Python JSON:

In this tutorial, we'll see how we can create, manipulate, and parse JSON in Python using the standard a `json` module. The built-in Python json module provides us with methods and classes that are used to parse and manipulate JSON in Python.

What is JSON:

JSON (an acronym for JavaScript Object Notation) is a data-interchange format and is most commonly used for client-server communication

Example:

```
{"name": "jane doe", "salary": 9000, "email": "JaneDoe@pynative.com"}
```

A JSON is an unordered collection of key and value pairs, resembling Python's native dictionary.

- Keys are unique Strings that cannot be null.
- Values can be anything from a String, Boolean, Number, list, or even null.
- A JSONO can be represented by a String enclosed within curly braces with keys and values separated by a colon, and pairs separated by a comma

Why do we use JSON?

- Whenever the client needs information, it calls the server using a URI, and the server returns data to the client in the form of JSON. Later we can use this data in our application as per our requirement.
- Also, when the client application wants to store the data on the server. It can POST that data in the form of JSON.
- It is used primarily to transmit data between a server and web application, serving as an alternative to XML.

JSON is most commonly used for client-server communication because:

- It is human readable.
- It's both easy to read/write and
- JSON is language-independent.

Why JSON is Better than XML ??

- XML is a much more difficult to Parse the data than JSON.
- JSON doesn't use the Tags . JSON is Shorter and Use Arrays.
- It is very Fast to read and write the data
- For AJAXs applications , JSON is Faster and easier than XML.

Python json Module:

Python comes with a built-in module called json for working with JSON data. You only need to add `import json` at the start of your file and you are ready to use it.

Example 1:

json_data = { "fname" : "Srinivas" , "lname" : "Rao" , "age" : 27 }

Example 2:

json_data = { "names" : ["Ramu" , "Ravi" , "Raju"], "data" : { "name" : "Sri" , "location" : "HYD"} }

Note 1: In JSON structure , Curly braces hold the objects and Square brackets hlod the Arrays.

Note 2: Each key and value should be in a Double quotes if it is strings.

MIME value:

- **MIME** stand for Multipurpose Internet Mail Extention.
- By uing MIME type attribute we will represents what type of data we want to return as response when we are sending the request.
- The official Internet media type for JSON is **application/json**.

Syntax: return HttpResponse(resp , content_type = 'application/json')

Example 1: Write a program to define an Employees object with an Array of 3 employees by uing JSON structure ????

Using JSON:

```
{
    "employees" : [
            { "fname" : "Srinivas" , "lname" : "Rao" } ,
            { "fname" :  "Virat", "lname" : "Kohli" } ,
            { "fname" : "Rohit" , "lname" : "Sharma" } ,
    ]
}
```

Example 2: Write a program to define an Employees object with an Array of 3 employees by uing XML structure ????

```
<employees>
    <employee>
        <fname> Srinivas </fname>
        <lname> Rao </lname>
    </employee>
    <employee>
        <fname>Rohit </fname>
        <lname>Sharma </lname>
    </employee>
    <employee>
        <fname> Virat</fname>
        <lname>Kohli </lname>
    </employee>
</employees>
```