REST stands for Representational State Transfer. It defines a framework wherein components in a networked environment can exchange data. These components are software packages running on computers or nodes on the network. Some components are consumers of data and are called clients, and some components are providers of data and are called servers.

The server to client can communicate and exchange data in a one to one, one to many, many to many and many to one architecture. The complexity of this network architecture requires the adoption of a standard way of communication between these components.

It hasn't been until the advent of markup languages such as XML, HTML, CSS, and JSON as data sources and transfer protocols such as Simple Object Access Protocol(SOAP) and Representational State Transfer that computer scientist and IT professionals can rely on a framework that can be used to achieve their goals.

The REST architectural style specifies a uniform interface that facilitates properties such as performance, scalability, and modifiability. These properties enable services on the web to exchange data seamlessly.

In the REST architectural style, data and functionality are considered resources and are accessed using Uniform Resource identifiers(URIs), typically linked on the web. It is a client/server architecture and is designed to use a stateless communication protocol such as HTTP. Clients and servers exchange the representations of resources by using a standard interface protocol such as JSON.

REST provides a standard way for a web service provider to expose its resources on the web. Similarly, the client accesses those services via the HTTP methods GET,PUT,DELETE, and POST. The reply to these is generally JSON or XML data which the client is prepared to digest. The resources can be considered as objects that represent the resource. As such these objects have methods and attributes. Since the objects are represented in markup languages, manipulations of them becomes standard. For example the python library for JSON can be used to parse a reply to GET. The description of each method is described below:

- GET – Used to retrieve information from the endpoint
- POST – Used to create a new entity on the endpoint or to update an existing entity
- PUT – Used to create a new entity or to update an existing entity. The most significant difference between POST and PUT is that PUT is idempotent. This means repeated calls of these methods has the same affect on the server side, but not necessarily on the client side.
- DELETE –Used to request that a resource be removed. DELETE is also idempotent except the reply to 1st DELETE is different than subsequent ones because the object dies not exist in subsequent times so the client will get a "No Content" or "Not found" error.
- PATCH – Used to request a partial update to a resource.

CRUD stands for Create, Read, Update, and Delete. But put more simply, in regards to its use in RESTful APIs, CRUD is the standardized use of HTTP Action Verbs as defined above.

When we make a request of a webservice the webservice will typically replies with a status code. These status codes can be used to determine if a request was successful if an error has occurred or other various operations.

- Common Status Codes: ○
    - 1xx – Informational
    - 2xx – Success
    - 3xx – Redirection
    - 4xx – Client Error
    - 5xx – Server Error

When making a connection to an webservice it is a good idea to validate the connection was successful with a try block before sending additional requests

APIs | REST | REST APIs Demystified

Python Requests Tutorial: Request Web Pages, Download Images, POST Data, Read JSON, and More