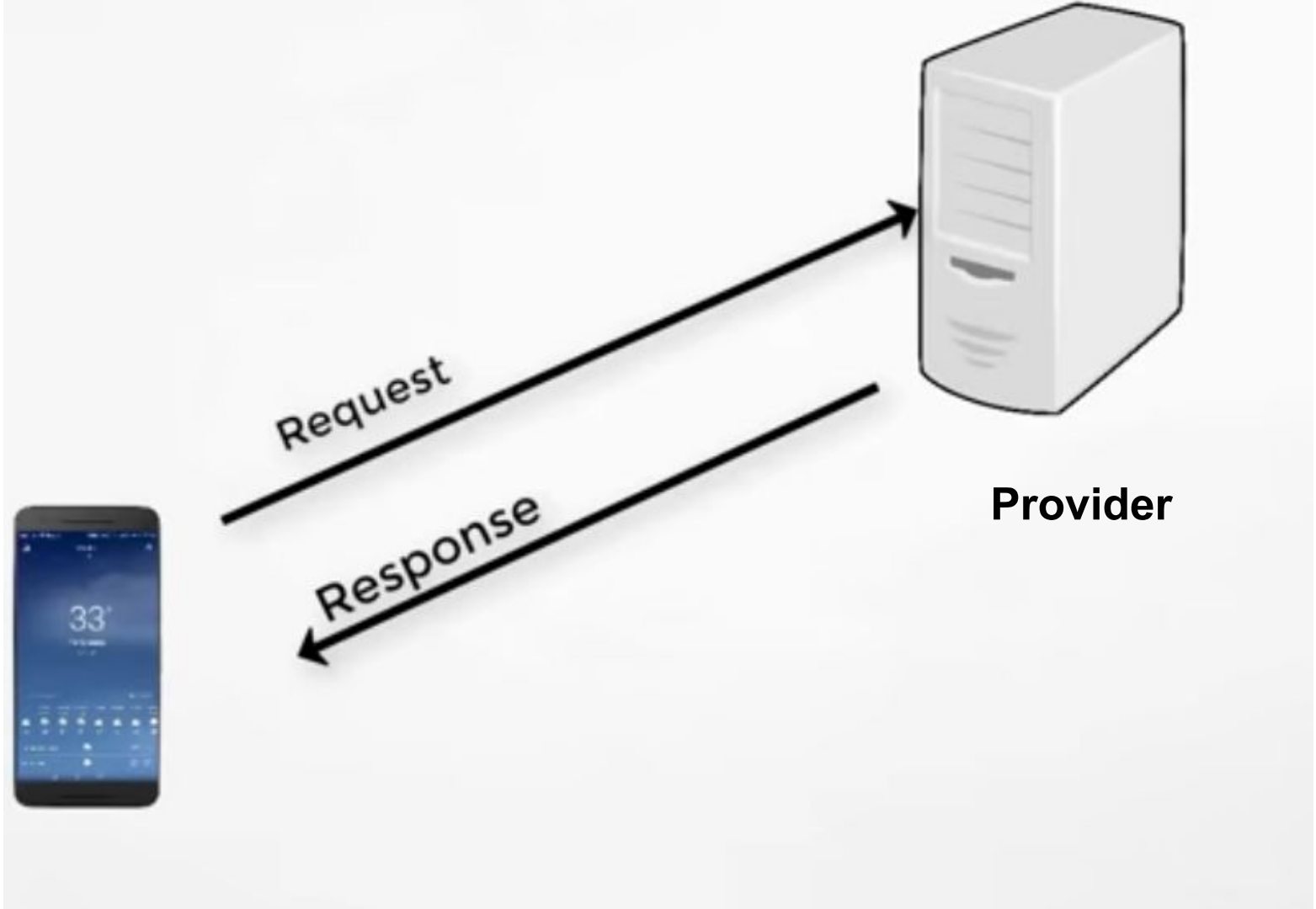
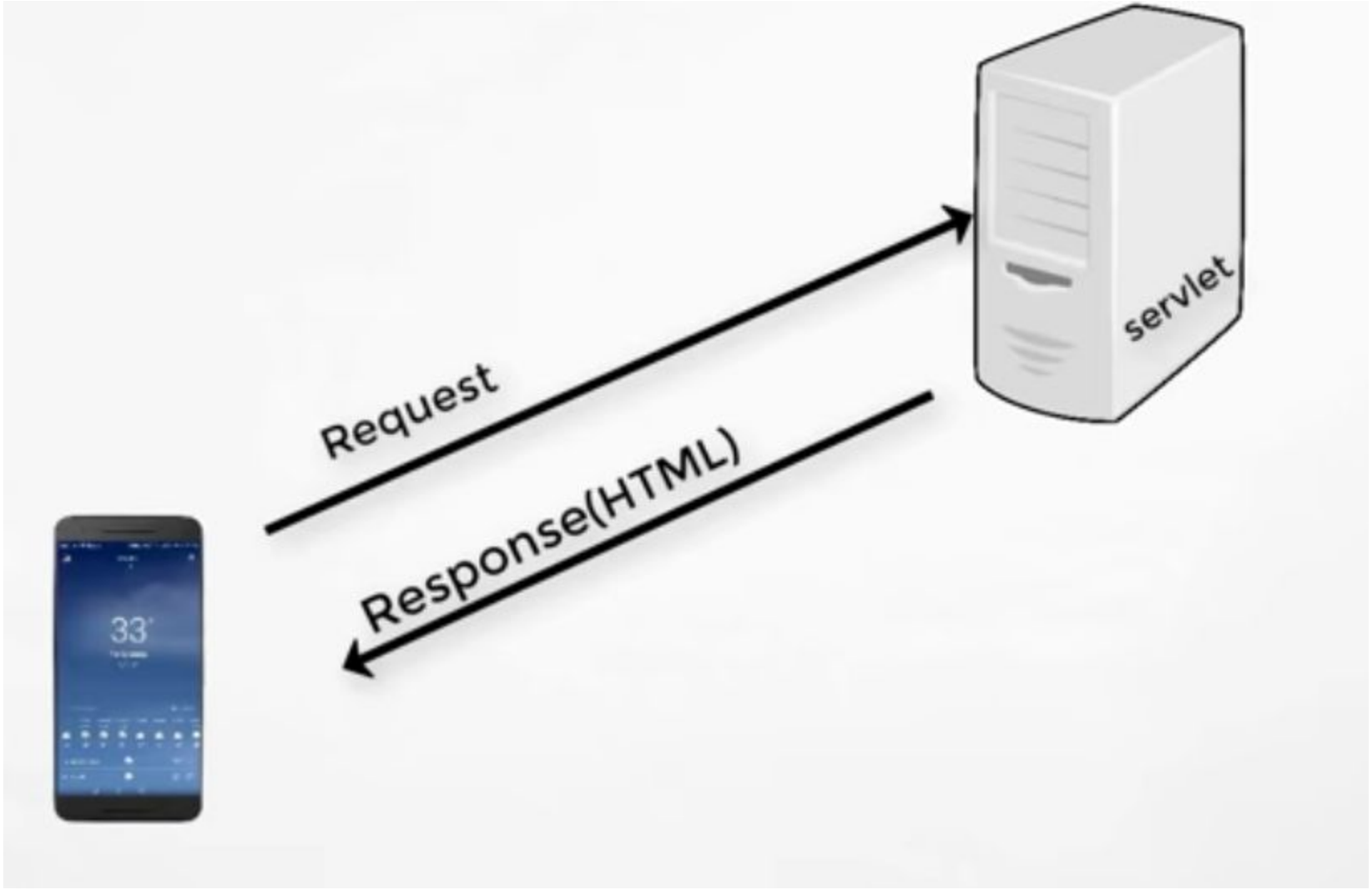
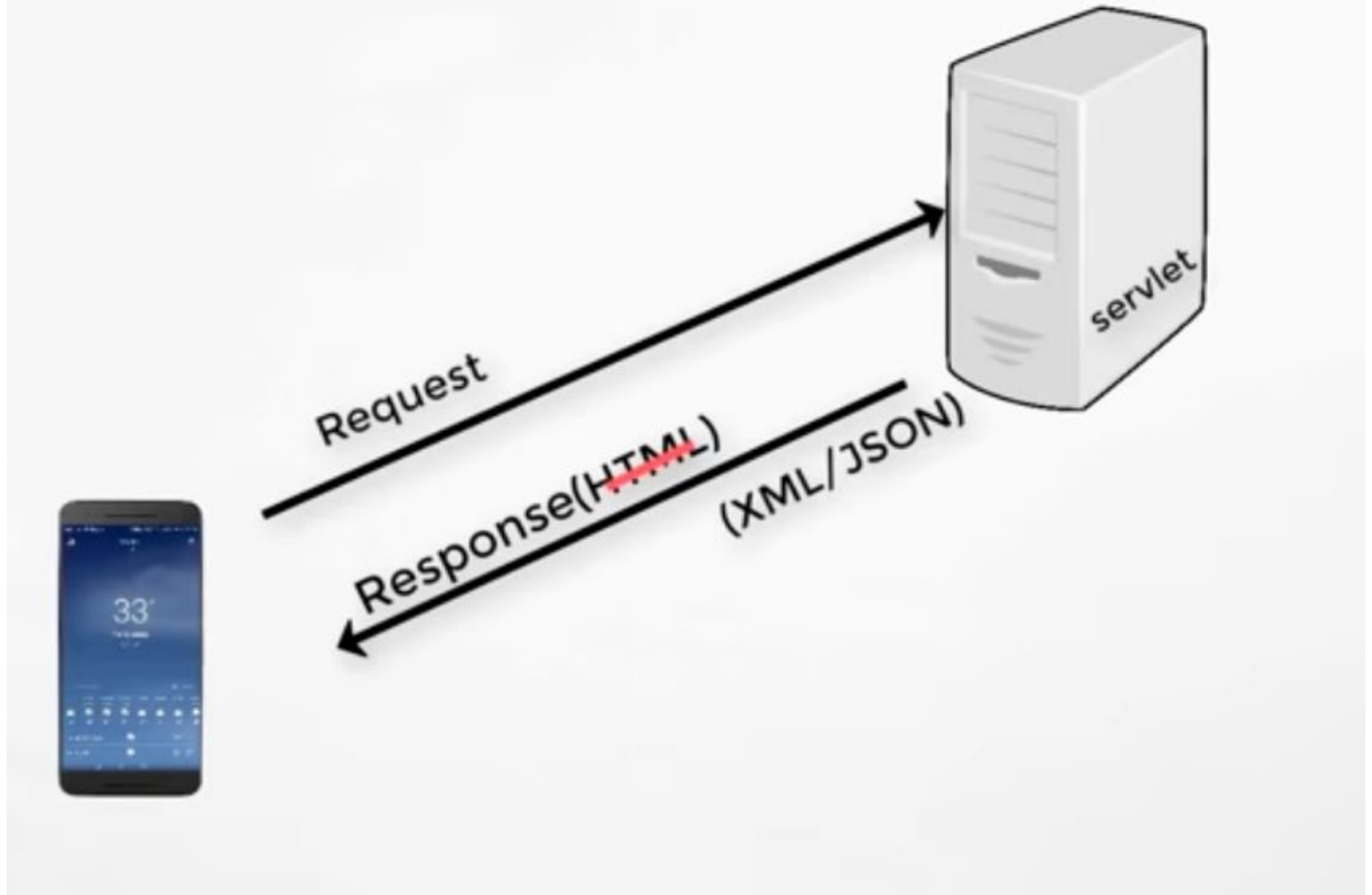
RESTFul Web Services

