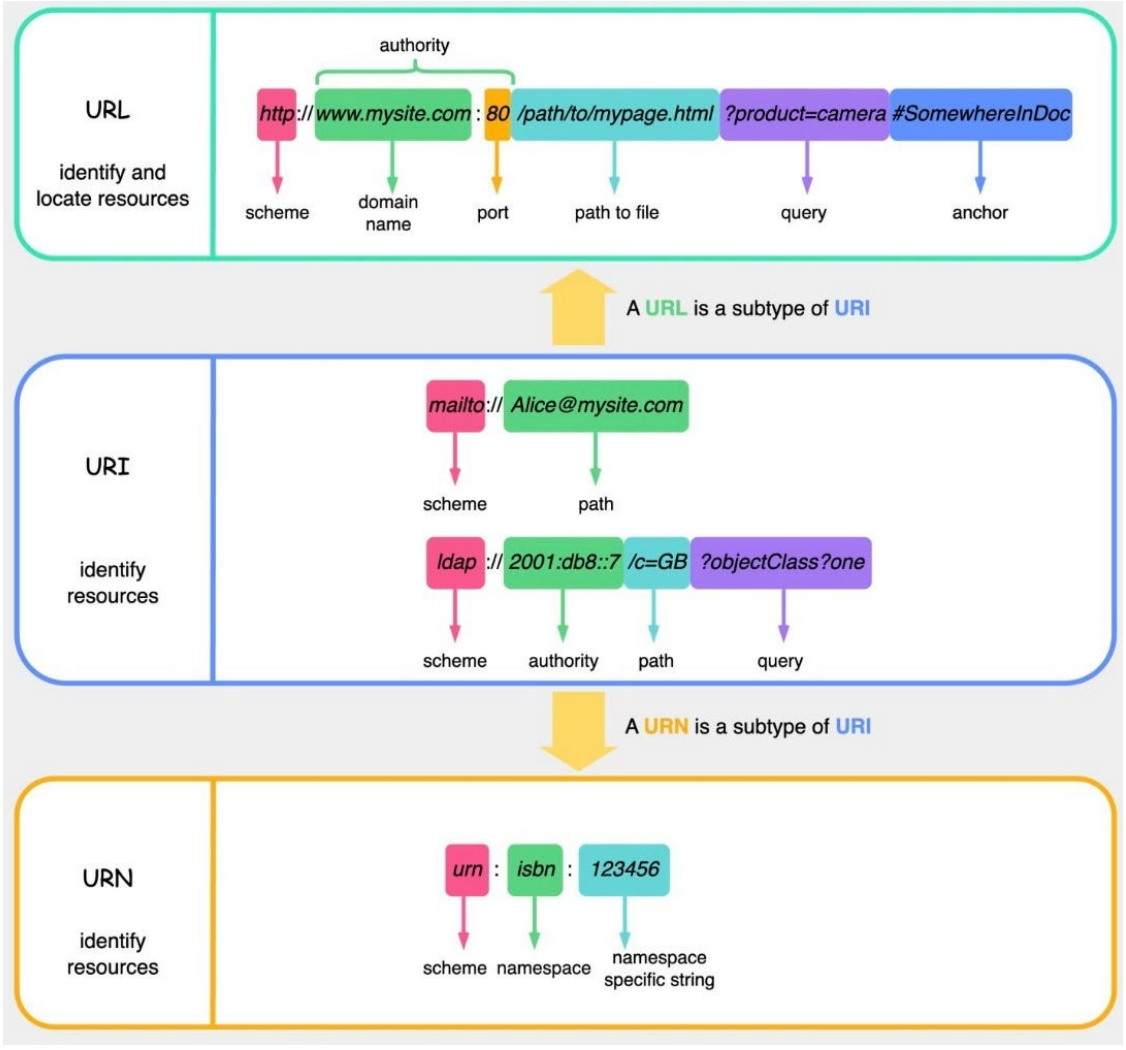
Protocolo HTTP

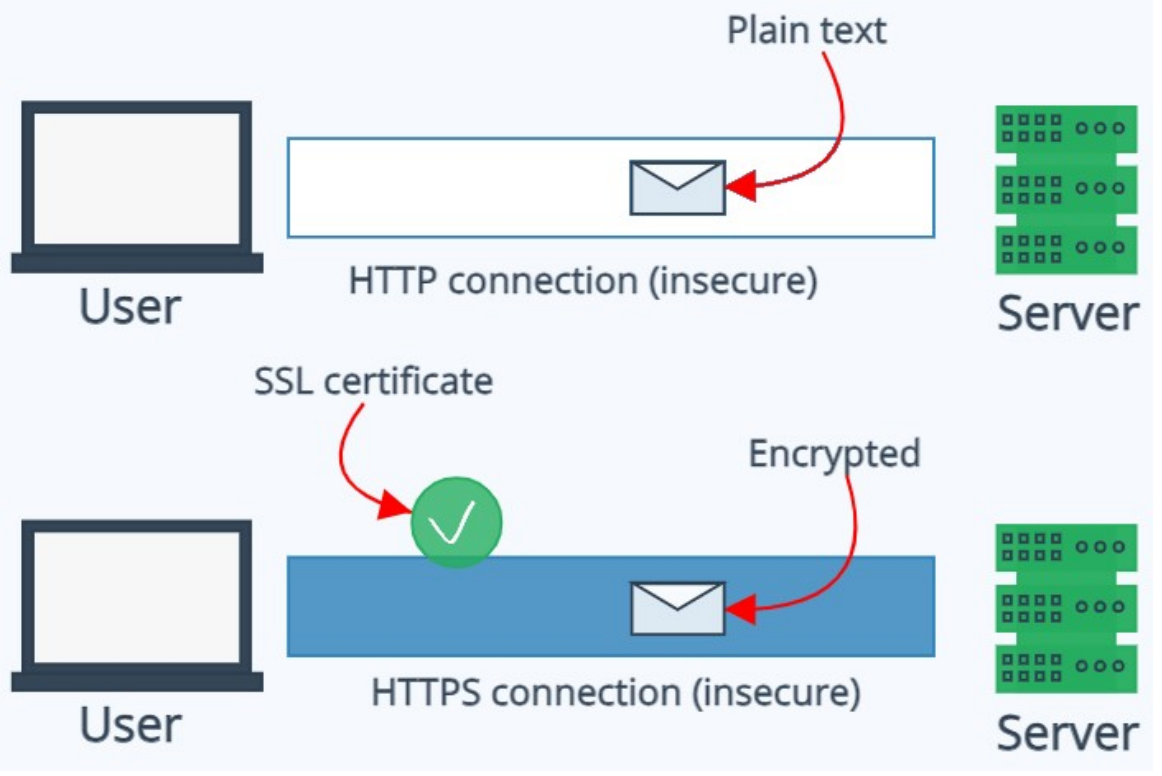
URL – Dominio
(Uniform Resource L

1. Protocolo
2. Subdominio
3. Dominio
4. TLD (Top Level Domain) o extension
5. Subcarpeta o directorio
6. Archivo

