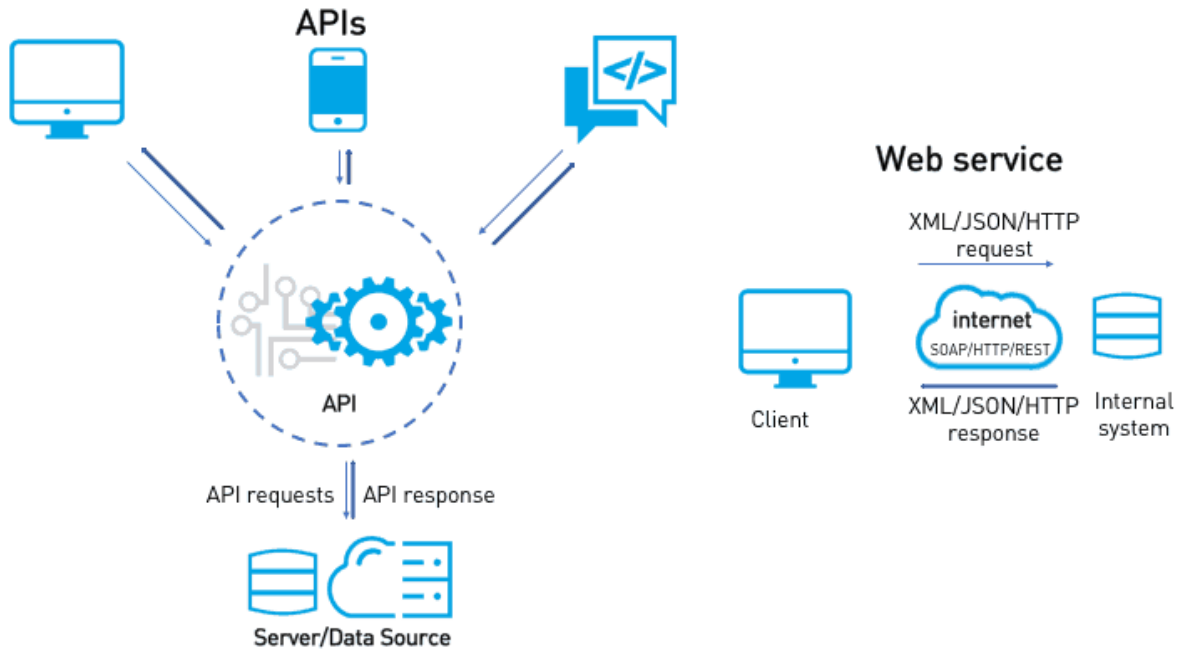
What is Web Service

(वेब सर्विस क्या है?)



API + HTTP Protocol → WebService (Example: we are able to book tickets of any airline from MakeMyTrip.com where the ability to book ticket of that airline is made available to the requestor through some authentication and way)

API → In a layman language API is set of methods which can give some output when appropriate input is given and can be independent

It is used for A2A (application-to-application) communication and interfacing. These processes involve programs, messages, documents, and/or objects

A key feature of web services is that applications can be written in various languages and are still able to communicate by exchanging data with one another via a web service between clients and servers. A client summons a web service by sending a request and the service then responses with an response

A web service comprises these essential functions

- Available over the internet or intranet networks
- Standardized messaging system
- Independent of a single operating system or programming language
- Discoverable through a simple location method