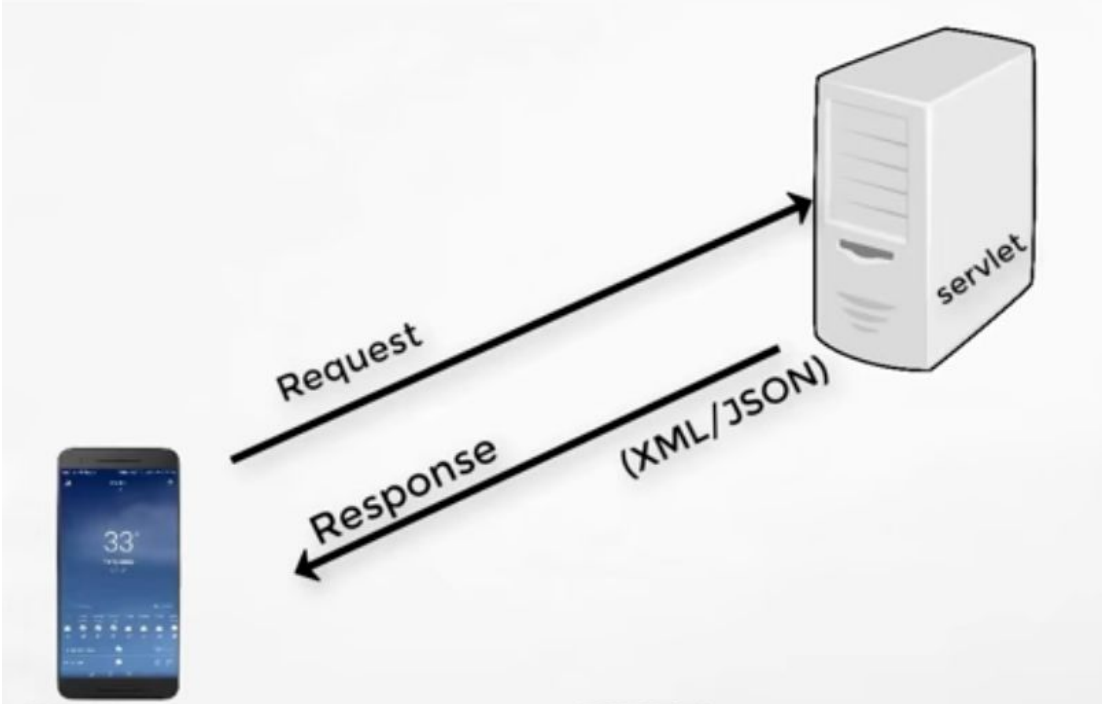
In IoT, there are 2 communication APIs –

- REST Based Communication APIs
- Web Socket Based Communication APIs


Request
Response
Provider


servlet
Request
Response(HTML)
33


servlet
Request
Response(HTML)
(XML/JSON)
33

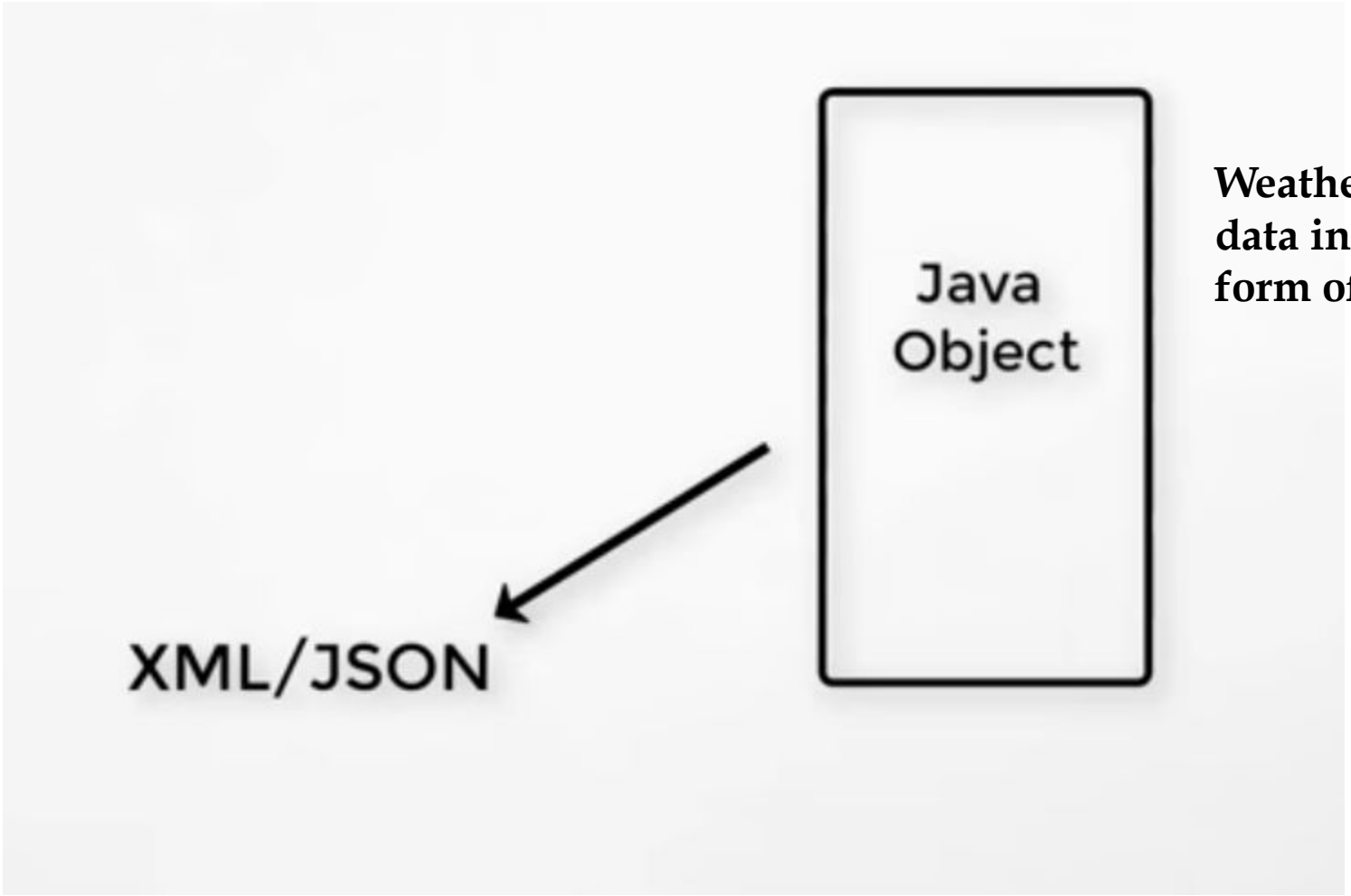


XML

```
<country>
<city>
   <temp>33</temp>
   <humidity>40</humidity>
</city>
</country>
```

JSON

```
{"country";{
        "city";{
         "temp":33,
         "humidity":40
              }
          }
```



Weather report data in the form of object

http://www.abc.com/questions?subject=java

http://www.abc.com/questions/java

Actually you are accessing resource. Instead of thinking there is a server which does a processing, just think as you are fetching data.

What is REST Architecture?

- REST stands for REpresentational State Transfer. REST is web standards based architecture and uses HTTP Protocol.
- It revolves around resource where every component is a resource and a resource is accessed by a common interface using HTTP standard methods.
- REST was first introduced by Roy Fielding in 2000.
- In REST architecture, a REST Server simply provides access to resources and REST client accesses and modifies the resources.
- Here each resource is identified by URIs/ global IDs.
- REST uses various representation to represent a resource like text, JSON, XML. JSON is the most popular one.
- **REST is a set of architectural principles by which you can design web services and web APIs that focus on a system resources**

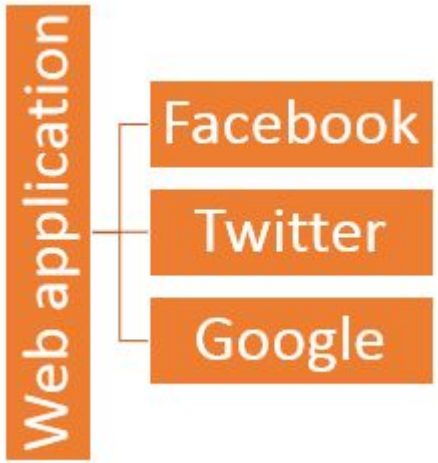
Introduction to RESTFul web services

- A web service is a collection of open protocols and standards used for exchanging data between applications or systems.
- Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer.
- Web services based on REST Architecture are known as RESTful web services. These web services uses HTTP methods to implement the concept of REST architecture.
- A RESTful web service usually defines a URI, Uniform Resource Identifier a service, provides resource representation such as JSON and set of HTTP Methods.

Why Restful?

Heterogeneous languages and environments –

- It enables web applications that are built on various programming languages to communicate with each other
- With the help of Restful services, these web applications can reside on different environments, some could be on Windows, and others could be on Linux.



Facebook, Twitter, and Google expose their functionality in the form of Restful web services. This allows any client application to call these web services via REST.

Finally is the event of the Cloud –

Since all Cloud-based architectures work on the REST principle, it makes more sense for web services to be programmed on the REST services based architecture to make the best use of Cloud-based services.

Restful Architecture

1. State and functionality are divided into distributed resources – This means that every resource should be accessible via the normal HTTP commands of GET, POST, PUT, or DELETE.

2. The architecture is client/server, stateless, layered, and supports caching

- Client-server is the typical architecture where the server can be the web server hosting the application, and the client can be as simple as the web browser.
- Stateless means that the state of the application is not maintained in REST.