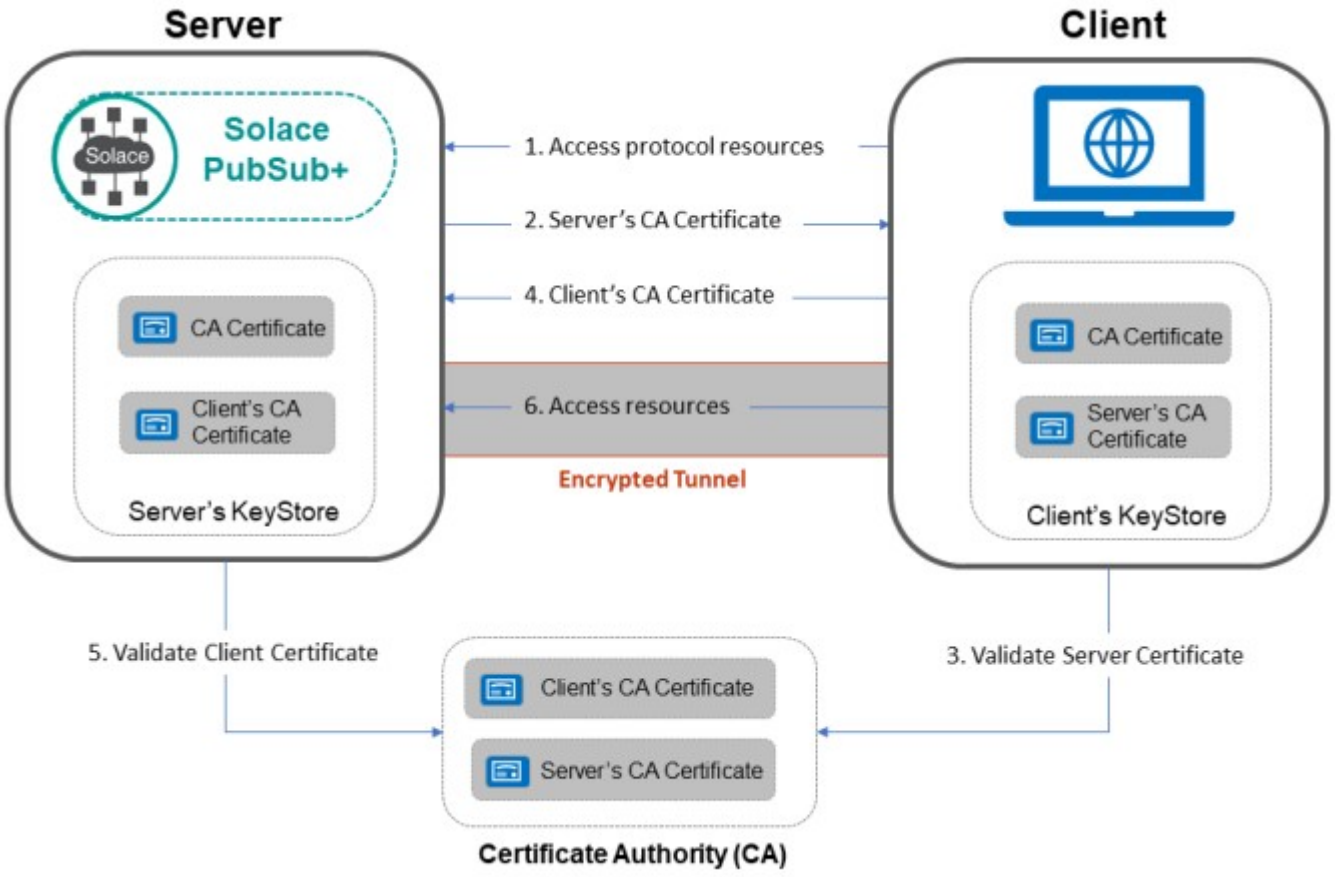

URL
identify and locate resources
authority
http://
www.mysite.com
:
80
/path/to/mypage.html
?product=camera
#SomewhereInDoc
scheme
domain name
port
path to file
query
anchor
A URL is a subtype of URI
URI
identify resources
mailto://
Alice@mysite.com
scheme
path
ldap
://
2001:db8::7
/c=GB
?objectClass?one
scheme
authority
path
query
A URN is a subtype of URI
URN
identify resources
urn
:
isbn
:
123456
scheme
namespace
namespace specific string