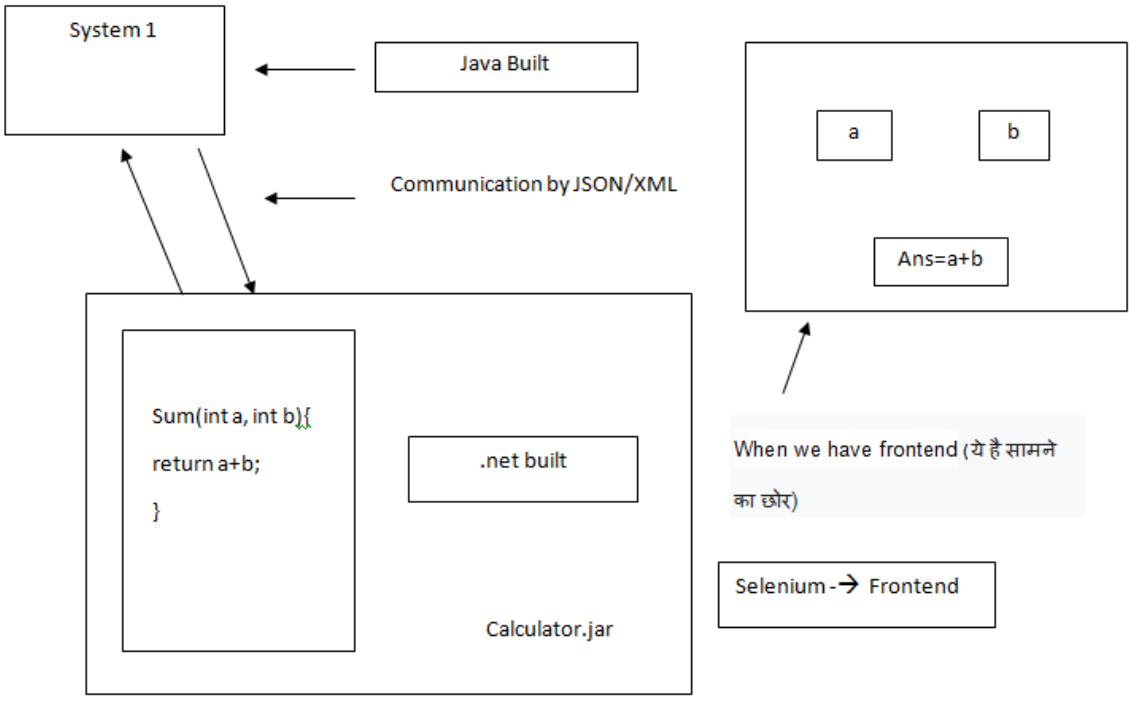
A web service supports communication among numerous apps with HTML, XML, WSDL, SOAP, and other open standards.

What are the Different Types of Web Services?

There are a few central types of web services: XML-RPC, UDDI, SOAP, and REST:

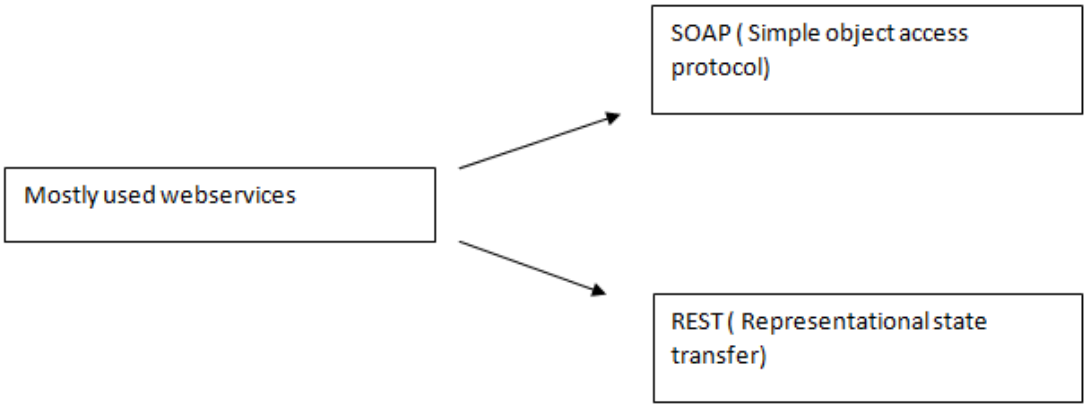
- **XML-RPC** (Remote Procedure Call) is the most basic XML protocol to exchange data between a wide variety of devices on a network. It uses HTTP to quickly and easily transfer data and communication other information from client to server
- **UDDI** (Universal Description, Discovery, and Integration) is an XML-based standard for detailing, publishing, and discovering web services. It's basically an internet registry for businesses around the world. The goal is to streamline digital transactions and e-commerce among company systems
- **SOAP**, which will be described in detail later in the blog, is an XML-based Web service protocol to exchange data and documents over HTTP or SMTP (Simple Mail Transfer Protocol). It allows independent processes operating on disparate systems to communicate using XML
- **REST**, which will also be described in great detail later in the blog, provides communication and connectivity between devices and the internet for API-based tasks. Most RESTful services use HTTP as the supporting protocol

Some more explanation:



Question: Why we need API or WebService

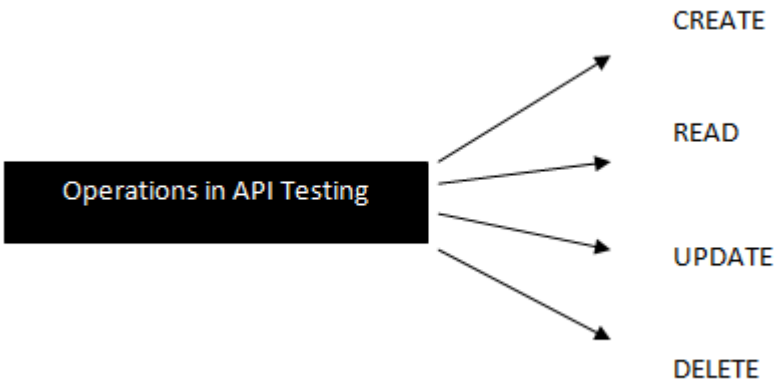
Answer: when we don't have front-end (also depend on our choice when we want to test specifically from back-end).



Question: What is the difference between SOAP and REST

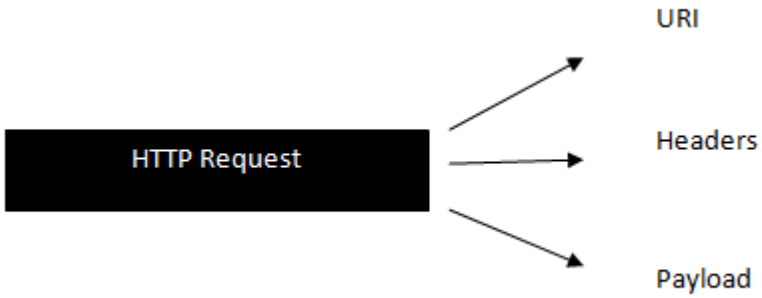
Answer:

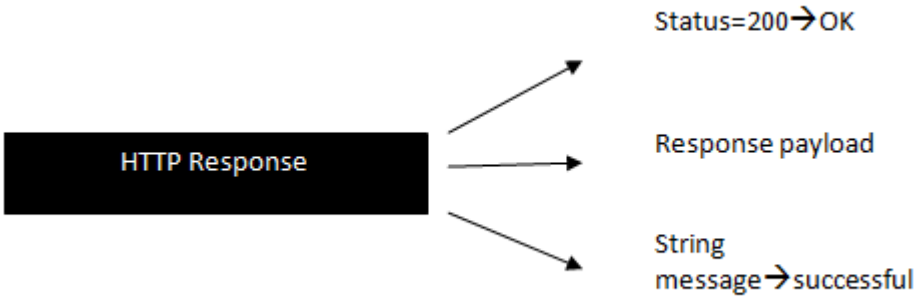
Sr No	SOAP	REST
1.	SOAP→Protocol	REST→Architectural style
2	Can not use REST in SOAP because it is a protocol	Can use SOAP in REST Architectural style
3	Use service interfaces to expose business logic	Uses URI to expose business logic
4	JAX-WS →Java API for SOAP	JAX-RS→Java API for RESTful services
5	Defines standard to be strictly followed	Does not defines too much standards
6	Requires more bandwidth + resource	Requires less bandwidth + resource
7	Defines own security	Inherit security measures from underlying transport
8	Permits XML data format only	Permits plain text, HTML, XML, JSON



Question : What does HTTP Request contains?

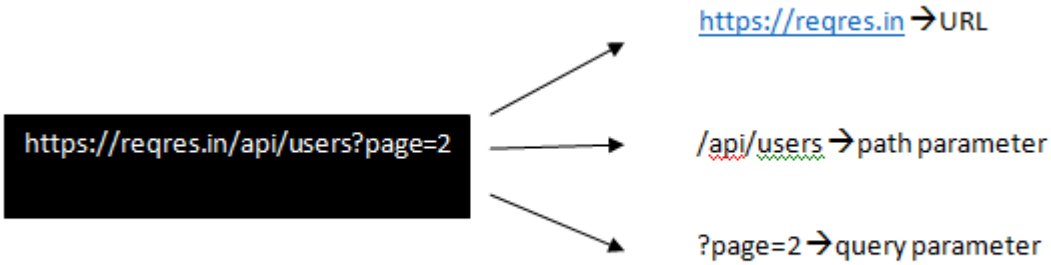
Answer:

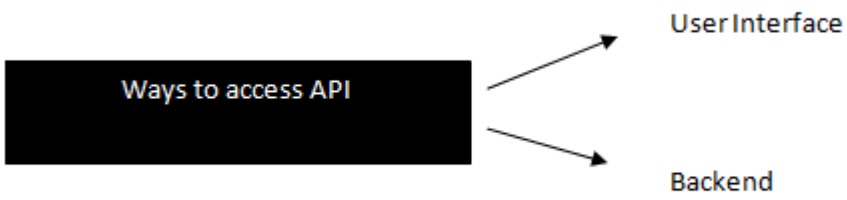




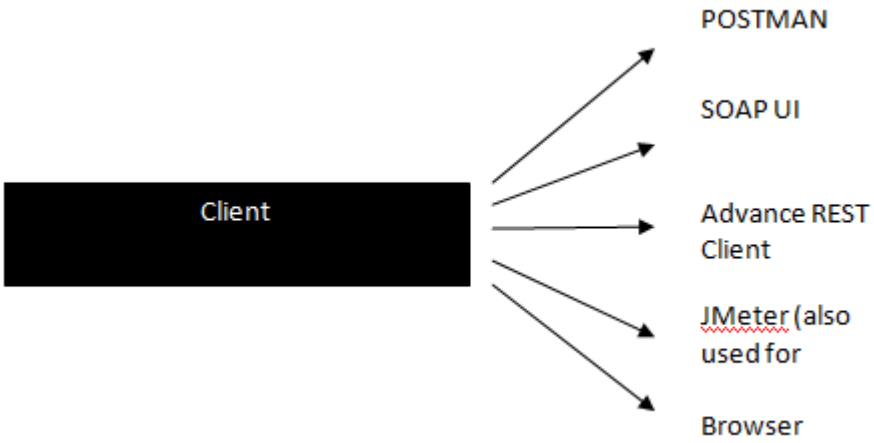
URI → URL + Path parameter + query parameter

Eg : http://api.com/service/account/1

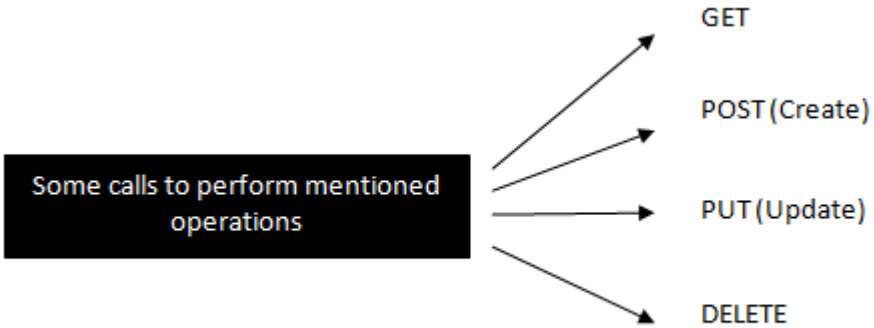




Client types



Calls we have to make to perform those operations:



All API testing will be focused on such type of operations where we have to make different type of calls to perform that operation successfully.

Question: But what are XML and JSON?

Answer:**1.--> XML**

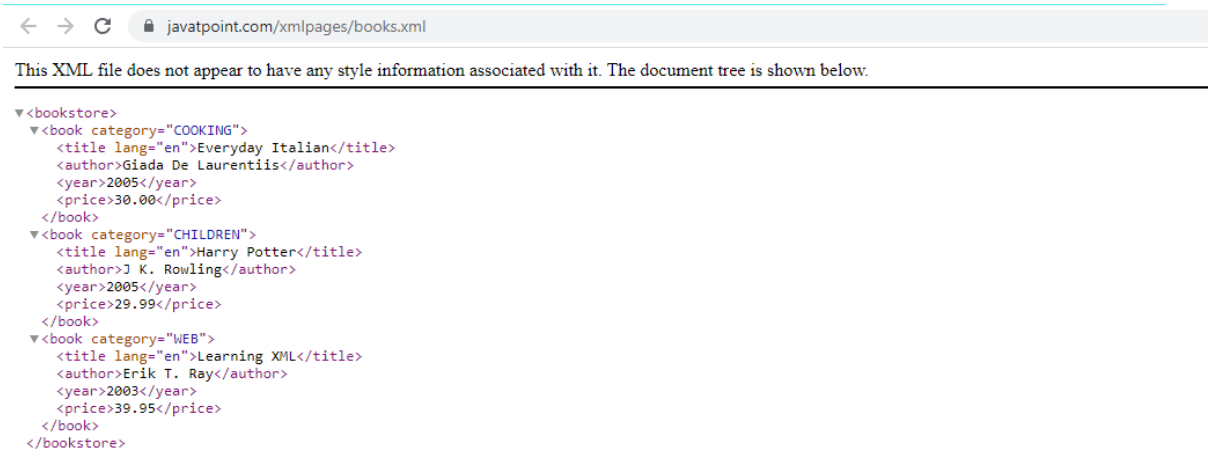
- XML is designed to store and transport data.
- XML is not a replacement for HTML.
- XML tags are not predefined. You must define your own tags.
- XML is platform independent and language independent.
- The main benefit of xml is that you can use it to take data from a program like Microsoft SQL, convert it into XML then share that XML with other programs and platforms. You can communicate between two platforms which are generally very difficult.