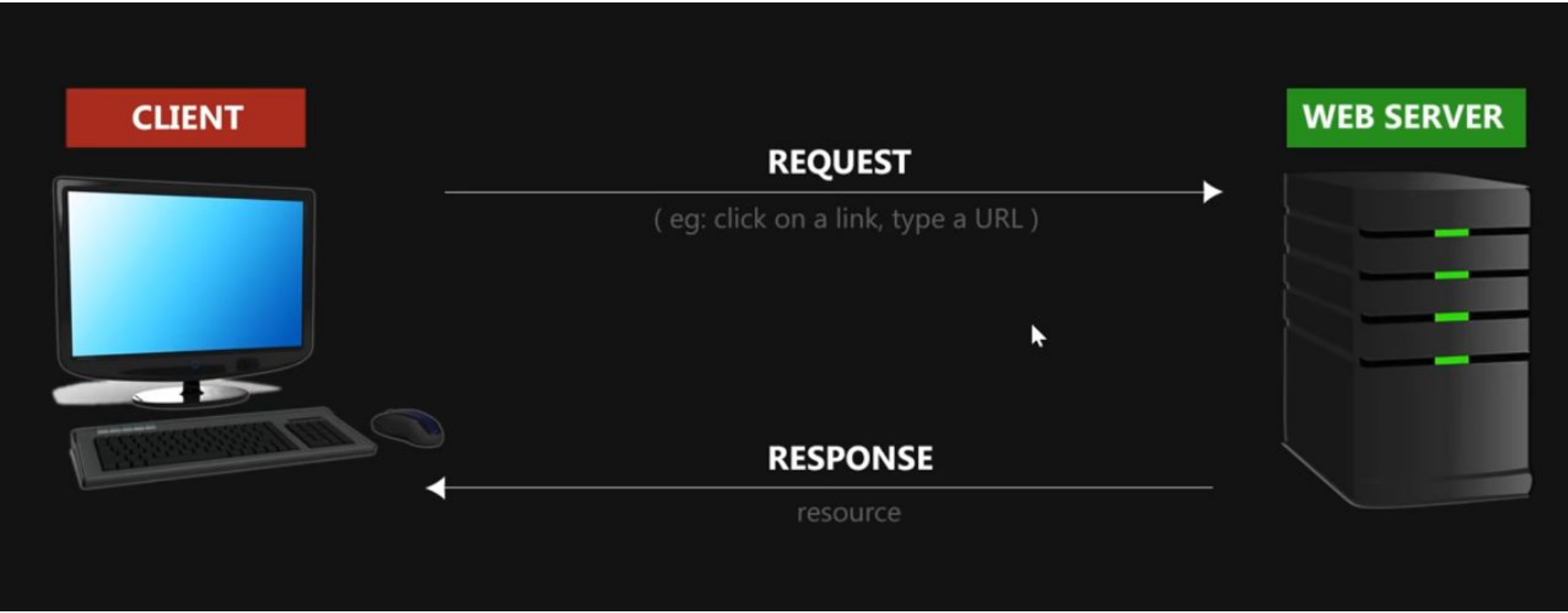
RESTFul Client-server

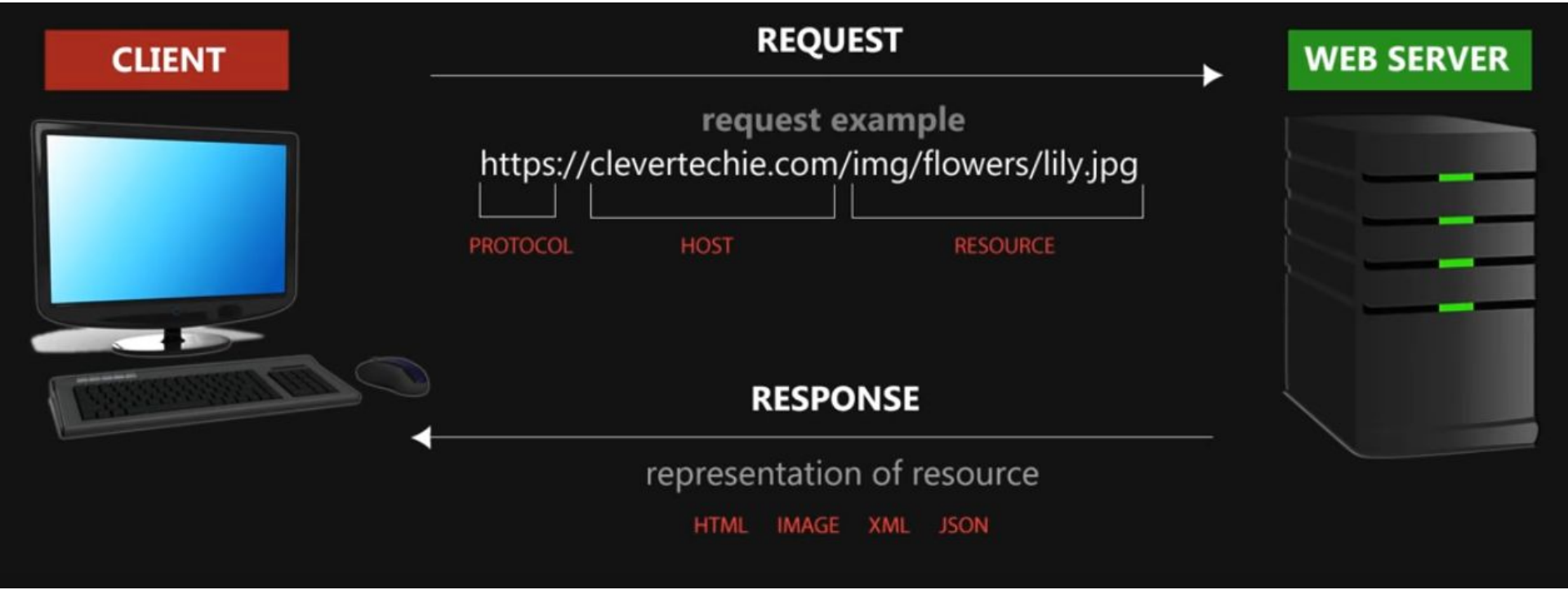
RESTFul web services

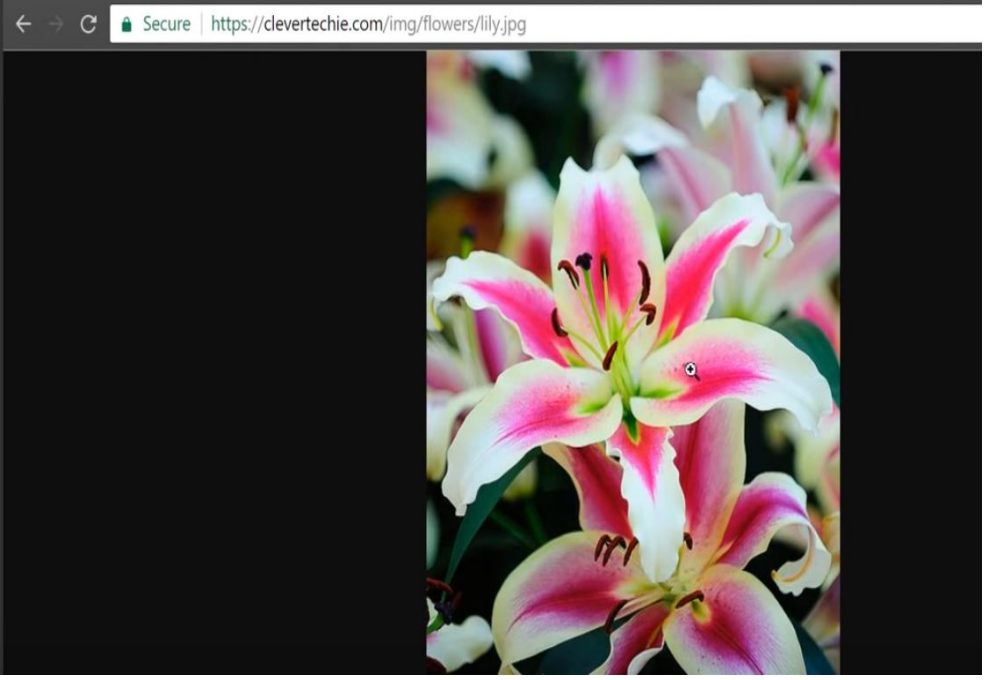
Stateless: The server should not maintain any sort of information between requests from the client. It's a very simple independent question-answer sequence. The client asks a question, the server answers it appropriately. The client will ask another question. The server will not remember the previous question-answer scenario and will need to answer the new question independently.

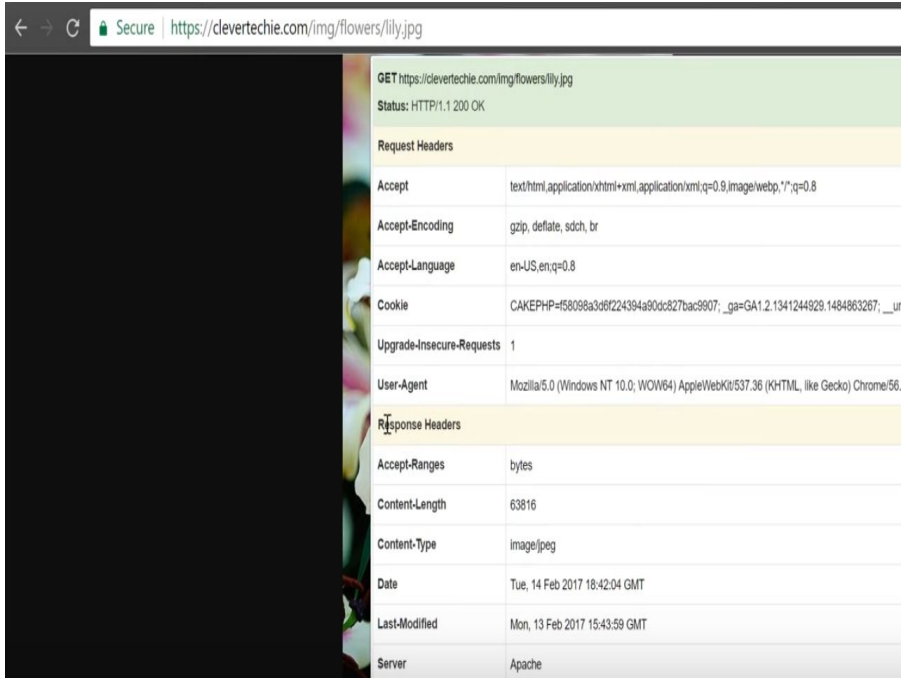
Cache: Sometimes the client might ask the server for the same request again. This is even though it had already asked for it in the past. This request will go to the server, and the server will give a response. This increases the traffic across the network. The cache is a concept implemented on the client to store requests which have already been sent to the server. So if the same request is given by the client, instead of going to the server, it would go to the cache and get the required information. This saves the amount of to and fro network traffic from the client to the server.

