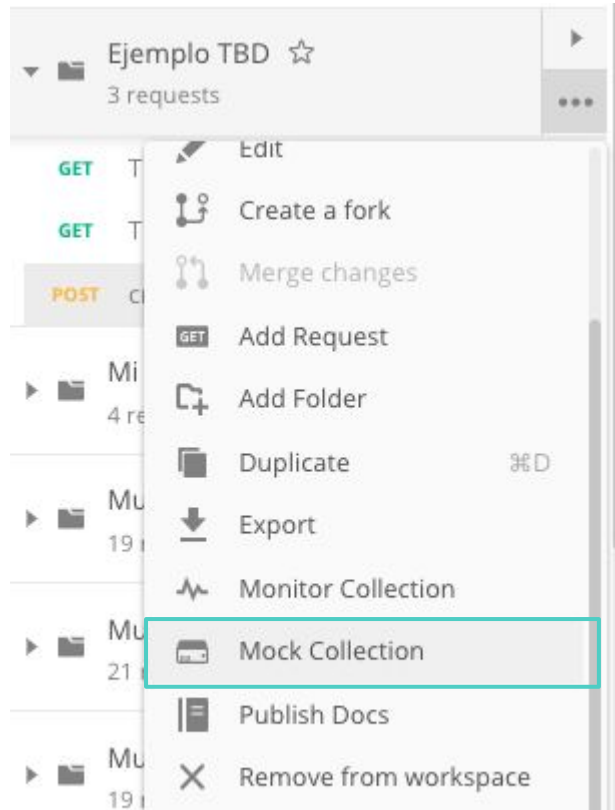
JSON



```
1 [
2   {
3     "id":1,
4     "name":"Chamuyo",
5     "color":"red"
6   },
7   {
8     "id":2,
9     "name":"Moneda",
10    "color":"black"
11  }
12 ]
```

# Crear un mockup

Las llamadas con ejemplos permiten generar un mockup de la API

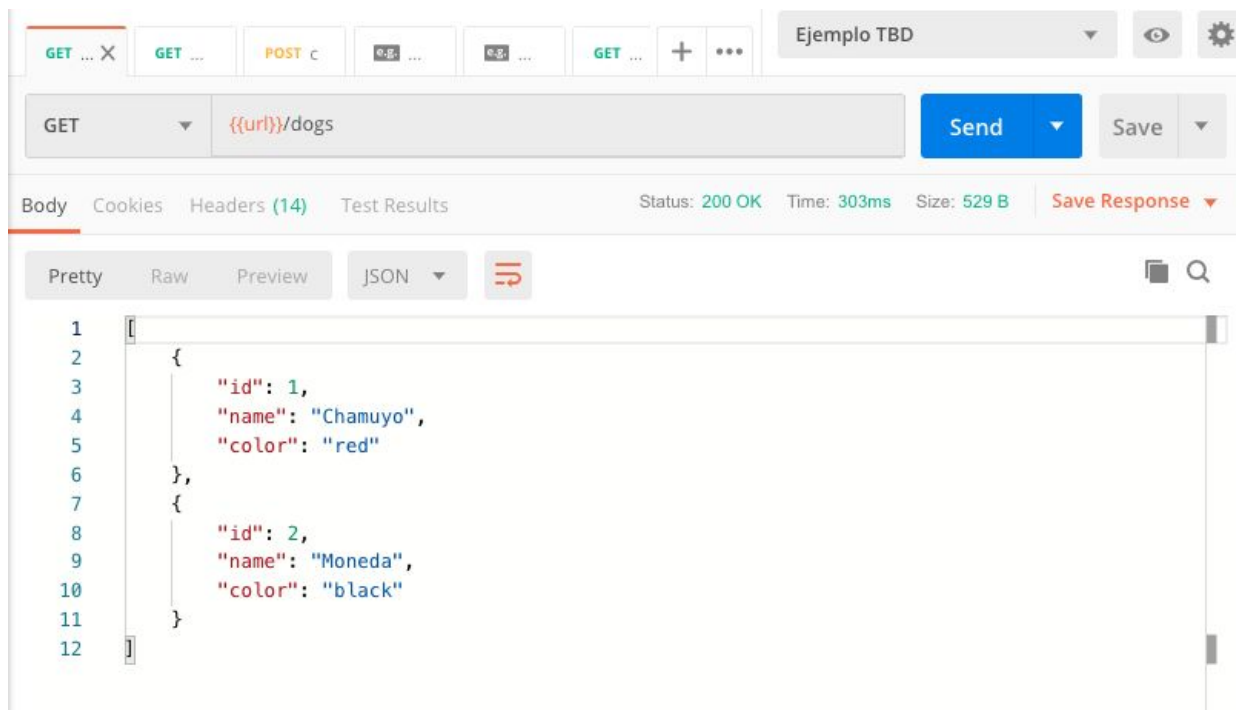


# Configurar variables

Ejemplo TBD			Edit
VARIABLE	INITIAL VALUE	CURRENT VALUE	
url	https://9153395c-26a1-4d06-bc87-6239160fbe01.mock.pstmn.io	https://9153395c-26a1-4d06-bc87-6239160fbe01.mock.pstmn.io	

GET	▼	{{url}}/dogs	Send	▼
-----	---	--------------	------	---

# Probar mockup



# Más información



<https://www.vinaysahni.com/best-practices-for-a-pragmatic-restful-api>

<https://www.slideshare.net/apigee/restful-api-design-second-edition>