



{ } RESTful API



REST API BÁSICO

Zapata Icart, Ernesto A.

Facultad Regional Paraná

Programación III



APLICACIONES TRADICIONALES WEB



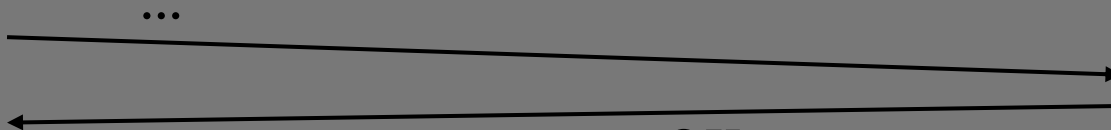
Cliente

GET /el-recurso



Server

Muestra la página
y el usuario
clickea el link.



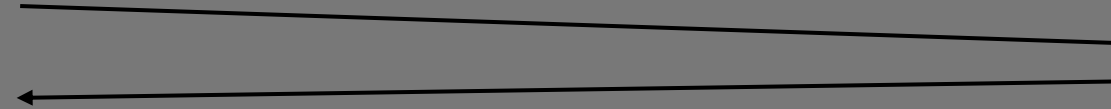
200 OK

`<html>código...</html>`

GET /otro-recurso

...

Muestra la otra
página, ...



200 OK

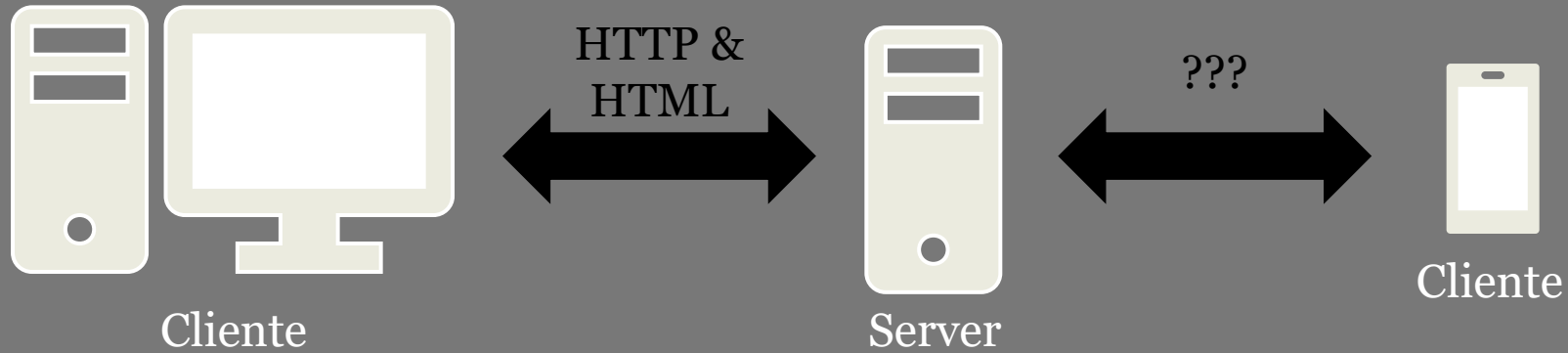
`<html>código...</html>`

APLICACIONES TRADICIONALES WEB

La interface se arma con HTML & HTTP.

- Inconvenientes:
 - El cliente debe entender HTTP y HTML.
 - Toda la página web se reemplaza con otra.
 - No hay manera de armar transiciones entre páginas.
 - Los mismos datos son usados en múltiples responses.
 - Ej.: Sólo código HTML para la capa de presentación.
-