HTTP

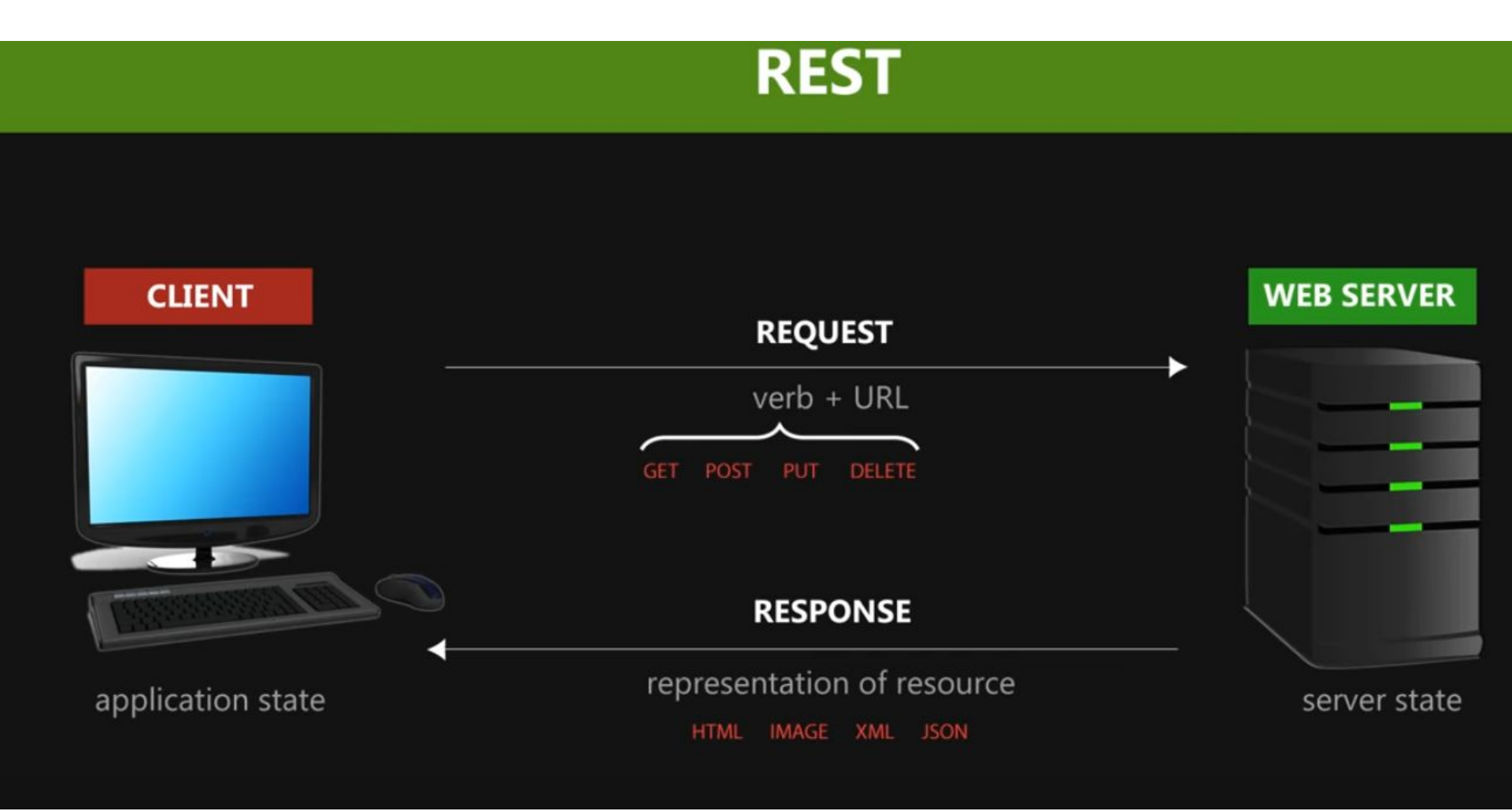

CLIENT
WEB SERVER
REQUEST
(eg: click on a link, type a URL)
RESPONSE
resource

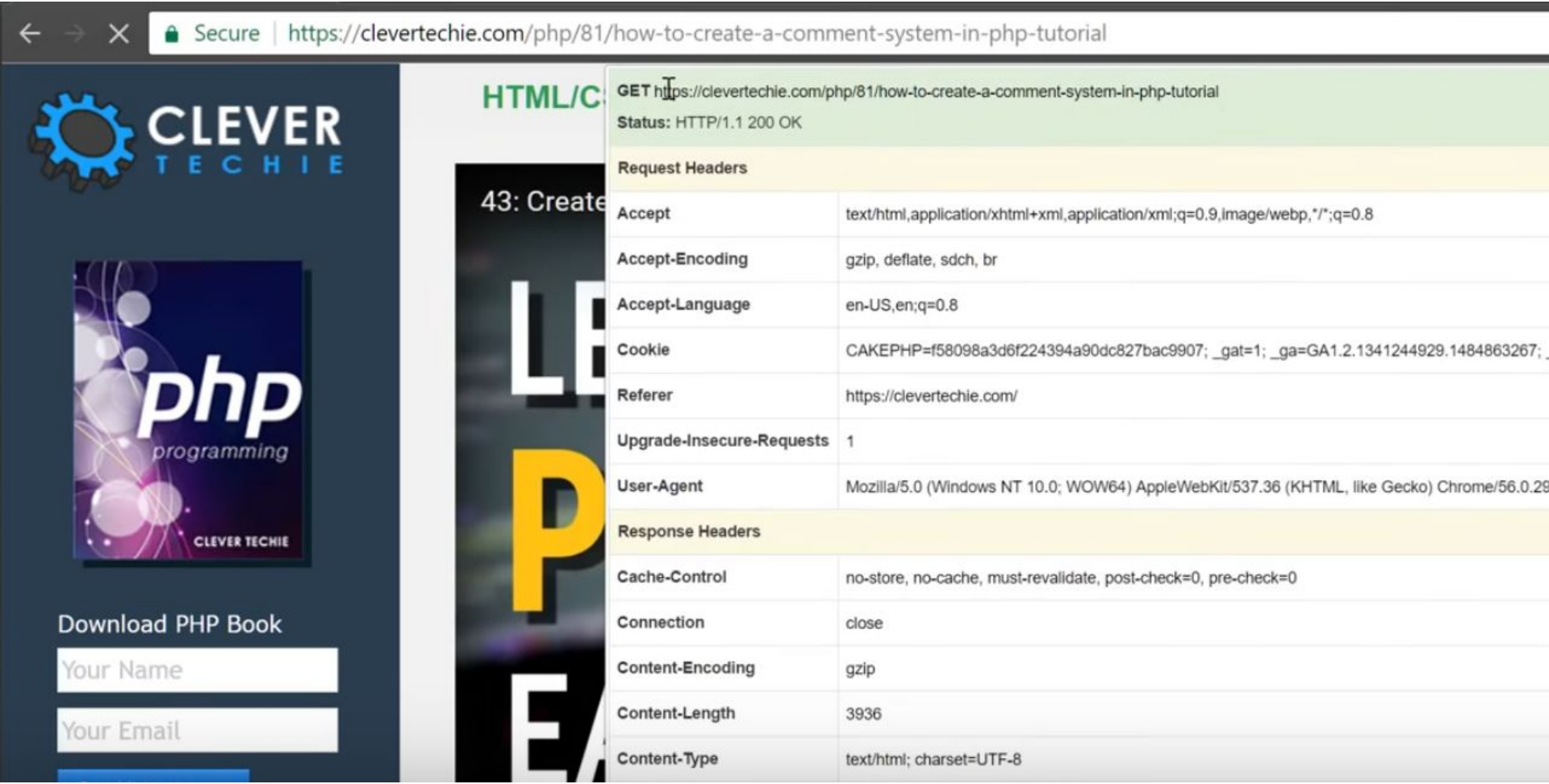

CLIENT
REQUEST
WEB SERVER
request example
https://clevertechie.com/img/flowers/lily.jpg
PROTOCOL
HOST
RESOURCE
RESPONSE
representation of resource
HTML
IMAGE
XML
JSON

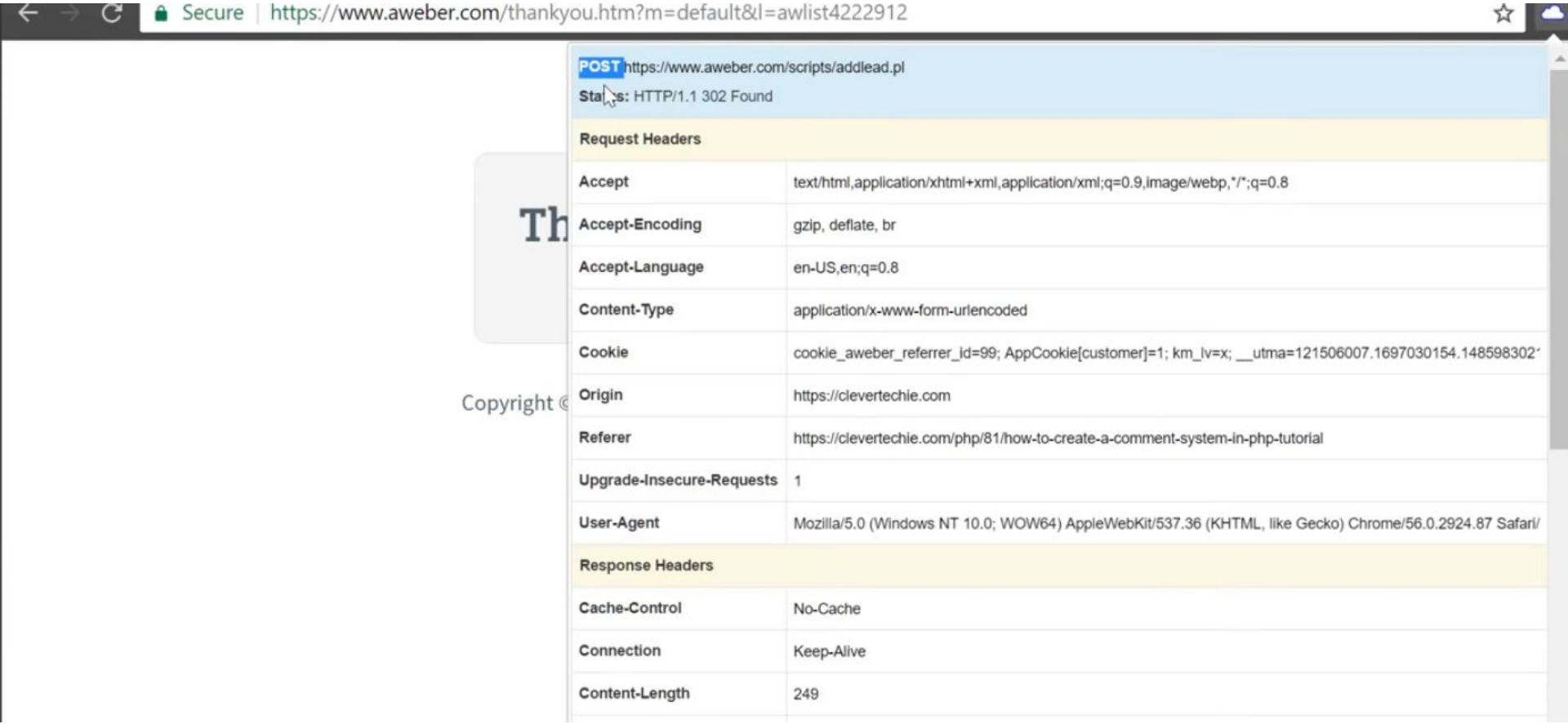

Secure
https://clevertechie.com/img/flowers/lily.jpg


Secure
https://clevertechie.com/img/flowers/lily.jpg
GET https://clevertechie.com/img/flowers/lily.jpg
Status: HTTP/1.1 200 OK
Request Headers
Accept
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding
gzip, deflate, sdch, br
Accept-Language
en-US,en;q=0.8
Cookie
CAKEPHP=f58098a3d6f224394a90dc827bac9907; _ga=GA1.2.1341244929.1484863267; __ut
Upgrade-Insecure-Requests
1
User-Agent
Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/56.
Response Headers
Accept-Ranges
bytes
Content-Length
63816
Content-Type
image/jpeg
Date
Tue, 14 Feb 2017 18:42:04 GMT
Last-Modified
Mon, 13 Feb 2017 15:43:59 GMT
Server
Apache

REST


CLIENT
WEB SERVER
REQUEST
verb + URL
GET POST PUT DELETE
RESPONSE
representation of resource
HTML IMAGE XML JSON
application state
server state


Secure | https://clevertechie.com/php/81/how-to-create-a-comment-system-in-php-tutorial
CLEVER TECHIE
HTML/C
43: Create
php programming
CLEVER TECHIE
Download PHP Book
Your Name
Your Email
GET https://clevertechie.com/php/81/how-to-create-a-comment-system-in-php-tutorial
Status: HTTP/1.1 200 OK
Request Headers
Accept
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding
gzip, deflate, sdch, br
Accept-Language
en-US,en;q=0.8
Cookie
CAKEPHP=f58098a3d6f224394a90dc827bac9907; _gat=1; _ga=GA1.2.1341244929.1484863267;
Referer
https://clevertechie.com/
Upgrade-Insecure-Requests
1
User-Agent
Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/56.0.29
Response Headers
Cache-Control
no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Connection
close
Content-Encoding
gzip
Content-Length
3936
Content-Type
text/html; charset=UTF-8


Secure | https://www.aweber.com/thankyou.htm?m=default&l=awlist4222912
POST https://www.aweber.com/scripts/addlead.pl
Status: HTTP/1.1 302 Found
Request Headers
Accept
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding
gzip, deflate, br
Accept-Language
en-US,en;q=0.8
Content-Type
application/x-www-form-urlencoded
Cookie
cookie_aweber_referrer_id=99; AppCookie[customer]=1; km_lv=x; __utma=121506007.1697030154.148598302
Origin
https://clevertechie.com
Referer
https://clevertechie.com/php/81/how-to-create-a-comment-system-in-php-tutorial
Upgrade-Insecure-Requests
1
User-Agent
Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/56.0.2924.87 Safari/
Response Headers
Cache-Control
No-Cache
Connection
Keep-Alive
Content-Length
249

REST API

CLIENT

application

REQUEST

REST API endpoint URL + API method + parameters

request example

https://api.flickr.com/services/rest/?method=flickr.photos.getinfo&photo_id=2079357948