Protocolo HTTP

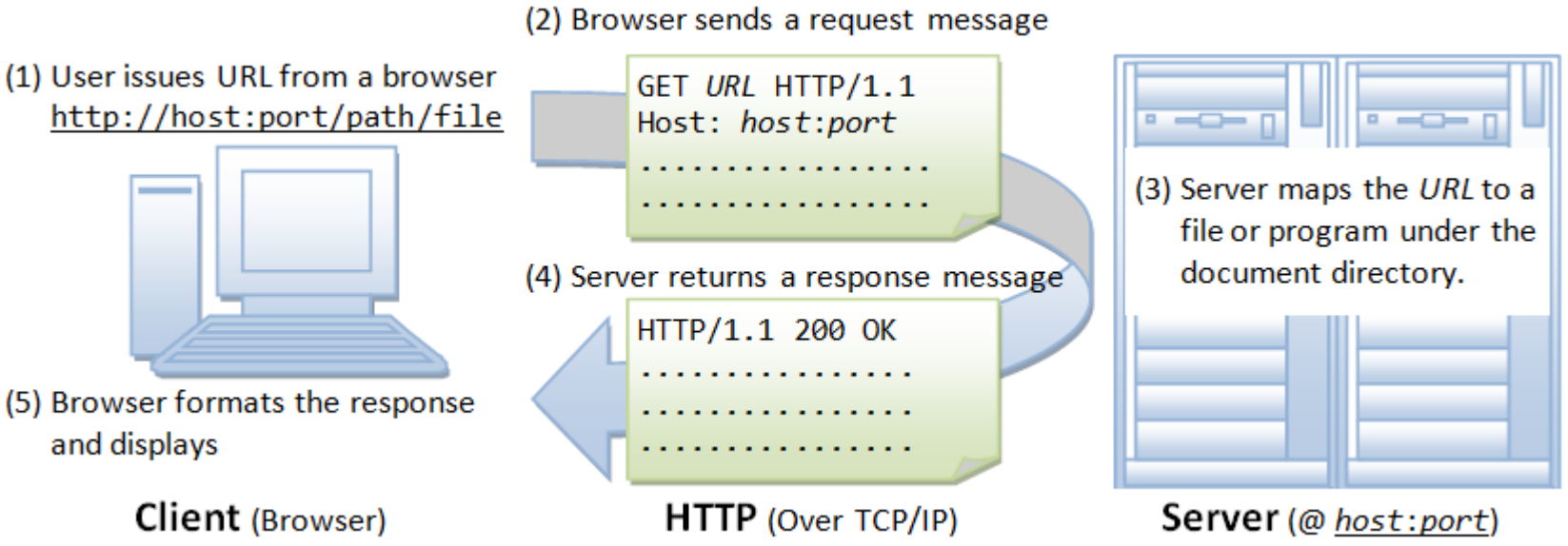
HTTP vs HTTPS



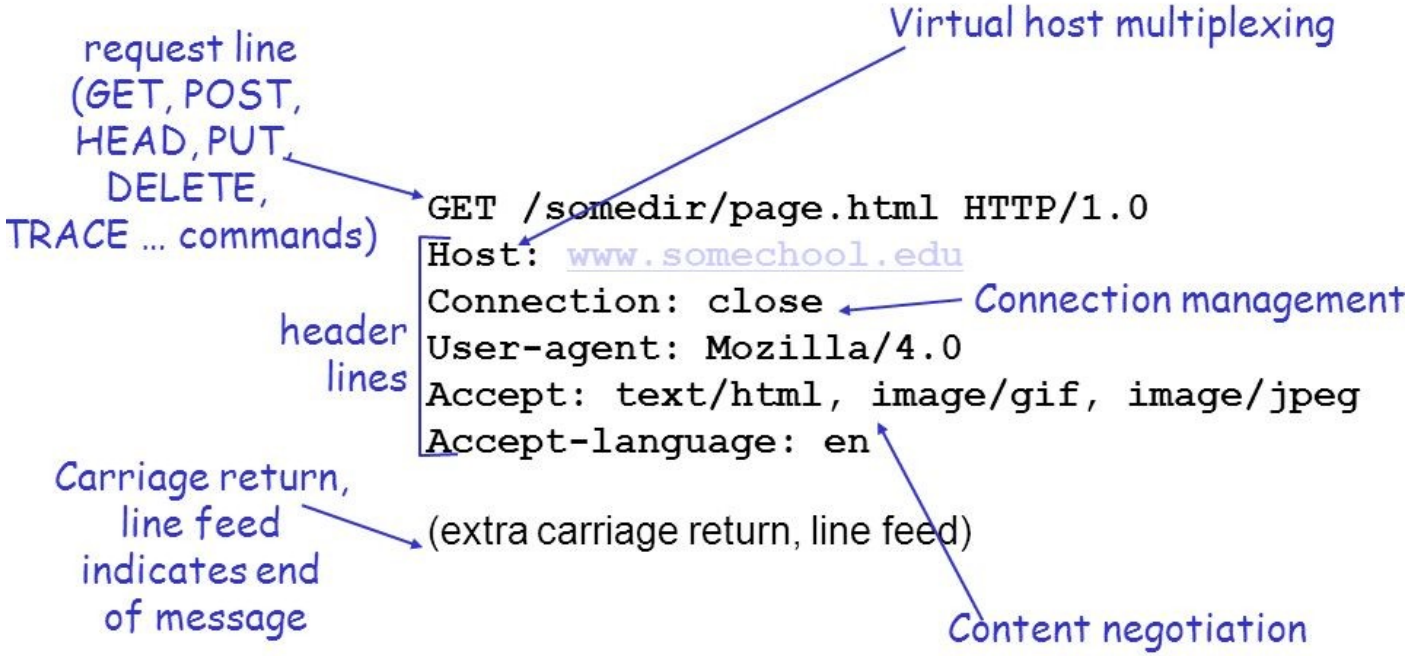
SSL


Server
Client
Solace
Solace PubSub+
1. Access protocol resources
2. Server's CA Certificate
4. Client's CA Certificate
6. Access resources
Encrypted Tunnel
CA Certificate
Client's CA Certificate
Server's KeyStore
CA Certificate
Server's CA Certificate
Client's KeyStore
5. Validate Client Certificate
3. Validate Server Certificate
Client's CA Certificate
Server's CA Certificate
Certificate Authority (CA)