Example:

```
<bookstore>
<book category="COOKING">
<title lang="en">Everyday Italian</title>
<author>Giada De Laurentiis</author>
```

```
<year>2005</year>
<price>30.00</price>
</book>
<book category="CHILDREN">
<title lang="en">Harry Potter</title>
<author>J K. Rowling</author>
<year>2005</year>
<price>29.99</price>
</book>
<book category="WEB">
<title lang="en">Learning XML</title>
<author>Erik T. Ray</author>
<year>2003</year>
<price>39.95</price>
</book>
</bookstore>
```

If you pass above xml document through browser to web service than it can be viewed as below in browser



2. --> JSON

- JSON stands for JavaScript Object Notation.
- JSON is lightweight data-interchange format.
- JSON is easy to read and write than XML.
- JSON is language independent.
- JSON supports array, object, string, number and values.

Example:

```
{"employees":[
{"name":"Sonoo", "email":"sonoojaiswal1987@gmail.com"},
{"name":"Rahul", "email":"rahul32@gmail.com"},
{"name":"John", "email":"john32bob@gmail.com"}
]}
```

Question: OK! But what are the advantages of using JSON over XML or vice versa

Answer:

Sr. No	JSON	XML
1	JSON object has a type	XML data is typeless
2	JSON types: string, number, array, Boolean	All XML data should be string
3	Data is readily accessible as JSON objects	XML data needs to be parsed.
4	JSON is supported by most browsers.	Cross-browser XML parsing can be tricky
5	JSON supports only text and number data type.	XML support various data types such as number, text, images, charts, graphs, etc. It also provides options for transferring the structure or format of the data with actual data
6	Retrieving value is easy	Retrieving value is difficult
7	A fully automated way of deserializing/serializing JavaScript.	Need to write JavaScript code to serialize/de-serialize from XML
8	It supports only UTF-8 encoding.	It supports various encoding.
9	It doesn't support comments.	It supports comments.
10	JSON files are easy to read as compared to XML.	XML documents are relatively more difficult to read and interpret.

JSON Code vs XML Code:

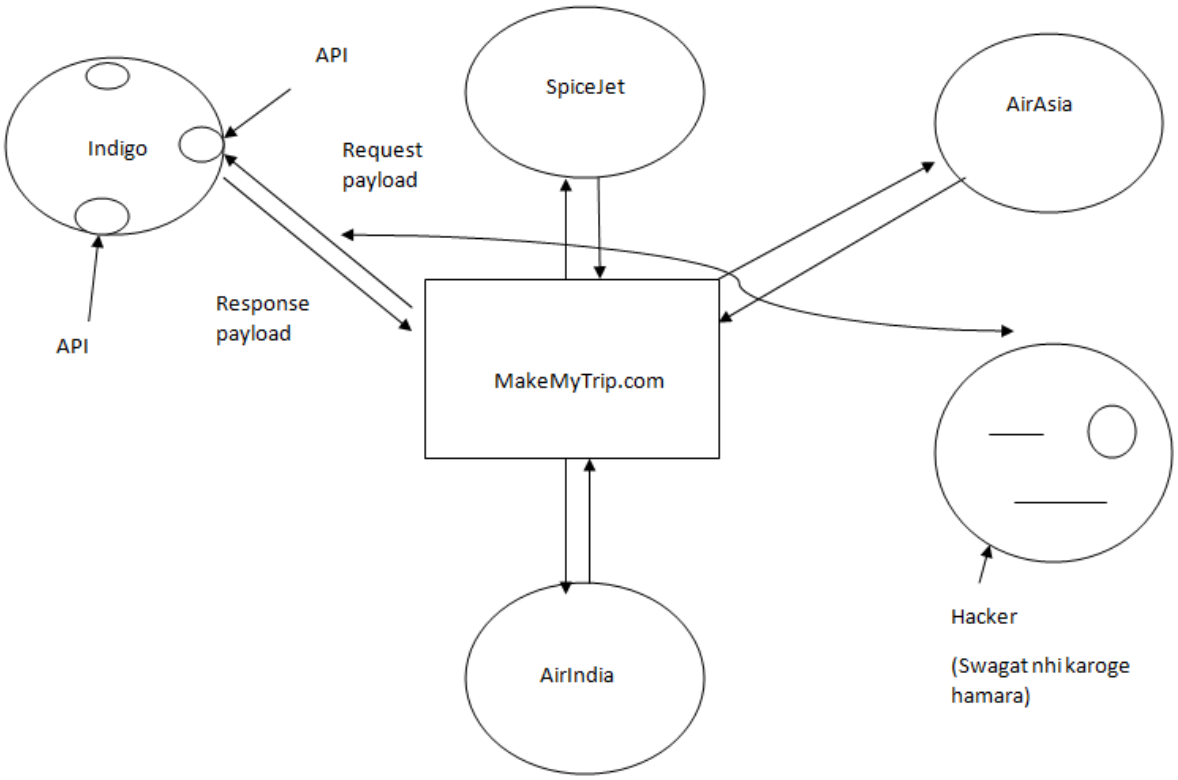
```
{
  "student": [

     {
        "id":"01",
        "name": "Tom",
        "lastname": "Price"
     },

     {
        "id":"02",
```

```
        "name": "Nick",
        "lastname": "Thameson"
     }
  ]
}
```