REST API ENDPOINT URL | METHOD | PARAMETERS

RESPONSE

representation of resource

REST XML-RPC SOAP JSON Serialized PHP

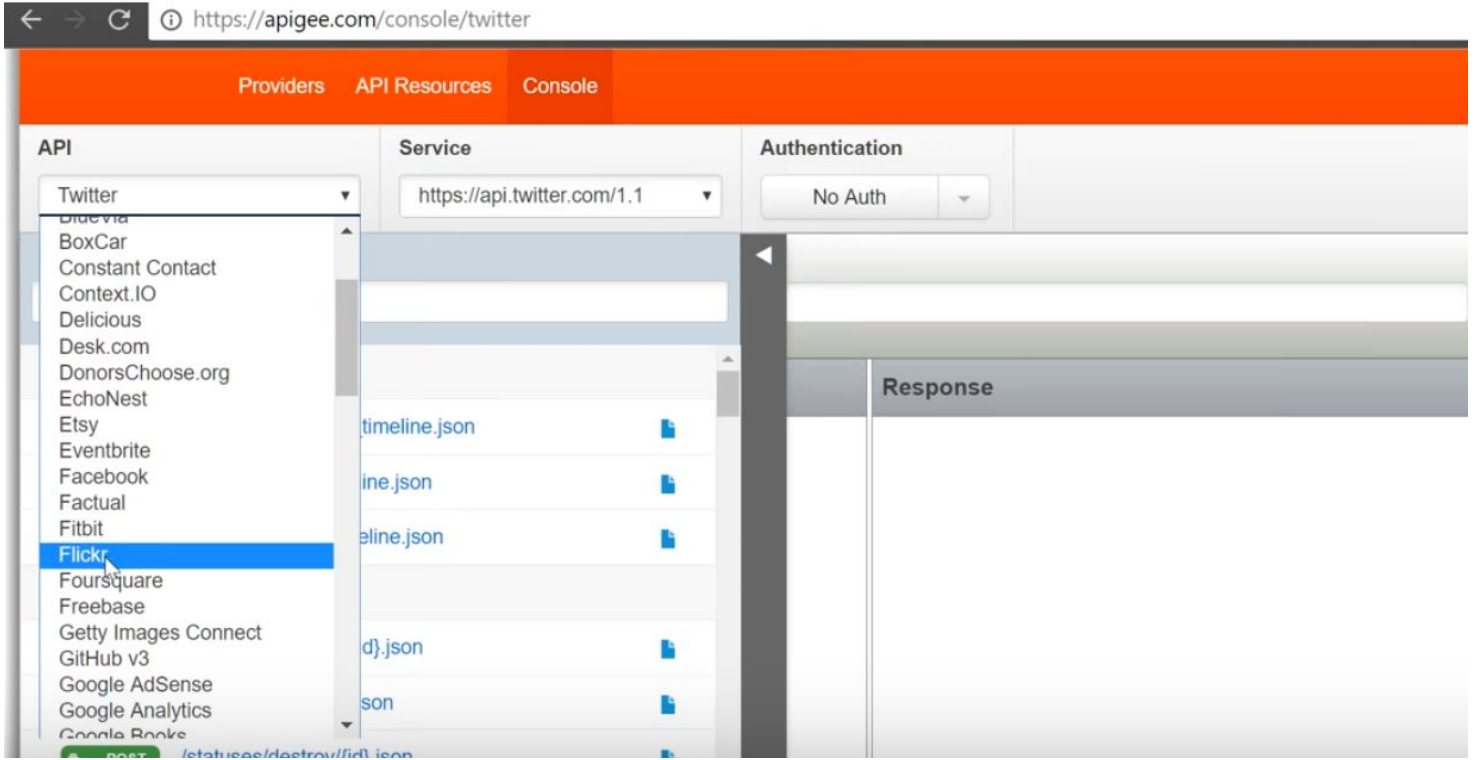
WEB SERVER

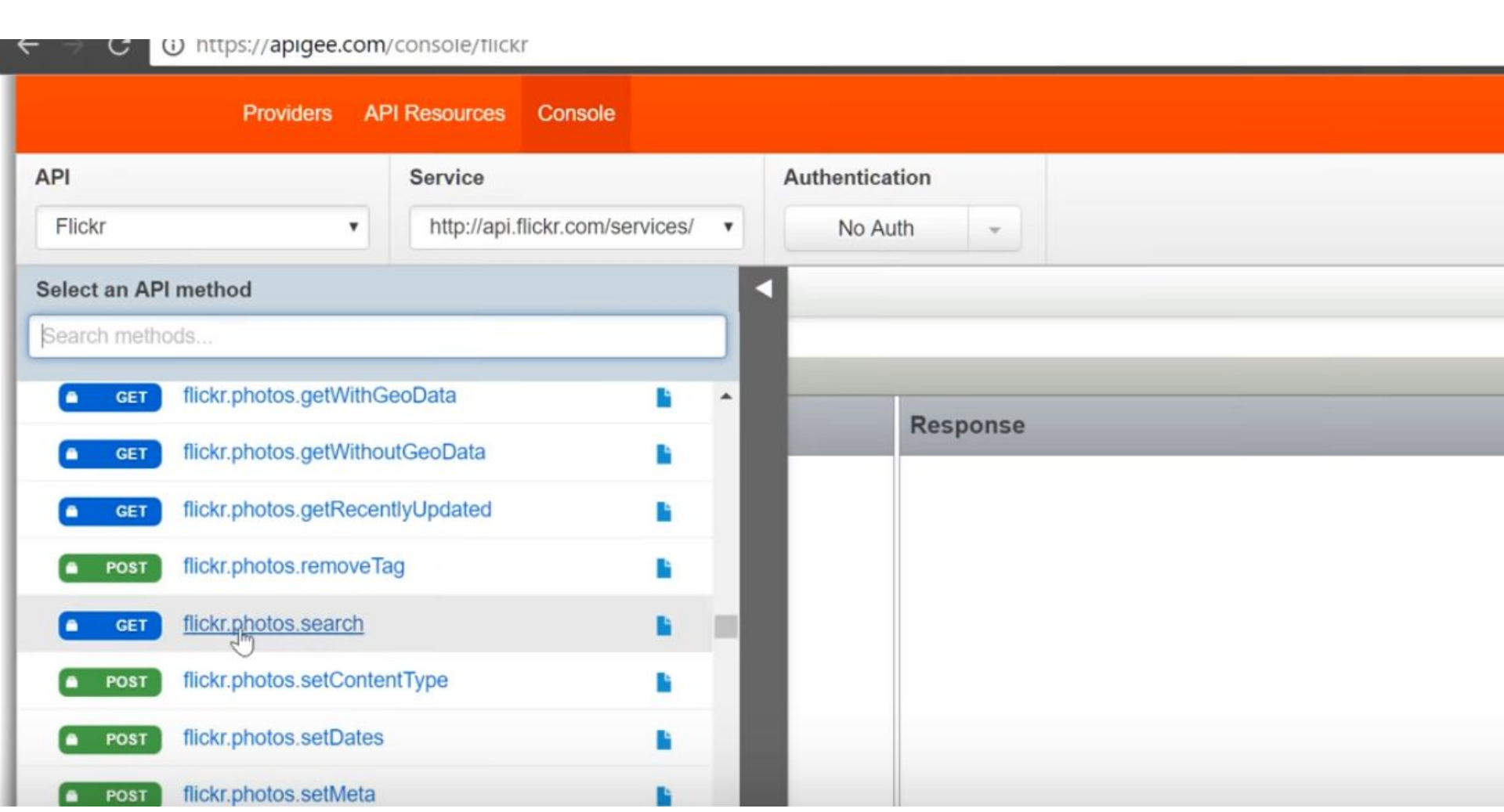
API

(useful methods)

representational state transfer application programming interface

https://www.flickr.com/services/api/




https://apigee.com/console/flickr
Providers
API Resources
Console
API
Flickr
Service
http://api.flickr.com/services/
Authentication
No Auth
Select an API method
Search methods...
GET flickr.photos.getWithGeoData
GET flickr.photos.getWithoutGeoData
GET flickr.photos.getRecentlyUpdated
POST flickr.photos.removeTag
GET flickr.photos.search
POST flickr.photos.setContentType
POST flickr.photos.setDates
POST flickr.photos.setMeta
Response

https://apigee.com/console/flickr

Providers | API Resources | Console | Switch to…

API: Flickr

Service: http://api.flickr.com/services/

Authentication: No Auth

Feedback

Request URL

GET | http://api.flickr.com/services/rest/?method=flickr.photos.search | Send

Query* | Template | Headers

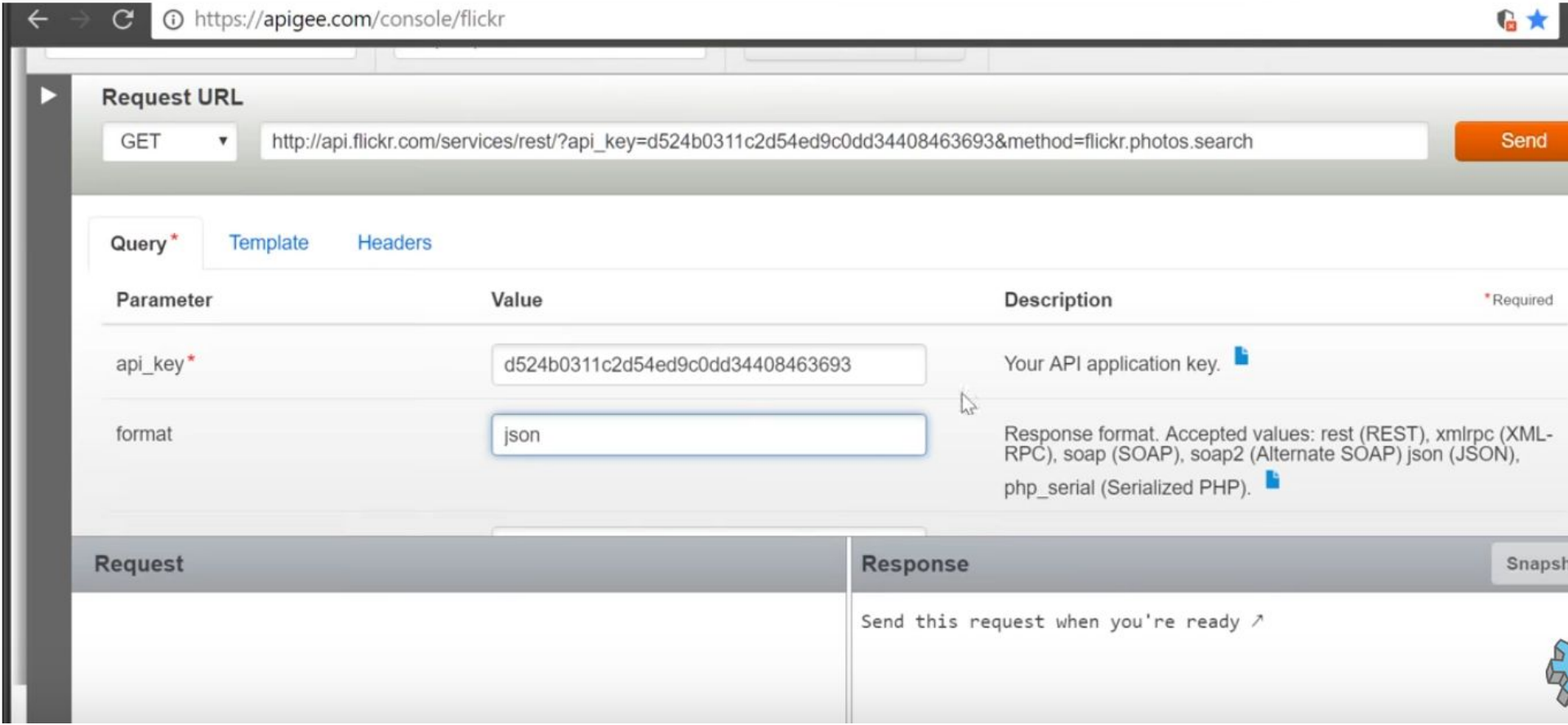
Parameter	Value	Description
api_key*		Your API application key.
format	rest (example)	Response format. Accepted values: rest (REST), xmlrpc (XML-RPC), soap (SOAP), soap2 (Alternate SOAP) json (JSON), php_serial (Serialized PHP).

