JSON

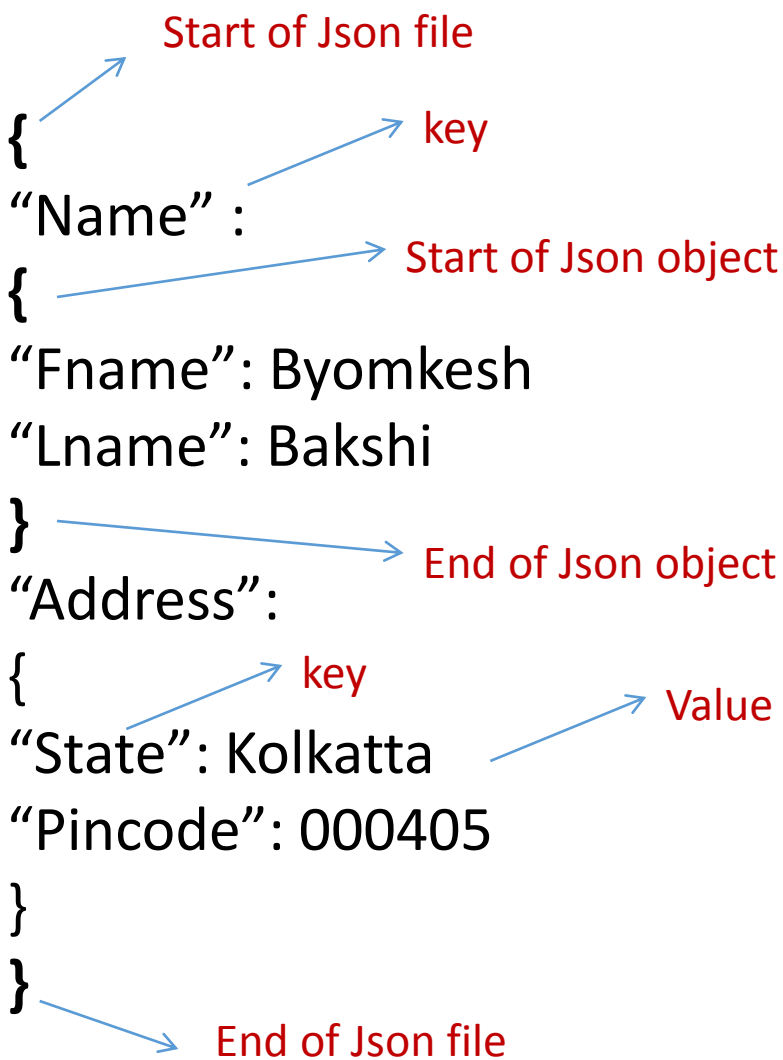
Introduction

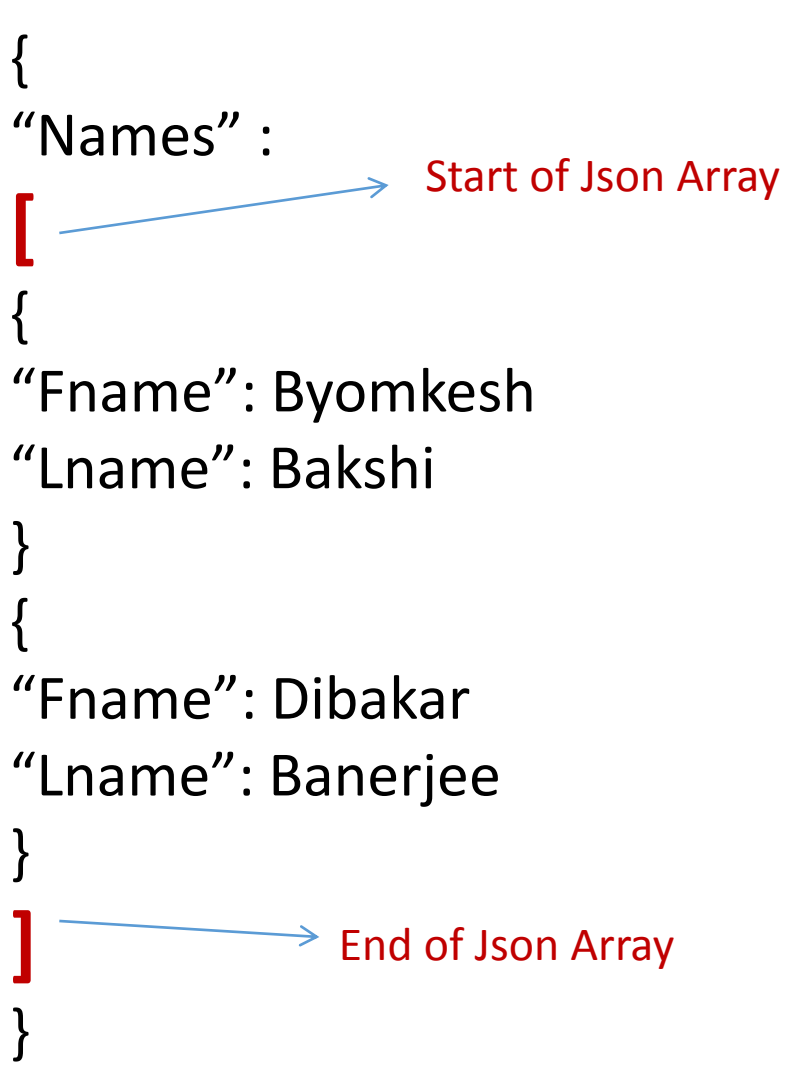
- JSON, or JavaScript Object Notation, is a minimal, readable format for **structuring data**.

- It is used primarily to transmit data between a **server** and **web application.**

- JSON is built **on two structures**:
  - A collection of **name/value pairs**. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array.
  - An ordered list of values. In most languages, this is realized as an **array, vector, list, or sequence**.

How does JSON data look like ?





GSON

Introduction

- Gson is an **Open Source project.**
- Gson is a Java library that can be used to convert Java Objects into their **JSON representation**.
- It can also be used to convert a **JSON string** to an equivalent Java object.
- Gson can work with arbitrary Java objects including **pre-existing objects.**
- For this purpose Gson provides several built in **serializers** and **deserializers**.

GSON

Usage

- Serializer allows to convert a Json string to corresponding Java type.
- Deserializers allows to convert from Java to a JSON representation.
- To use Gson in a **Gradle build**, add compile '**com.google.code.gson:gson:2.6.2**'or a later version
- Serialization Example :

```
String jsonData= new Gson().toJson(YourClassObjectHere);
```

- Deserialization Example

```
Gson gson = new Gson();
Type type = new TypeToken<YourClass>() {}.getType();
YourClassObjectHere = gson.fromJson(Util.jsonData, type);
```