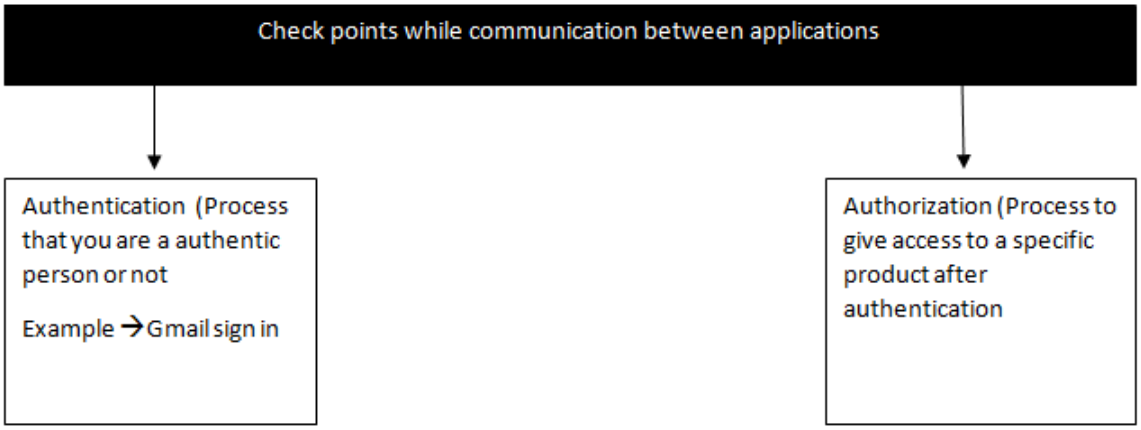
AND

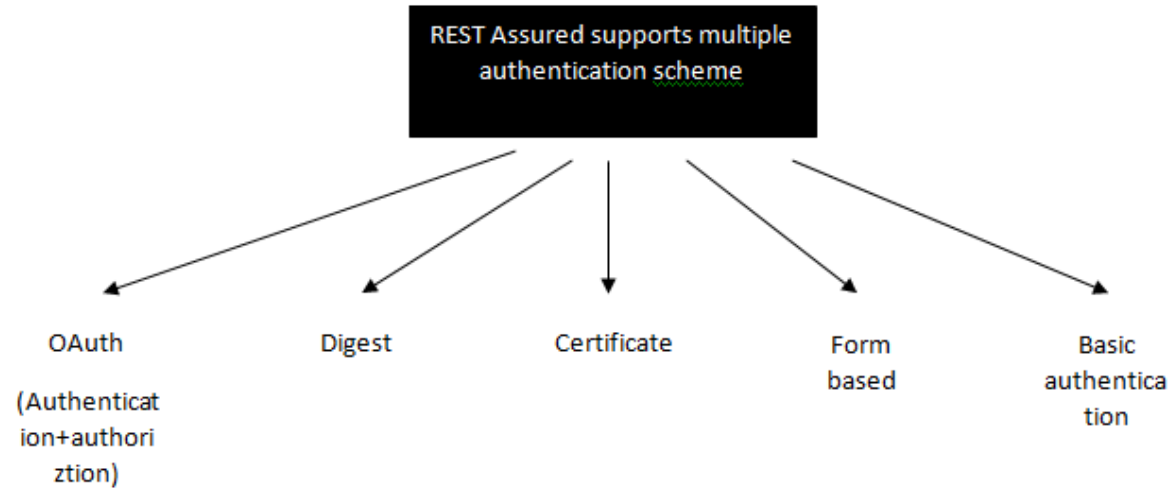
```
<?xml version="1.0" encoding="UTF-8" ?>
<root>
        <student>
                <id>01</id>
                <name>Tom</name>
                <lastname>Price</lastname>
        </student>
        <student>
                <id>02</id>
                <name>Nick</name>
                <lastname>Thameson</lastname>



        </student>
</root>
```