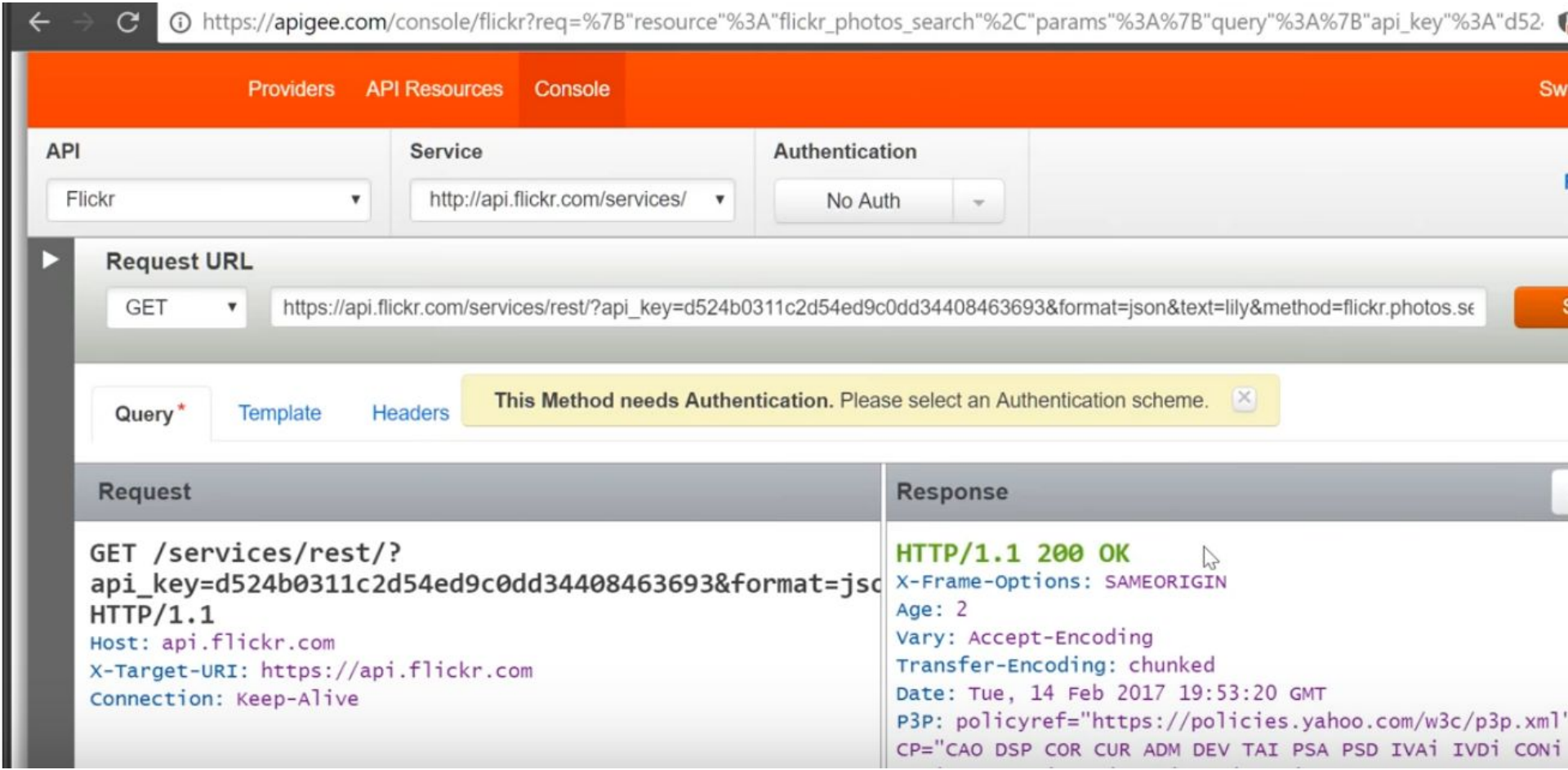
*Required

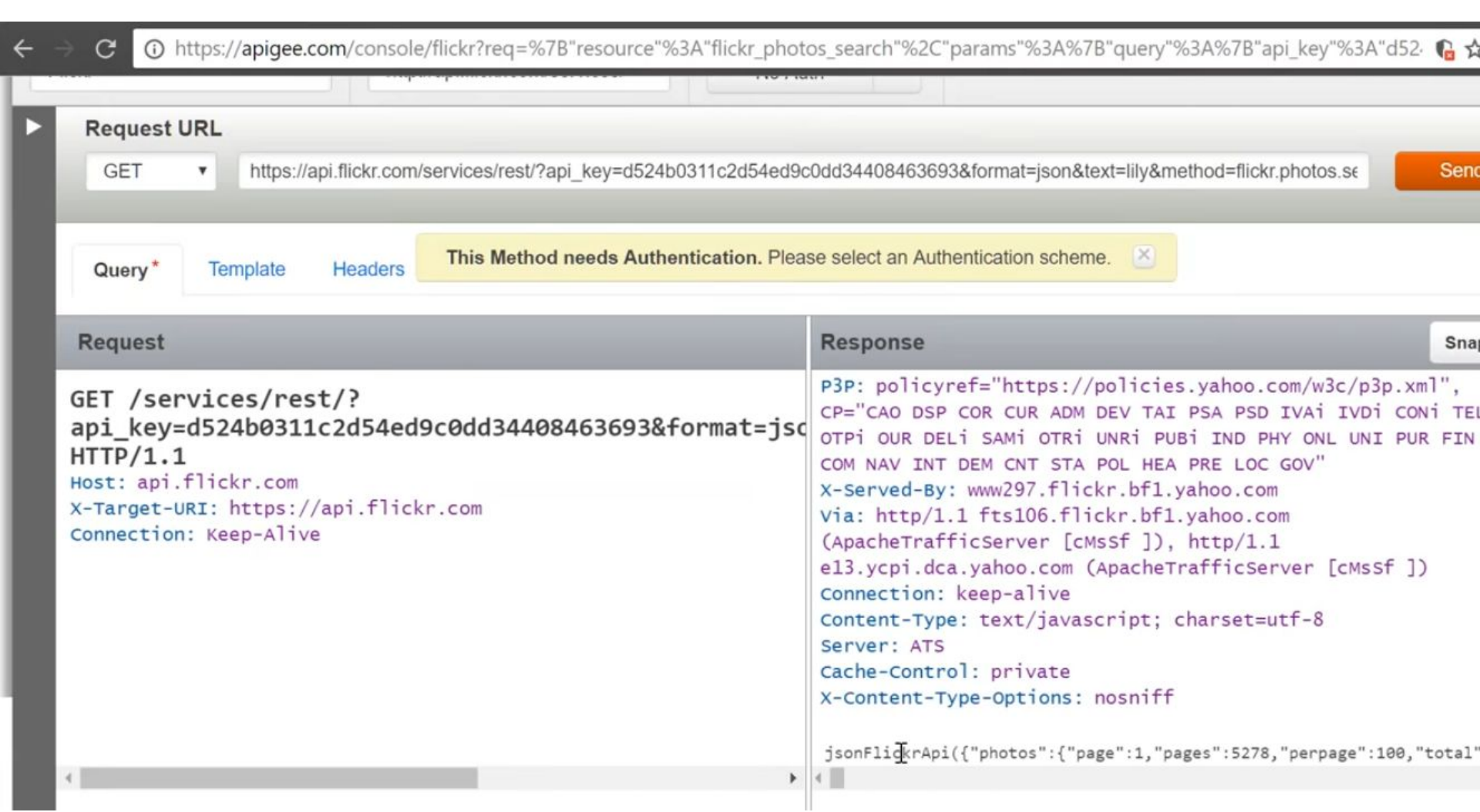
Request

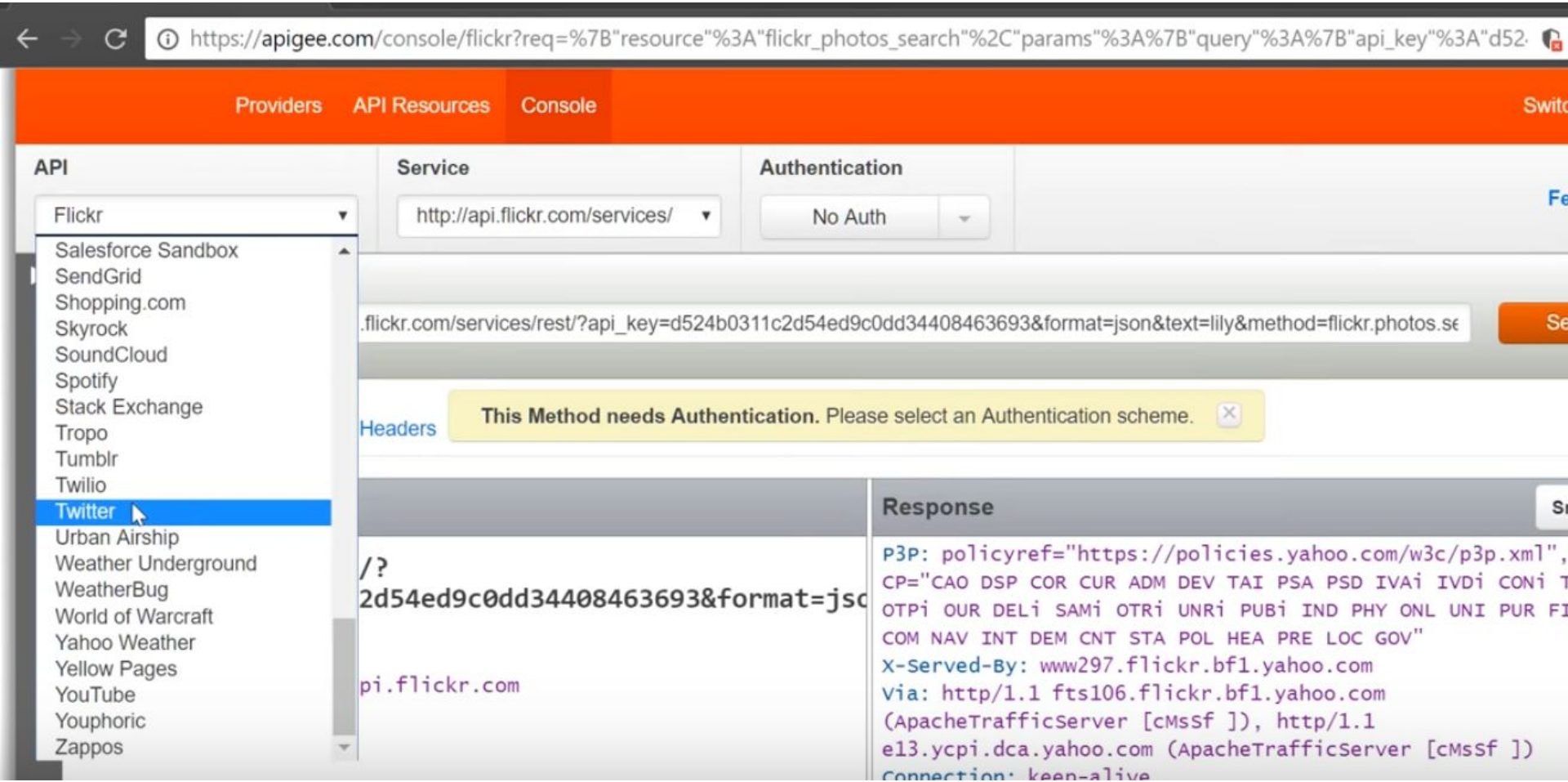
Response | Snapshot

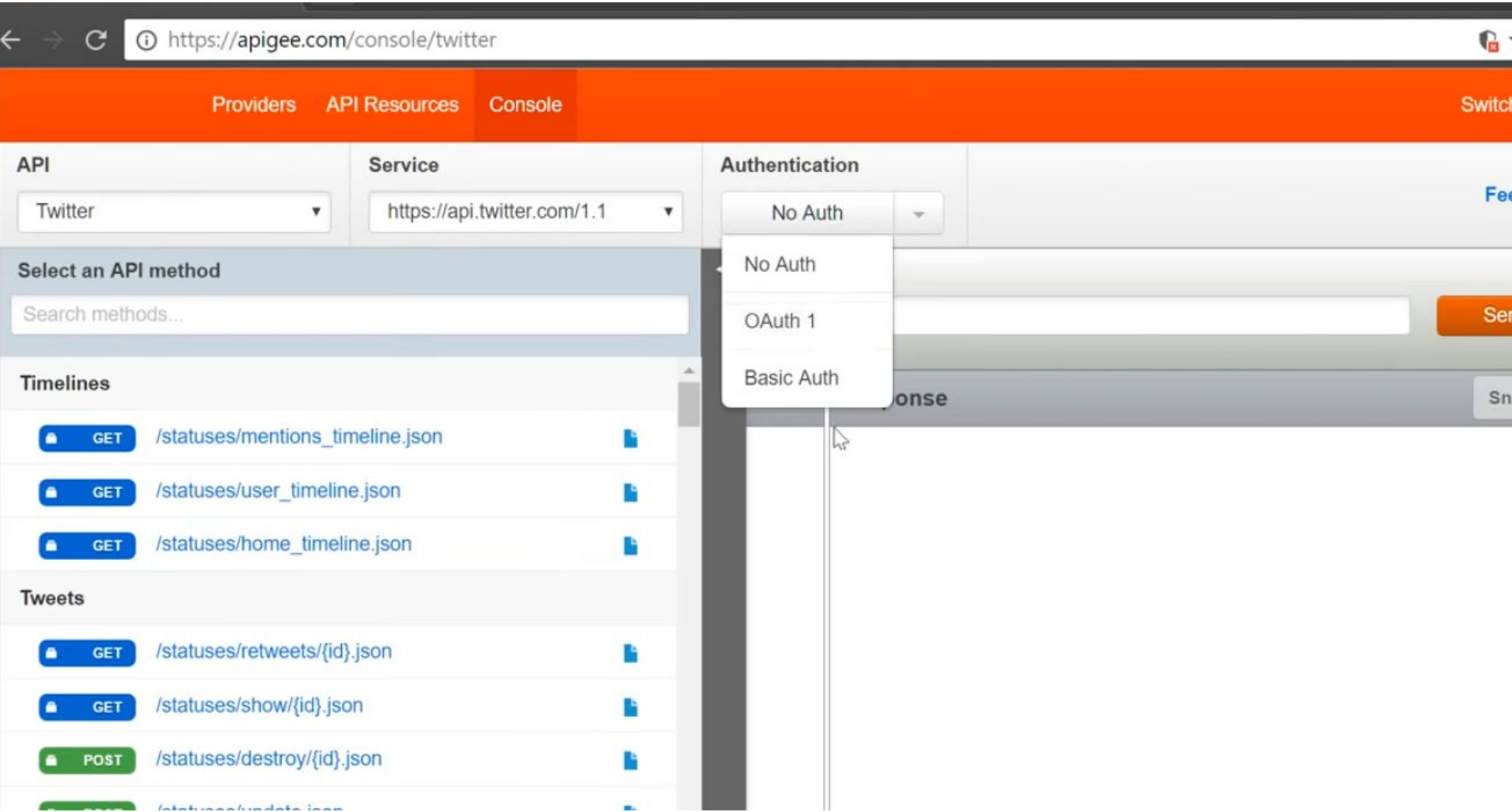
Send this request when you're ready ↗

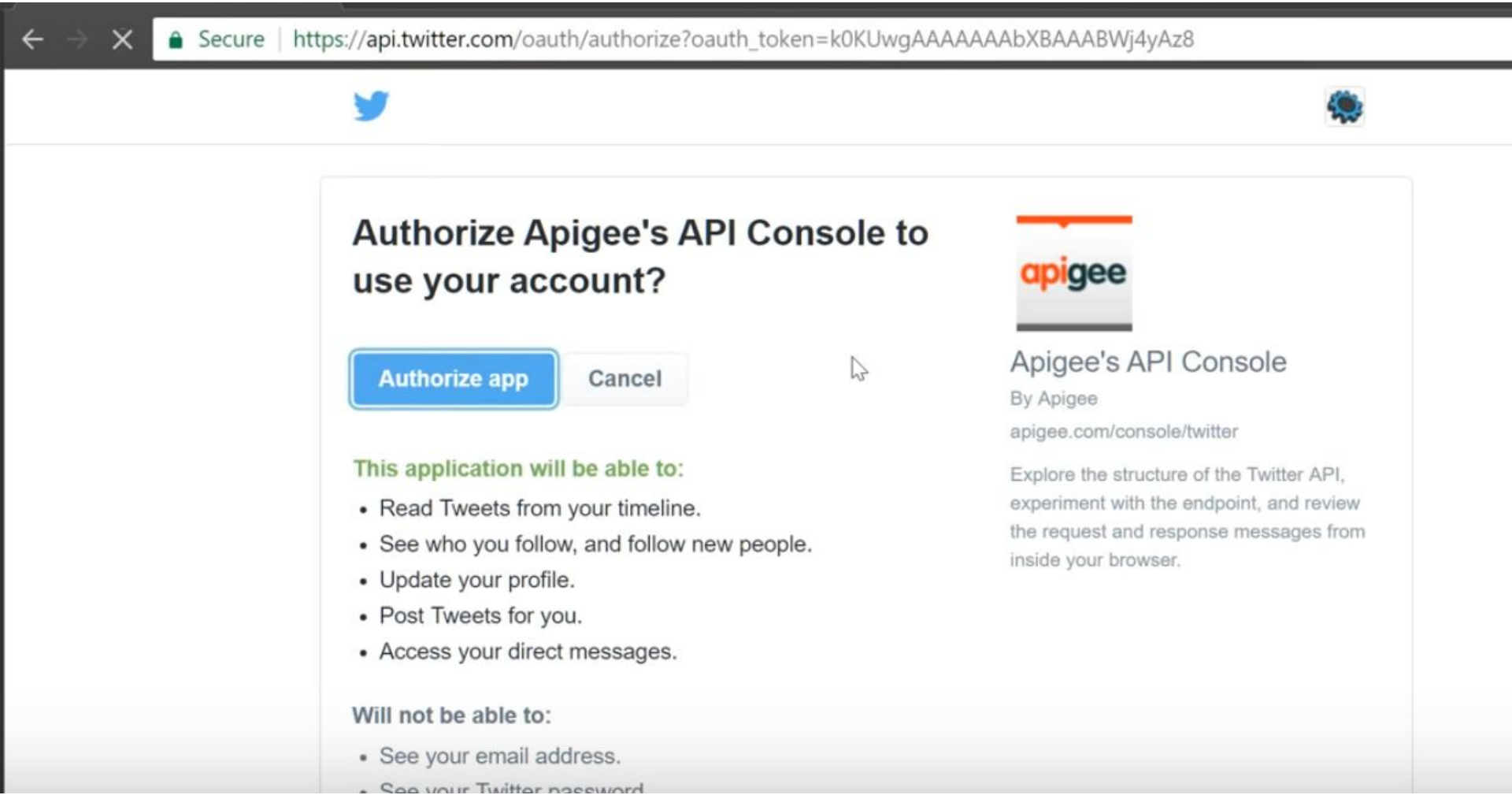

https://apigee.com/console/flickr
Request URL
GET
http://api.flickr.com/services/rest/?api_key=d524b0311c2d54ed9c0dd34408463693&method=flickr.photos.search
Send
Query*
Template
Headers
Parameter
Value
Description
*Required
api_key*
d524b0311c2d54ed9c0dd34408463693
Your API application key.
format
json
Response format. Accepted values: rest (REST), xmlrpc (XML-RPC), soap (SOAP), soap2 (Alternate SOAP) json (JSON), php_serial (Serialized PHP).
Request
Response
Snapsh
Send this request when you're ready ↗


https://apigee.com/console/flickr?req=%7B"resource"%3A"flickr_photos_search"%2C"params"%3A%7B"query"%3A%7B"api_key"%3A"d52
Providers
API Resources
Console
API
Flickr
Service
http://api.flickr.com/services/
Authentication
No Auth
Request URL
GET
https://api.flickr.com/services/rest/?api_key=d524b0311c2d54ed9c0dd34408463693&format=json&text=lily&method=flickr.photos.se
Query*
Template
Headers
This Method needs Authentication. Please select an Authentication scheme.
Request
GET /services/rest/?
api_key=d524b0311c2d54ed9c0dd34408463693&format=jsc
HTTP/1.1
Host: api.flickr.com
X-Target-URI: https://api.flickr.com
Connection: Keep-Alive
Response
HTTP/1.1 200 OK
X-Frame-Options: SAMEORIGIN
Age: 2
Vary: Accept-Encoding
Transfer-Encoding: chunked
Date: Tue, 14 Feb 2017 19:53:20 GMT
P3P: policyref="https://policies.yahoo.com/w3c/p3p.xml"
CP="CAO DSP COR CUR ADM DEV TAI PSA PSD IVAi IVDi CONi


https://apigee.com/console/flickr?req=%7B"resource"%3A"flickr_photos_search"%2C"params"%3A%7B"query"%3A%7B"api_key"%3A"d52
Request URL
GET
https://api.flickr.com/services/rest/?api_key=d524b0311c2d54ed9c0dd34408463693&format=json&text=lily&method=flickr.photos.se
Send
Query*
Template
Headers
This Method needs Authentication. Please select an Authentication scheme.
Request
GET /services/rest/?
api_key=d524b0311c2d54ed9c0dd34408463693&format=jsc
HTTP/1.1
Host: api.flickr.com
X-Target-URI: https://api.flickr.com
Connection: Keep-Alive
Response
Snap
P3P: policyref="https://policies.yahoo.com/w3c/p3p.xml",
CP="CAO DSP COR CUR ADM DEV TAI PSA PSD IVAi IVDi CONi TEL
OTPi OUR DELi SAMi OTRi UNRi PUBi IND PHY ONL UNI PUR FIN
COM NAV INT DEM CNT STA POL HEA PRE LOC GOV"
X-Served-By: www297.flickr.bf1.yahoo.com
Via: http/1.1 fts106.flickr.bf1.yahoo.com
(ApacheTrafficServer [cMsSf]), http/1.1
e13.ycpi.dca.yahoo.com (ApacheTrafficServer [cMsSf])