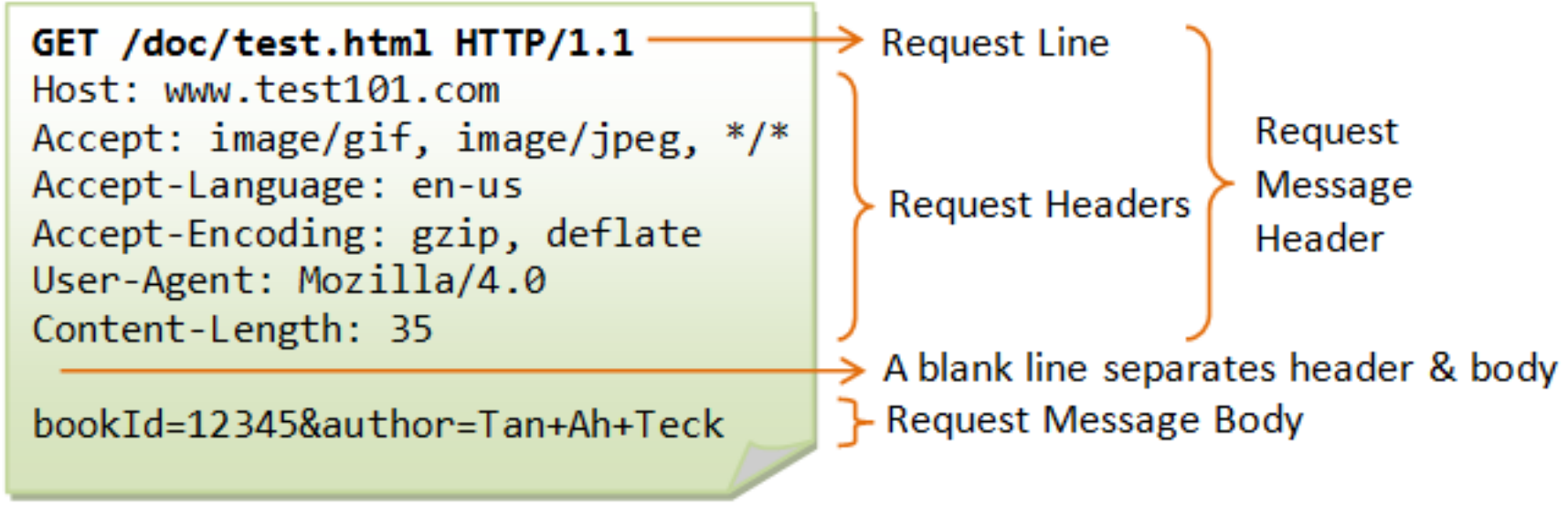
Request / Response



REQUEST


request line (GET, POST, HEAD, PUT, DELETE, TRACE ... commands)
Virtual host multiplexing
GET /somedir/page.html HTTP/1.0
Host: www.somechool.edu
Connection: close
Connection management
header lines
User-agent: Mozilla/4.0
Accept: text/html, image/gif, image/jpeg
Accept-language: en
Carriage return, line feed indicates end of message
(extra carriage return, line feed)
Content negotiation

Request GET


GET /doc/test.html HTTP/1.1
Host: www.test101.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0