MakeMyTrip example further explained

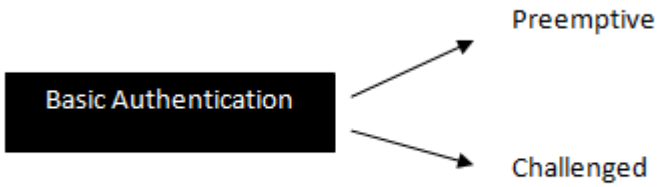


So we need to add some authentication/authorization with our request payload so that it can not be hacked by any Hacker/unauthorized access can be avoided.





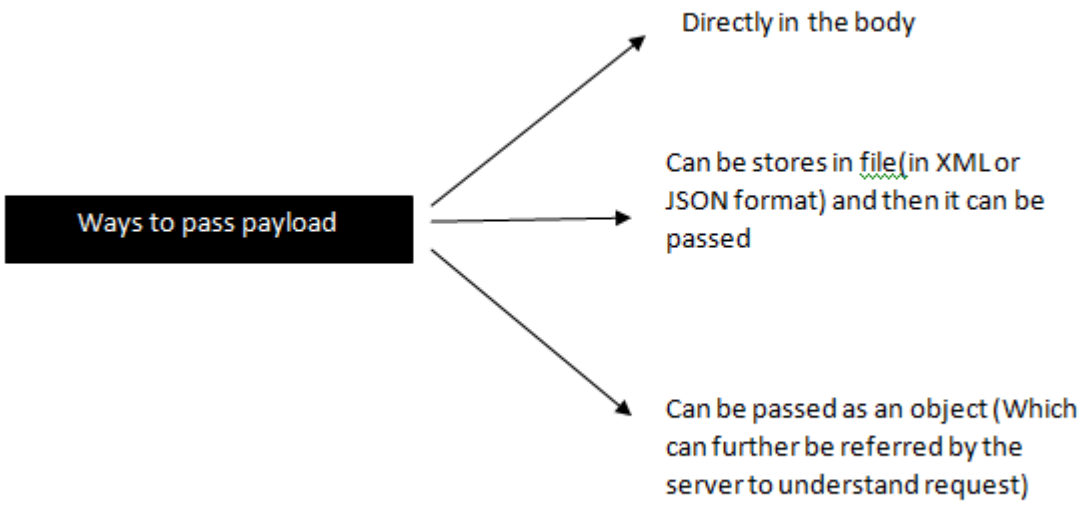
Where Basic authentication is further divided into two types:



Preemptive : It will send basic authentication credential (automatically in header) even before the server gives an unauthorized response in certain situations , thus reducing the overhead of making an additional connection

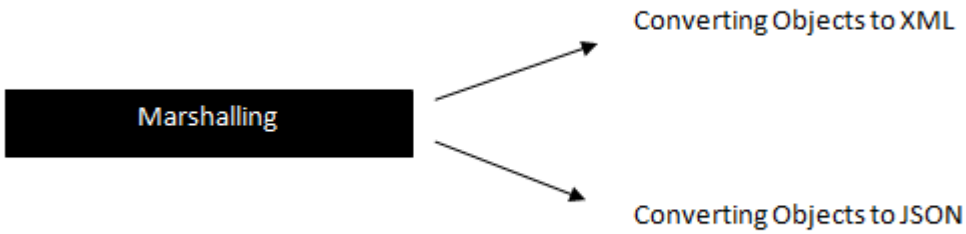
Challenged : When asked then only need to provide credentials