Connection: keep-alive
Content-Type: text/javascript; charset=utf-8
Server: ATS
Cache-Control: private
X-Content-Type-Options: nosniff
jsonFlickrApi({"photos":{"page":1,"pages":5278,"perpage":100,"total"


https://apigee.com/console/flickr?req=%7B"resource"%3A"flickr_photos_search"%2C"params"%3A%7B"query"%3A%7B"api_key"%3A"d52
Providers
API Resources
Console
API
Service
Authentication
Flickr
http://api.flickr.com/services/
No Auth
Salesforce Sandbox
SendGrid
Shopping.com
Skyrock
SoundCloud
Spotify
Stack Exchange
Tropo
Tumblr
Twilio
Twitter
Urban Airship
Weather Underground
WeatherBug
World of Warcraft
Yahoo Weather
Yellow Pages
YouTube
Youphoric
Zappos
.flickr.com/services/rest/?api_key=d524b0311c2d54ed9c0dd34408463693&format=json&text=lily&method=flickr.photos.se
Headers
This Method needs Authentication. Please select an Authentication scheme.
/?
2d54ed9c0dd34408463693&format=jsc
pi.flickr.com
Response
P3P: policyref="https://policies.yahoo.com/w3c/p3p.xml",
CP="CAO DSP COR CUR ADM DEV TAI PSA PSD IVAi IVDi CONi T
OTPi OUR DELi SAMi OTRi UNRi PUBi IND PHY ONL UNI PUR FI
COM NAV INT DEM CNT STA POL HEA PRE LOC GOV"
X-Served-By: www297.flickr.bf1.yahoo.com
Via: http/1.1 fts106.flickr.bf1.yahoo.com
(ApacheTrafficServer [cMsSf]), http/1.1
e13.ycpi.dca.yahoo.com (ApacheTrafficServer [cMsSf])
Connection: keep-alive


https://apigee.com/console/twitter
Providers
API Resources
Console
API
Twitter
Service
https://api.twitter.com/1.1
Authentication
No Auth
No Auth
OAuth 1
Basic Auth
Select an API method
Search methods...
Timelines
GET /statuses/mentions_timeline.json
GET /statuses/user_timeline.json
GET /statuses/home_timeline.json
Tweets
GET /statuses/retweets/{id}.json
GET /statuses/show/{id}.json
POST /statuses/destroy/{id}.json


Secure
https://api.twitter.com/oauth/authorize?oauth_token=k0KUwgAAAAAAAbXBAAABWj4yAz8
Authorize Apigee's API Console to use your account?
Authorize app
Cancel
This application will be able to:
Read Tweets from your timeline.
See who you follow, and follow new people.
Update your profile.
Post Tweets for you.
Access your direct messages.
Will not be able to:
See your email address.
apigee
Apigee's API Console
By Apigee
apigee.com/console/twitter
Explore the structure of the Twitter API, experiment with the endpoint, and review the request and response messages from inside your browser.

https://apigee.com/console/twitter?authTypeVal=oauth1&afterSuccessfulAuth=true

Providers API Resources Console

API

Twitter

Service

https://api.twitter.com/1.1

Authentication

twitter-theclevertechie

Select an API method

Search methods...