APLICACIONES TRADICIONALES WEB

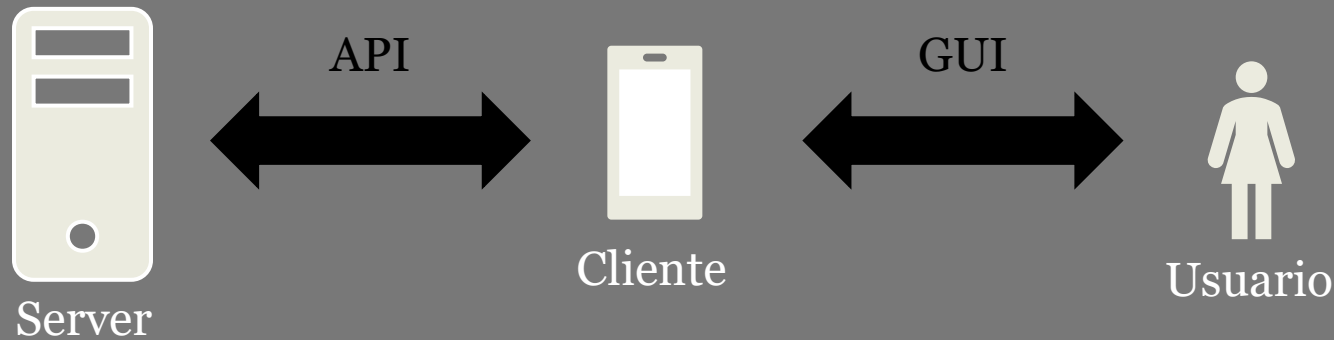


- Se puede usar HTTP & HTML pero no es óptimo.
 - Los teléfonos móviles no usan HTML.
 - Ej.: GET /users/3:

Nombre → `<h1>Ernesto</h1>`
`<p>Ernesto` ← `tiene` `58` ← `años y vive en` `Paraná` `</p>`
Edad Ciudad

APPLICATION PROGRAMMING INTERFACE

Una GUI es una interface para seres humanos ↔ Comunicación máquina.



Una API es una interface para una máquina ↔ Comunicación máquina

- Una API que hace uso de HTTP se llama *Web API*.
-

DIFERENTES TIPOS DE WEB APIS

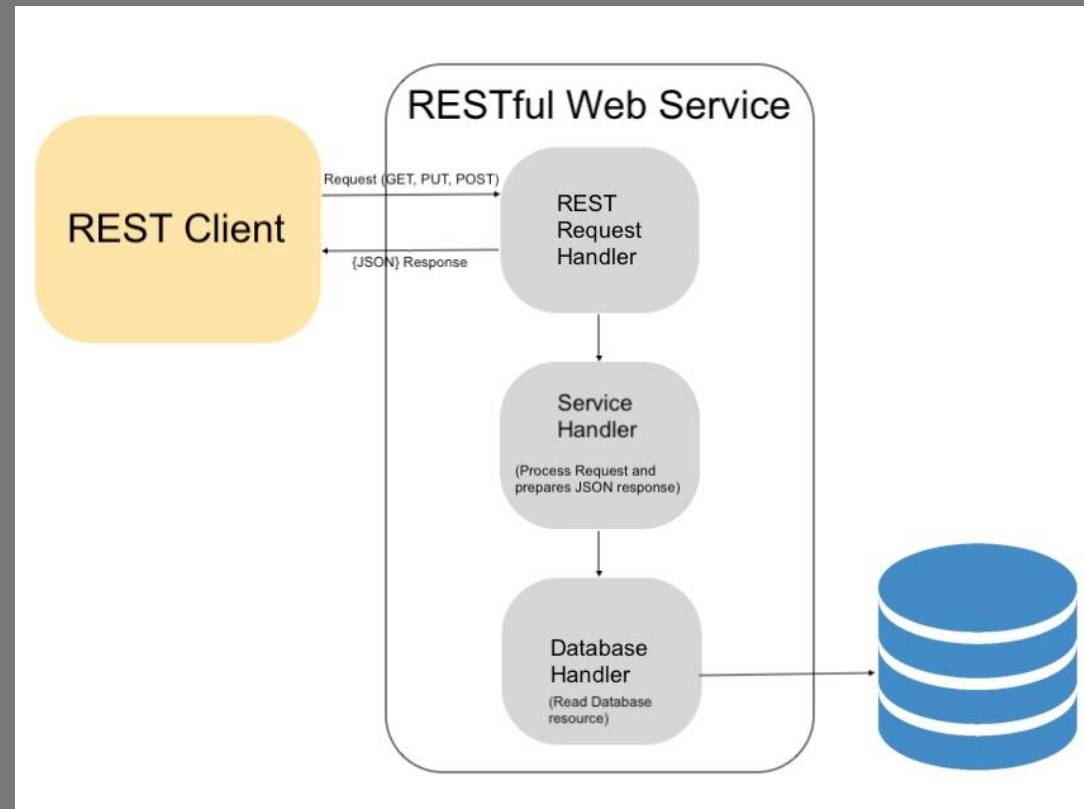
- *Remote Procedure Call*, RPC.
 - Los clientes pueden llamar a funciones en el server.
 - *Remote Method Invocation*, RMI.
 - Los clientes pueden llamar a métodos sobre objetos en el server.
 - *Representational State Transfer*, REST.
 - Los clientes pueden aplicar operaciones CRUD en recursos del server.
-