Content-Length: 35
bookId=12345&author=Tan+Ah+Teck
Request Line
Request Headers
Request Message Header
A blank line separates header & body
Request Message Body

Request GET

- HTTP request message:
  - ASCII (human-readable format)

Request POST

```
POST /?id=1 HTTP/1.1
```

Request line

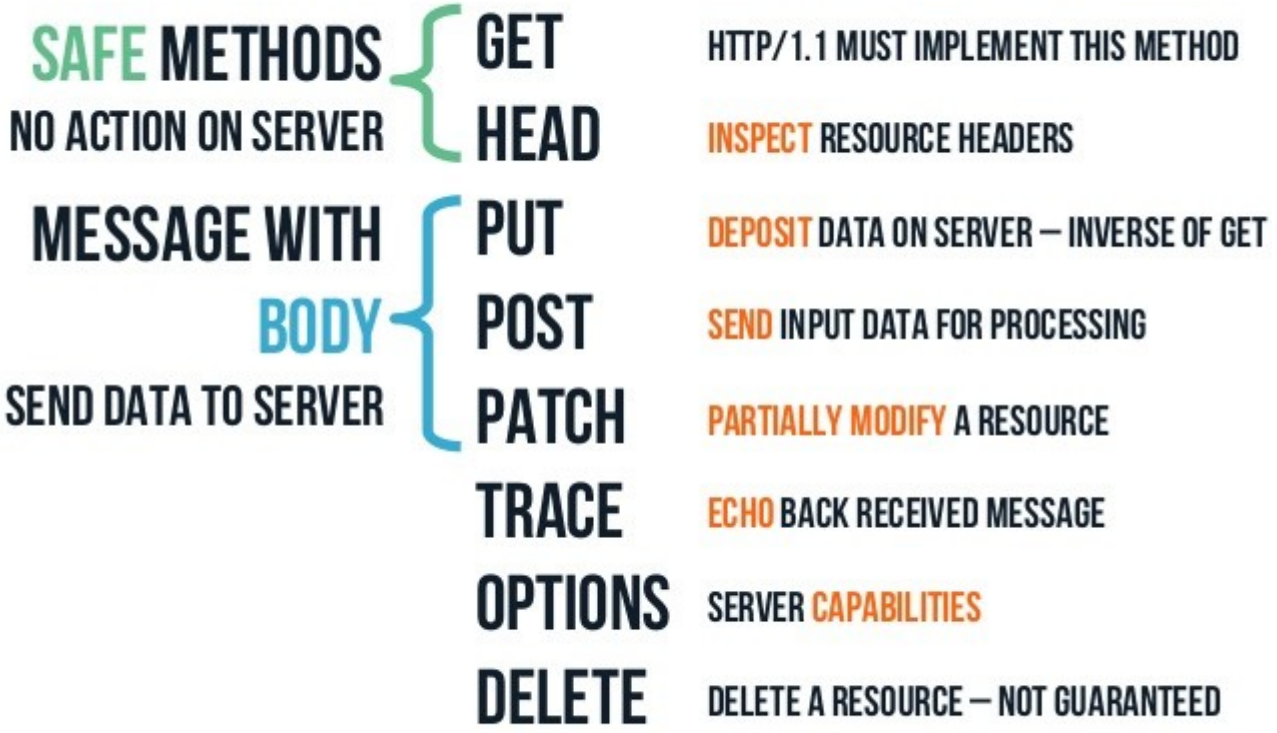
```
Host: www.swingvy.com
Content-Type: application/json; charset=utf-8
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.12; rv:53.0)
Gecko/20100101 Firefox/53.0
Connection: close
Content-Length: 136
```