GET /statuses/home_timeline.json

Tweets

GET /statuses/retweets/{id}.json

GET /statuses/show/{id}.json

POST /statuses/destroy/{id}.json

POST /statuses/update.json

POST /statuses/

Response

Send this request when you're ready ↗

Updates the authenticating user's status, also known as tweeting. To upload an image to accompany the tweet, use POST statuses/update_with_media

update_with_media). For each update attempt, the update text is compared with the authenticating user's recent tweets. Any attempt that would resu

Therefore, a user cannot submit the same status twice in a row. While not rate limited by the API a user is limited in the number of tweets they can cr

https://apigee.com/console/twitter?authTypeVal=oauth1&afterSuccessfulAuth=true

Request URL

POST

https://api.twitter.com/1.1/statuses/update.json?status=Happy%20Valentine's%20Day!&display_coordinates=false

Sen

Query* Template Headers Body

Parameter	Value	Description
status*	Happy Valentine's Day!	The text of your status update, typically up to 140 characters. URL encode as necessary. t.co link short-url wrapping (https://dev.twitter.com/docs/tco-link-wrapper/faq) may effect character counts.
in_reply_to_status_id		The ID of an existing status that the update is in reply to. Not This parameter will be ignored unless the author of the tweet this parameter references is mentioned within the status text

*Requi

Request

Response

Sna

Send this request when you're ready ↗

https://apigee.com/console/twitter?req=%7B"resource"%3A"statuses_update"%2C"params"%3A%7B"query"%3A%7B"status"%3A"Happy%

Request URL

POST ▾ https://api.twitter.com/1.1/statuses/update.json?status=Happy%20Valentine's%20Day!&display_coordinates=false

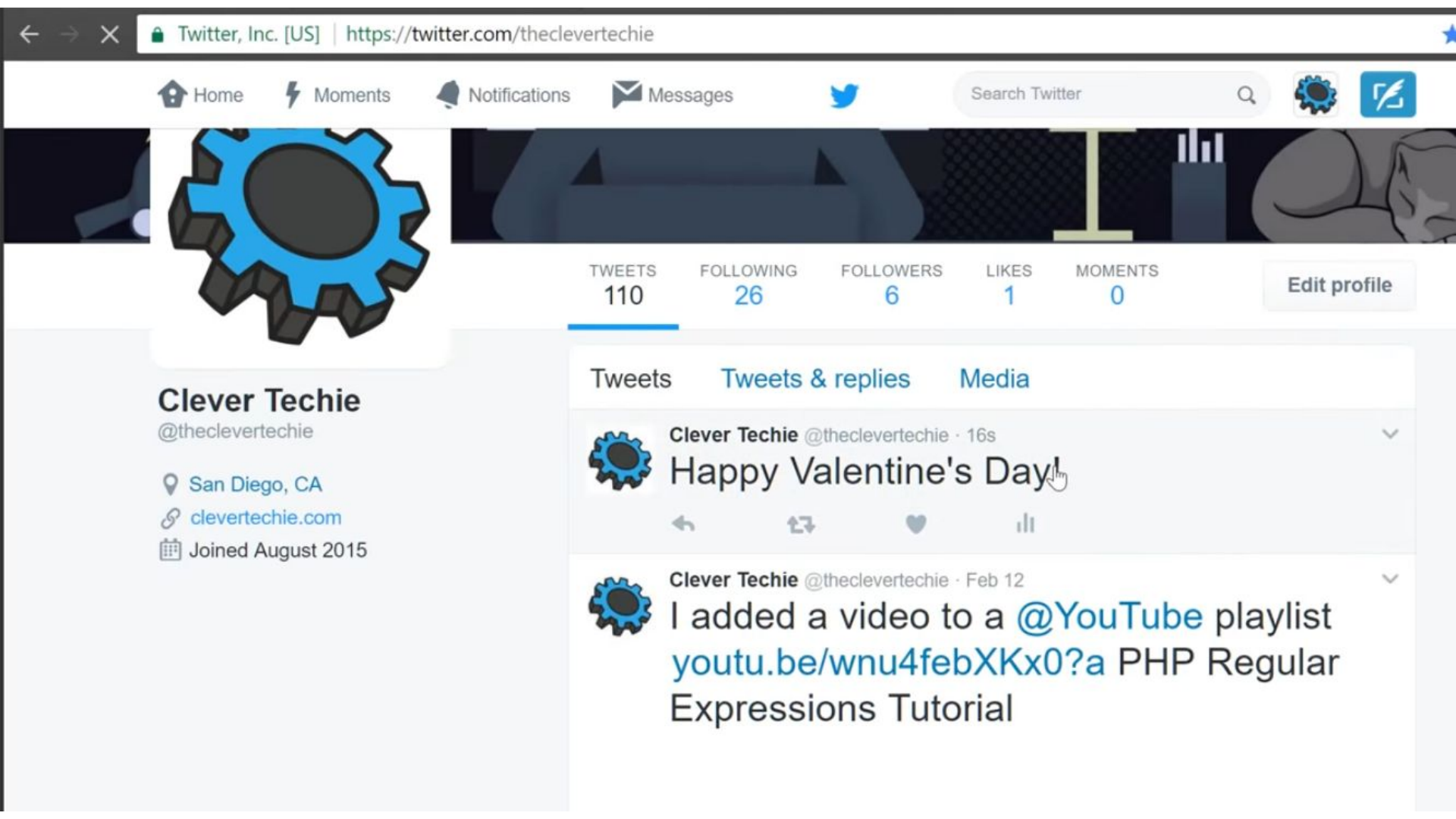
Query * | Template | Headers | Body

Request

```
POST /1.1/statuses/update.json?
status=Happy%20Valentine's%20Day!&display_coordinat
HTTP/1.1
Authorization: OAuth
oauth_consumer_key="DC0sePOBbQ8bYdC8r4Smg",oauth_signature_metho
SHA1",oauth_timestamp="1487102177",oauth_nonce="3142447219",oaut
KKN2O1xGOJYb9tj57e90gE8TGpI2iisGiU644fe",oauth_signature="c7vxx5
Host: api.twitter.com
Content-Length: 0
X-Target-URI: https://api.twitter.com
Connection: Keep-Alive
```

Response

```
HTTP/1.1 200 OK
x-frame-options: SAMEORIGIN
last-modified: Tue, 14 Feb 2017 19:56:17 GMT
status: 200 OK
Content-Length: 2289
x-response-time: 148
Connection: keep-alive
x-transaction: 00c27be40051f77d
Server: tsa_b
pragma: no-cache
cache-control: no-cache, no-store, must-revalidate, pre-
check=0, post-check=0
x-connection-hash: 84c54dcb969606b079fb2829aecc9e01
x-xss-protection: 1; mode=block
```


Twitter, Inc. [US]
https://twitter.com/theclevertechie
Home
Moments
Notifications
Messages
Search Twitter
TWEETS 110
FOLLOWING 26
FOLLOWERS 6
LIKES 1
MOMENTS 0
Edit profile
Clever Techie
@theclevertechie
San Diego, CA
clevertechie.com
Joined August 2015
Tweets
Tweets & replies
Media
Clever Techie @theclevertechie · 16s
Happy Valentine's Day!
Clever Techie @theclevertechie · Feb 12
I added a video to a @YouTube playlist youtu.be/wnu4febXKx0?a PHP Regular Expressions Tutorial

No.	SOAP	REST
1)	SOAP is a **protocol**.	REST is an **architectural style**.
2)	SOAP stands for **Simple Object Access Protocol**.	REST stands for **REpresentational State Transfer**.
3)	SOAP **can't use REST** because it is a protocol.	REST **can use SOAP** web services because it is a concept and can use any protocol like HTTP, SOAP.
4)	SOAP **uses services interfaces to expose the business logic**.	REST **uses URI to expose business logic**.
5)	**JAX-WS** is the java API for SOAP web services.	**JAX-RS** is the java API for RESTful web services.
6)	SOAP **defines standards** to be strictly followed.	REST does not define too much standards like SOAP.
7)	SOAP **requires more bandwidth** and resource than REST.	REST **requires less bandwidth** and resource than SOAP.

8)	SOAP **defines its own security**.	RESTful web services **inherits security measures** from the underlying transport.
9)	SOAP **permits XML** data format only.	REST **permits different** data format such as Plain text, HTML, XML, JSON etc.
10)	SOAP is **less preferred** than REST.	REST **more preferred** than SOAP.

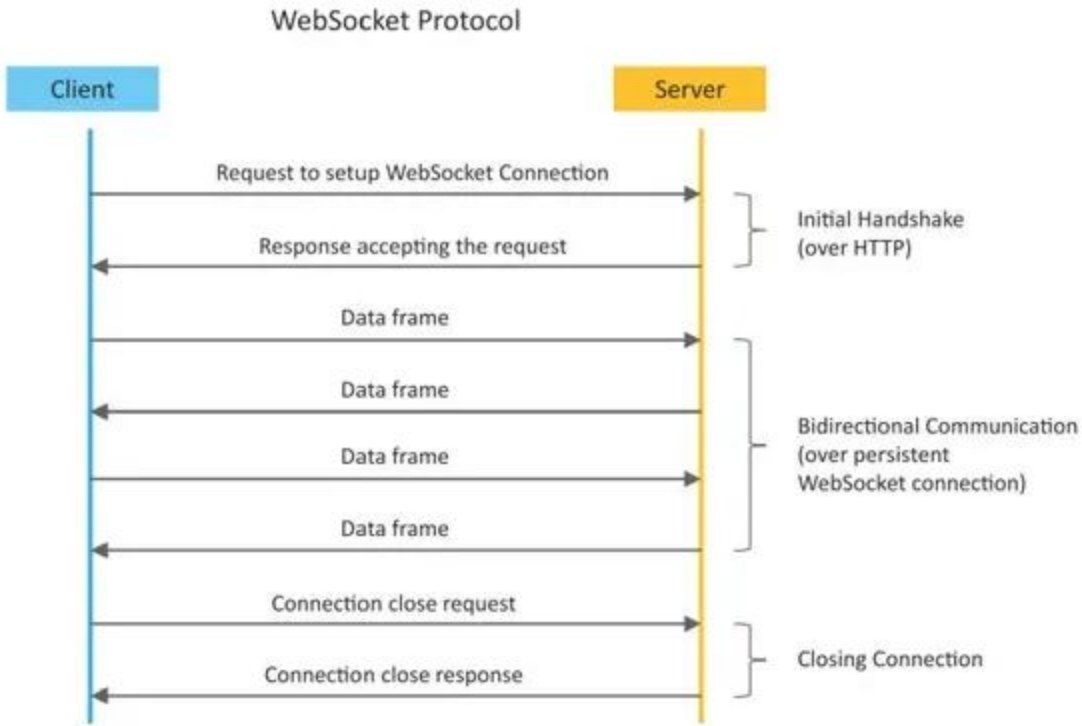
. **Web Socket Based Communication APIs :**

Web Socket APIs allow bi-directional, full-duplex communication between clients and servers.

It follows the exclusive pair communication model. This Communication API does not require a new connection to be set up for each message to be sent between clients and servers.

Once the connection is set up the messages can be sent and received continuously without any interruption.

WebSocket APIs are suitable for IoT Applications with low latency or high throughput requirements.


WebSocket Protocol
Client
Server
Request to setup WebSocket Connection
Response accepting the request
Initial Handshake
(over HTTP)
Data frame
Data frame
Data frame
Data frame
Bidirectional Communication
(over persistent
WebSocket connection)
Connection close request
Connection close response
Closing Connection

S.NO.	REST API	WEB SOCKET API
1.	It is Stateless protocol. It will not store the data.	It is Stateful protocol. It will store the data.
2.	It is Uni-directional. Only either server or client will communicate.	It is Bi-directional. Messages can be received or sent by both server or client.
3.	It is Request-response model.	It is Full duplex model.
4.	HTTP request contains headers like head section, title section.	It is suitable for real-time applications. It does not have any overhead.

5.	New TCP connection will be set up for each HTTP request.	Only Single TCP connection.
6.	Both horizontal and vertical scaling (we can add many resources and number of users both horizontally and vertically).	Only vertical scaling (we can add resources only vertically).
7.	It depends upon the HTTP methods to retrieve the data..	It depends upon the IP address and port number to retrieve the data
8.	It is slower than web socket regarding the transmission of messages.	web socket transmits messages very fastly than REST API.
9.	It does not need memory or buffers to store the data.	It requires memory and buffers to store the data.