¿QUE ES REST?

Un estilo arquitectónico para *sistemas distribuidos* descrito por Roy Thomas Fielding en su tesis doctoral del año 2000.

- Restricciones:

1. Client - Server
2. Stateless
3. Cache
4. Interface Uniforme
5. Layered System
6. Code-On-Demand



¿QUÉ SIGNIFICA REST?

El nombre "Representational State Transfer" pretende evocar una imagen de cómo se comporta una aplicación web bien diseñada: una red de páginas web (una máquina de estado virtual), donde el usuario avanza a través de la aplicación seleccionando enlaces (transiciones de estado) , lo que da como resultado que la página siguiente (que representa el siguiente estado de la aplicación) se transfiera al usuario y se presente para su uso.

Del discurso de Roy.

¿QUÉ SIGNIFICA REST?



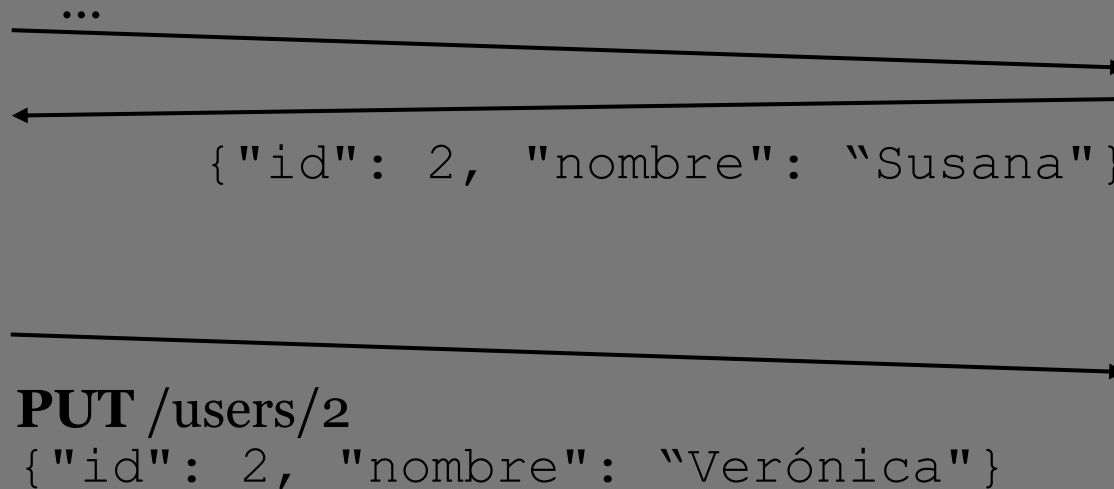
Cliente **GET** /users/2



Server

Cambia el estado.

```
{"id": 2,  
"nombre":  
"Verónica"}
```



Usuarios	
Id	Nombre
1	Ernesto
2	Susana
3	Luis

USANDO HTTP COMO INTERFACE UNIFORME

- Use URIs para identificar recursos.
 - Use métodos HTTP para especificar operaciones:

	<u>Mal</u>	<u>Bien</u>
• Create: POST (o PUT)	POST /login	POST /login-sessions
• Retrieve: GET		
• Update: PUT (o PATCH)	POST /create-libros	POST /libros
• Delete: DELETE	GET /get-top-10-libros	GET /top-10-libros
 - Use headers HTTP Content-Type y Accept para especificar formato de datos para los recursos.
 - Use los códigos de status HTTP para indicar éxitos/fallos.
-

USANDO HTTP COMO INTERFACE UNIFORME

REST es un estilo arquitectónico, no una especificación.