Header

```
{
  "status": "ok",
  "extended": true,
  "results": [
    {"value": 0, "type": "int64"},
    {"value": 1.0e+3, "type": "decimal"}
  ]
}
```

Body message

Tipo de peticiones HTTP



Uso de peticiones HTTP

- **GET**: recupera datos del servidor
- **POST**: para crear un recurso
- PUT: reemplaza (modifica) un recurso
- DELETE: elimina un recurso

Características GET

HTTP GET

- **Purpose**: Retrieve data.

Characteristics:

- **Cacheable**: GET responses can be cached, making subsequent requests faster.
- **Browsable**: GET requests are visible in the browser history and can be bookmarked.
- **Idempotent**: Multiple identical GET requests have the same effect as a single request.
- **Safe**: Safe as they do not alter the server's state.
- **Length Restrictions**: The length of the URL (including parameters) may be limited by the browser or server.
- **Example Usage**:

Fetching a specific post:

```
GET /api/v1/posts/{id}
```

Características POST

HTTP POST

- **Purpose**: Send data to the server to create a new resource.

Characteristics:

- **Non-Cacheable**: POST responses are not cached.
- **Not Bookmarkable**: POST requests cannot be bookmarked.
- **Non-Idempotent**: Each POST request creates a new resource, potentially with different outcomes.
- **No Length Restrictions**: The body of the POST request can be large and includes data to be sent to the server.
- **Example Usage**:

Creating a new post:

```
POST /api/v1/post
```

Características PUT

HTTP PUT

- **Purpose**: Update or replace an existing resource.

Characteristics:

- **Idempotent**: Multiple identical PUT requests result in the same resource state.
- **Not Bookmarkable**: PUT requests cannot be bookmarked.
- **Non-Cacheable**: Though idempotent, responses to PUT requests are typically not cached because they alter the state of resources.
- **Example Usage**:

Updating a specific post:

```
PUT /api/v1/posts/{id}
```

Características DELETE

HTTP DELETE

- **Purpose**: Remove a resource from the server.

Characteristics:

- **Idempotent**: Multiple DELETE requests for the same resource will return the same result, effectively deleting it.
- **Not Cacheable**: DELETE responses are not cached.
- **Potential Response Data**: The server might return some data or an acknowledgment while deleting.
- **Example Usage**:

Deleting a specific post:

```
DELETE /api/v1/posts/{id}
```