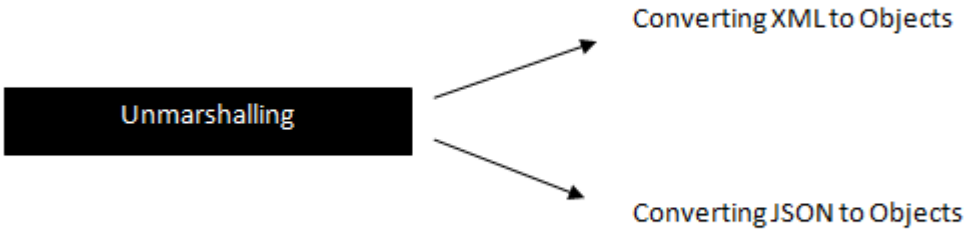
Question: Ok! But how to pass our request as payload? Is there any way to compress that request because that request can be big or small or of any size or of any lines
Answer: There are mostly 3 types of ways through which we can pass the data



Question: The process of passing this information have any name

Answer: Yes, the process is known as Marshalling or Unmarshalling





So in short we have to save our data in XML format or JSON format in a file.

Question: I'm having apprehensions regarding the method by which you are converting objects to XML/JSON, can you please show some demo.

Answer: Sure, I'm going to present a demo but now I can show you a glimpse of both methods

Marshaller method:

```
try{
    //creating the JAXB context
    JAXBContext jContext = JAXBContext.newInstance(Student.class);
    //creating the marshaller object
    Marshaller marshallObj = jContext.createMarshaller();
    //setting the property to show xml format output
    marshallObj.setProperty(Marshaller.JAXB_FORMATTED_OUTPUT, true);
    //setting the values in POJO class
    Student student = new Student("abhishek", 1163, "hadoop");
    //calling the marshall method
    marshallObj.marshal(student, new FileOutputStream("/home/knoldus/Deskto
p/student.xml"));
} catch(Exception e) {
    e.printStackTrace();
}
```

Unmarshaller method::

```
try{
    //getting the xml file to read
    File file = new File("/home/knoldus/Desktop/student.xml");
    //creating the JAXB context
    JAXBContext jContext = JAXBContext.newInstance(Student.class);
    //creating the unmarshall object
    Unmarshaller unmarshallerObj = jContext.createUnmarshaller();
    //calling the unmarshall method
    Student student=(Student) unmarshallerObj.unmarshal(file);
    System.out.println(student.getName()+" "+student.getId()+" "+student.ge
tSubject());
}catch(Exception e){
    e.printStackTrace();
}
```

Question: OK! But what will be advantage of automating WebService, why you are doing all this, how this can be beneficial to client.

Answer: See, when we have repetitive process such as booking order in mutual funds project or issuing policies in insurance project, that time we can run our script which consist of code for automating web Services, thus we will save time which could be used to do other important work..

Whatever information or data we provide in xml/json file will hit the endpoints of respective API and can give output directly without waiting for the UI to be present.

Question: What are the key steps while executing request

Answer: These are general and common steps but can be more refined according to your requirements.

- Create the client
- Generate the request
- Define the payload
- Define header
- Execute request

Question: How to implement them in terms of code?
Answer:: There are two ways with which we can implement both

1. By using HTTPClient
2. By using RESTAssured

But the way is common as mentioned before

1. Create the client
2. Generate the request
3. Define the payload
4. Define header
5. Execute request

But implementation is different for both HTTPClient and RESTAssured respectively

1. By HTTPClient --> for GET Call

i.First we need to instantiate httpClient
ii. Hit the URL
iii. Instantiate closableHTTPResponse
iv. Execute the request
v. Get status code
vi. Get body
vii. Assertions if any

Example -->

```
CloseableHttpClient httpClient = HttpClients.createDefault();
HttpGet httpget = new HttpGet(url);
CloseableHttpResponse closeableHttpResponse =
HttpClient.execute(httpget);
Int
statusCode=closeableHttpResponse.getStatusLine().getStatusCode();
System.out.println("Status code -->" +statuscode);
String responseString =
EntityUtils.toString(closableHttpResponse.getEntity(),"UTF-8");
System.out.println(responseString );
```

2. By RESTAssured --> for GET Call

i. Instantiate RESTAssured Client
ii. Hit the URL
iii. Store response into RESTAssured response
iv. Get status code

v. Get body
vi. Assertion if any

Example -->

```
Response resp=RestAssured.get(url);
Int code = resp.getStatusCode();
System.out.println("Status code is: "+code);
Assert.assertEquals(code,200);
```

Question: Can you please show some demo?

Answer: Sure