- En la práctica, se puede usar de diferentes maneras.
 - Pero algunas son mejores que otras.

Buenas recomendaciones:

- Web API Design - Crafting Interfaces that Developers Love
 - <https://pages.apigee.com/rs/apigee/images/api-design-ebook-2012-03.pdf>
-

EJEMPLO REST

Un server con información acerca de los usuarios.

- El método GET se usa para recuperar recursos.

- GET /users
- GET /users/2
- GET /users/pages/1
- GET /users/genero/femenino
- GET /users/edad/18
- GET /users/???
- GET /users/2/nombre
- GET /users/2/mascotas

GET /users?page=1

GET /users?genero=femenino

GET /users?edad=18

GET /users?genero=femenino&edad=18



Id	Nombre
1	Ernesto
2	Susana
3	Luis

Usuarios

EJEMPLO REST

Un server con información acerca de los usuarios


- El método GET se usa para recuperar recursos.
 - Con qué formato de datos? Se especifica en el header Accept!

```
GET /users HTTP/1.1
Host: the-website.com
Accept: application/json
```

application/xml
fue popular antes de
JSON.

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 66
```

```
[
  {"id": 1, "nombre": "Ernesto"},
  {"id": 2, "nombre": "Susana"}
]
```



Id	Nombre
1	Ernesto
2	Susana
3	Luis

Usuarios

EJEMPLO REST

Un server con información acerca de los usuarios

- El método POST se usa para crear recursos.
 - Con qué formato de datos? Especificado en los headers Accept y Content-Type!

Id	Nombre
1	Ernesto
2	Susana
3	Luis

Usuarios

```
POST /users HTTP/1.1
Host: the-website.com
Accept: application/json
Content-Type: application/json
Content-Length: 49
```

```
<user>
  <nombre>Augusto</nombre>
</user>
```

```
HTTP/1.1 201 Created
Location: /users/3
Content-Type: application/json
Content-Length: 28

{"id": 3, "nombre": "Augusto"}
```

EJEMPLO REST

Un server con información acerca de los usuarios

- El método PUT se usa para actualizar un recurso completo.

Id	Nombre
1	Ernesto
2	Susana
3	Luis

Usuarios

```
PUT /users/3 HTTP/1.1
Host: the-website.com
Content-Type: application/json
Content-Length: 52
```

```
<user>
  <id>3</id>
  <nombre>Cecilia</nombre>
</user>
```

```
HTTP/1.1 204 No Content
```

PUT puede ser usado para
crear un recurso si se
conoce de antemano la
URI.

EJEMPLO REST

Un server con información acerca de los usuarios

- El método DELETE se usa para borrar un recurso.

```
DELETE /users/4 HTTP/1.1  
Host: the-website.com
```

```
HTTP/1.1 204 No Content
```

Id	Nombre
1	Ernesto
2	Susana
3	Luis

Usuarios

EJEMPLO REST

Un server con información acerca de los usuarios

- El método PATCH se usa para actualizar parte de un recurso.

```
PATCH /users/1 HTTP/1.1
Host: the-website.com
Content-Type: application/json
Content-Length: 37
```

```
<user>
  <name>Amanda</name>
</user>
```

```
HTTP/1.1 204 No Content
```

Id	Nombre
1	Ernesto
2	Susana
3	Luis

Usuarios

EJEMPLO REST

Un server con información acerca de los usuarios

- ¿Qué pasa si algo sale mal?
 - Use los códigos de status HTTP para indicar éxito/fallo.

```
GET /users/999 HTTP/1.1  
Host: the-website.com  
Accept: application/json
```

```
HTTP/1.1 404 Not Found
```

- Más códigos de status:
 - <http://www.restapitutorial.com/httpstatuscodes.html>
- Opcionalmente se pueden incluir los mensajes de error en el cuerpo del response.

Id	Nombre
1	Ernesto
2	Susana
3	Luis

Usuarios



DISEÑANDO UNA API REST