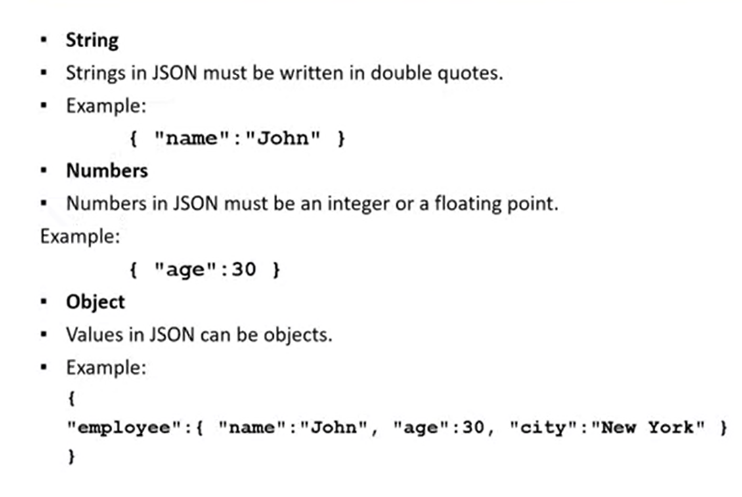
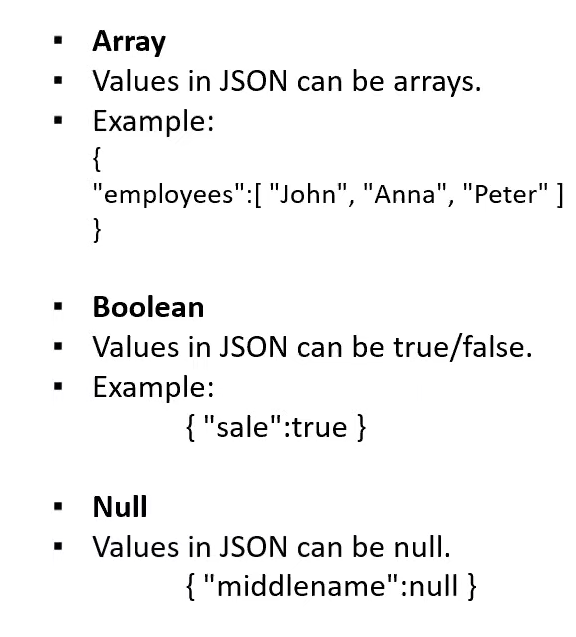
JSON

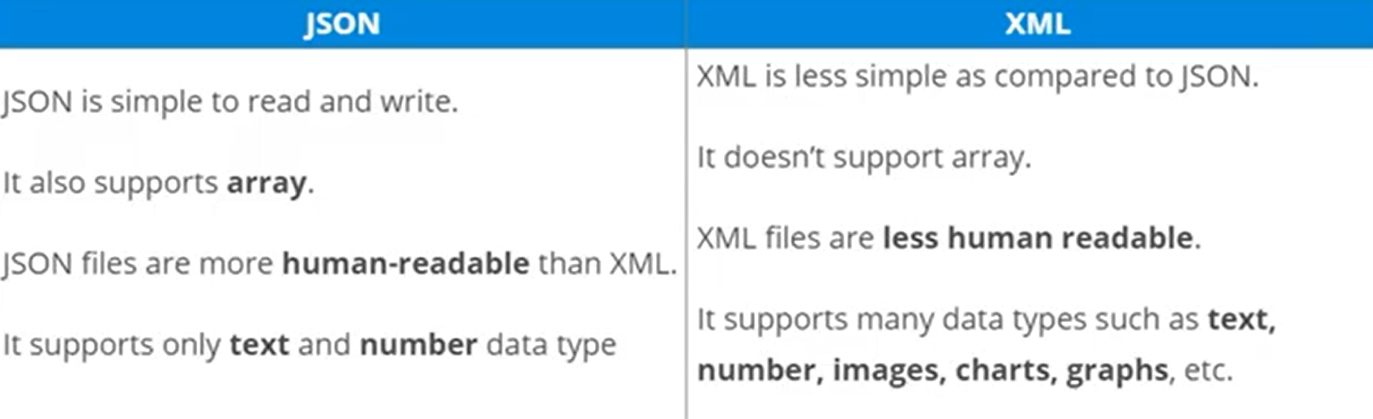
Introduction:

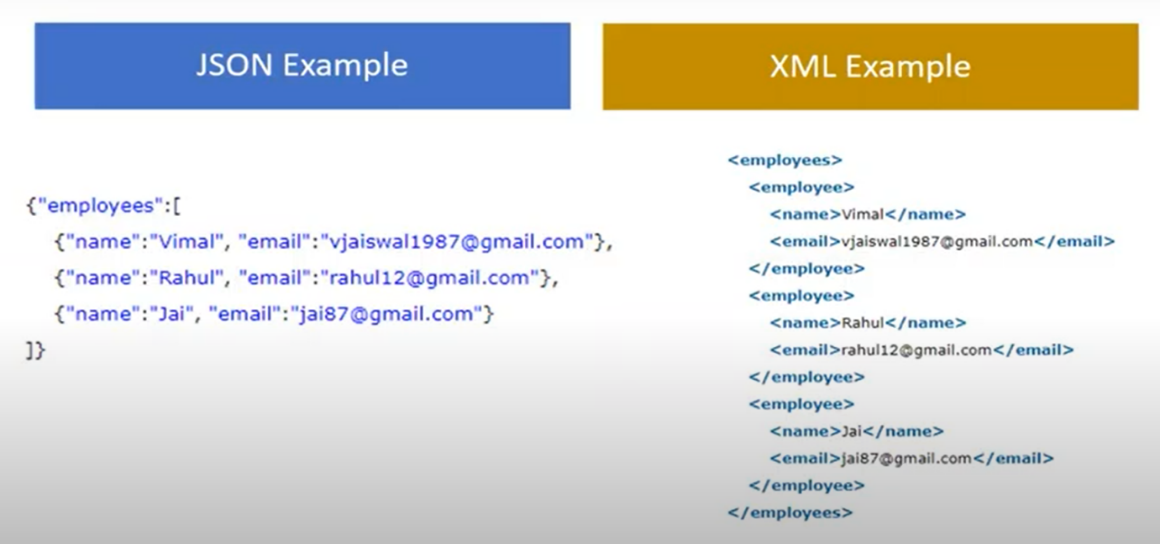
1. What is JSON?  
   🡪JSON – Java Script Object Notation.  
   🡪JSON is a syntax for storing and exchanging data.  
   🡪This format was specified by Douglas Crockford.  
   🡪Basically it was designed for human – readable interchange.  
   🡪JSON is a text, written with Java Script Object Notation.  
   🡪It has been extended from the java script scripting language.  
   🡪JSON internet media type is application / JSON.
2. Uses of JSON?  
   🡪Helps you to transfer data from a server.  
   🡪JSON format helps transmit and serialize all types of structured data.  
   🡪Allows you to perform asynchronous data calls without the need to do a page refresh.  
   🡪It is widely used for java script based application, which includes browser extension and websites.  
   🡪You can transmit data between server and web application using JSON.  
   🡪We can use JSON with modern programming languages.  
   🡪It is used for writing Java Script – based applications that include browser addons.  
   🡪Web services and API’s used the JSON format to get public data.
3. Characteristics:  
   🡪JSON is easy to read and write.  
   🡪It is light – weight text based interchange format.  
   🡪JSON is language independent.
4. JSON Data Types:  
   🡪Number  
   🡪String  
   🡪Boolean  
   🡪Null  
   🡪Object  
     
     
   🡪Array  
   

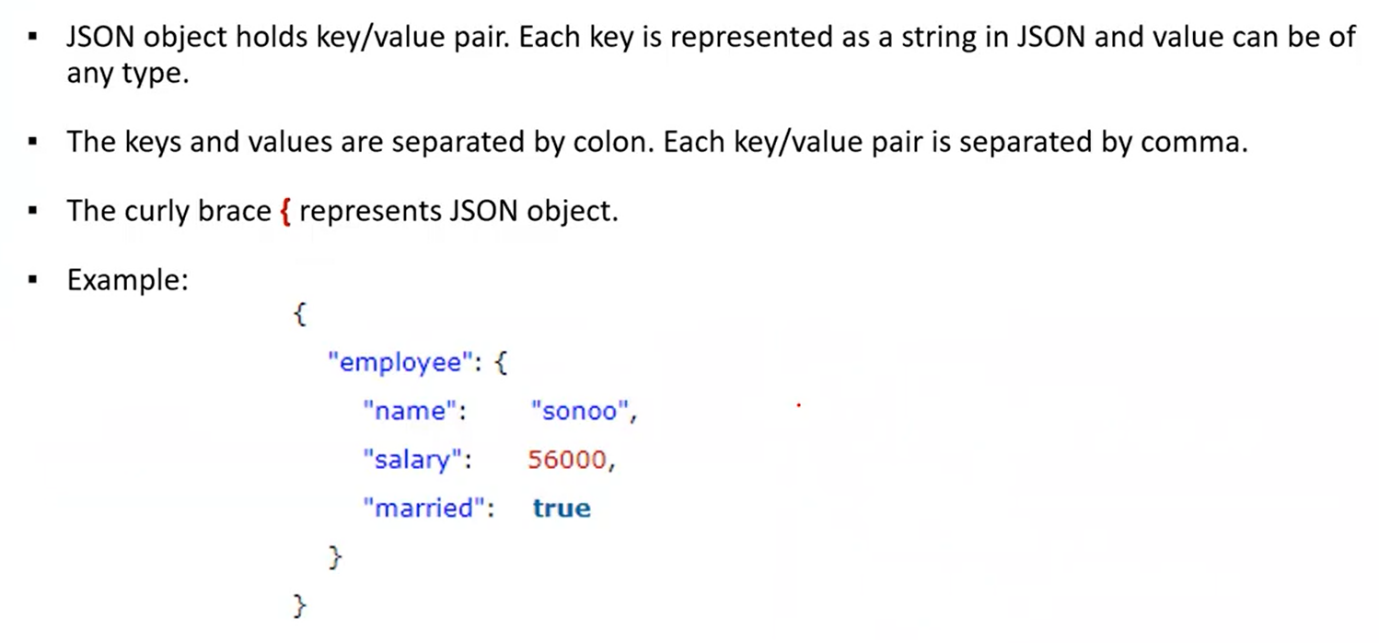
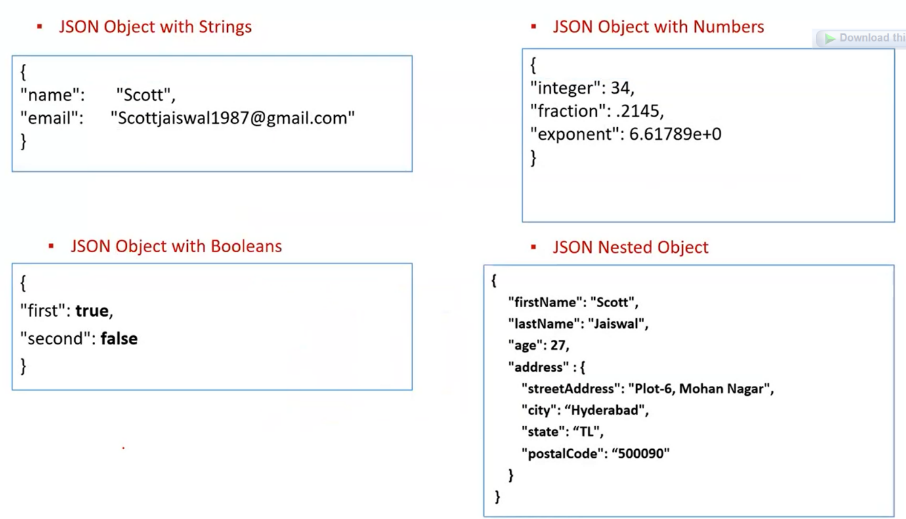


1. JSON Syntax:  
   🡪Data should be in name / value pairs.  
   🡪Data should be separated by commas.  
   🡪Curly braces should hold objects.  
   🡪Square brackets hold arrays.  
   

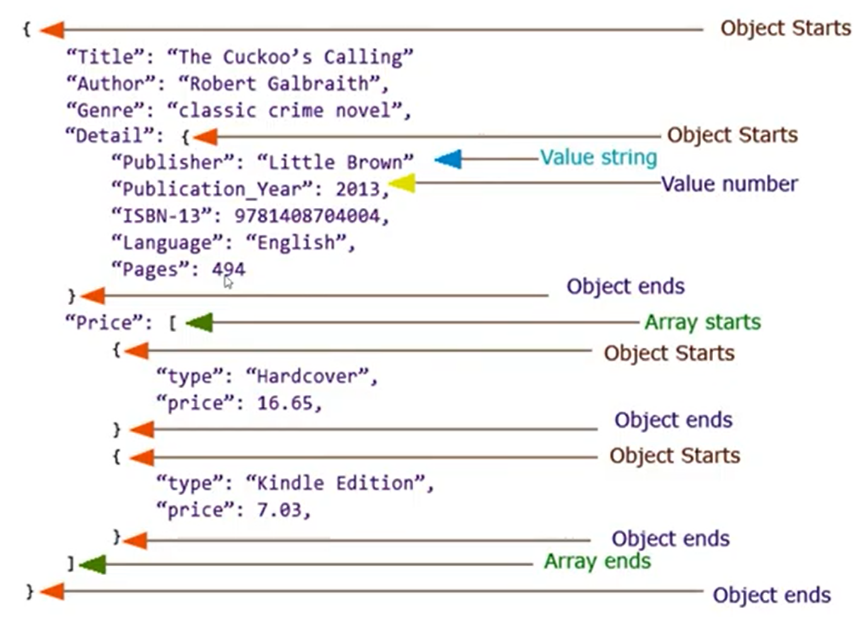
JSON Vs XML / JSON Object & JSON Array:

1. Similarities between JSON and XML?  
   🡪Both are simple and open.  
   🡪Both supports Unicode.  
   🡪Both represents self describing data.  
   🡪Both are language independent.  
     
   



1. JSON Object & JSON Array?  
   JSON Object:  
     
     
     
     
   JSON Array:  
   🡪JSON array represents ordered list of values.  
   🡪JSON array can store multiple values. It can be store string, number, Boolean or object in JSON array.  
   🡪In JSON array, values must be separated by comma.  
   🡪The ‘[‘ represents JSON array.  
     
   
2. Example:  
   

How to represent data in JSON:



Example:  


If we want to add some more data then put simply ‘,’ and add.

How to retrieve data from JSON using JSON path Expressions:

🡪JSON path is a query language for JSON, similar to XPath for XML.  
🡪A JSON path expression specifies a path to an element (or a set of elements) in a JSON structure.  
Paths can be use the dot notation:  
$.store.book[0].title  
or the bracket notation:  
$[‘store’][‘book’][0][‘title’]  
🡪$ represents the root object or array and can be omitted.  
for e.g., $foo.bar and foo.bar are same,  
and so are $[0].status and [0].status  
  
🡪JSON path expressions, including property names and values, are case sensitive.  
🡪Unlike X – path, JSON path does not have operations for accessing parent or sibling nodes from the given node.

🡪To validate JSON path:  
http://jsonpath.herokuapp.com/  
<https://jsonpath.com/>?

🡪To capture JSON path   
<https://jsonpathfinder.com/>

🡪JSON path finder plugin download to chrome:  
JSONPath Finder – Google Chrome

How to Read Data from JSON using JAVA:

Steps:

1. Create one maven project “jsonproject”.
2. Open pom.xml and add dependency “JSON.simple – 1.1.1”.
3. Create a folder “JSONFile” and under that create one file “employee.JSON”.  
     
   {

"id": "0001",

"type": "donut",

"name": "Cake",

"ppu": 0.55,

"batters":

{

"batter":

[

{ "id": "1001", "type": "Regular" },

{ "id": "1002", "type": "Chocolate" },

{ "id": "1003", "type": "Blueberry" },

{ "id": "1004", "type": "Devil's Food" }

]

},

"topping":

[

{ "id": "5001", "type": "None" },

{ "id": "5002", "type": "Glazed" },

{ "id": "5005", "type": "Sugar" },

{ "id": "5007", "type": "Powdered Sugar" },

{ "id": "5006", "type": "Chocolate with Sprinkles" },

{ "id": "5003", "type": "Chocolate" },

{ "id": "5004", "type": "Maple" }

]

}

1. Create a package “jsonproject” under src/test/java.
2. Create a class “GetValueFromJsonExample” under the package jsonproject.

**package** jsonproject;

**import** java.io.FileNotFoundException;

**import** java.io.FileReader;

**import** java.io.IOException;

**import** org.json.simple.JSONObject;

**import** org.json.simple.parser.JSONParser;

**import** org.json.simple.parser.ParseException;

**public** **class** GetValueFromJsonExample {

**public** **static** **void** main(String[] args) **throws** FileNotFoundException, IOException, ParseException {

//creating a JSON parser object

JSONParser jsonparse = **new** JSONParser();

//parsing the content of the json object

JSONObject jsonobject = (JSONObject) jsonparse.parse(**new** FileReader("C:\\Users\\ADMIN\\eclipse-workspace\\jsonproject\\JSONFile\\employee.json"));

//reading the data from the JSON file

String ID = (String) jsonobject.get("id");

String Firstname = (String) jsonobject.get("name");

System.***out***.println(ID);

System.***out***.println(Firstname);

}

}

🡪Now execute with Java application.