Best practice of JSON:-

* JSON is a lightweight data-interchange format that is completely language independent.
* It was derived from JavaScript, but many modern programming languages include code to generate and parse JSON-format data
* The official Internet media type for JSON is application/json.
* It was designed for human-readable data interchange.
* The filename extension is .json

**1. Publish data using developer friendly JSON**

JSON [json] is the most popular format for publishing data through APIs; it is easy to parse, and it is supported natively in most programming languages.

Example JSON representing a Person

{

"name": "Barack Obama",

"givenName": "Barack",

"familyName": "Obama",

"jobTitle": "44th President of the United States"

}

**2. Use a top-level object**

JSON documents may be in the form of a object, or an array of objects. For most purposes, developers need a single entry point, so the JSON SHOULD be in the form of a single top-level object.

**3. Use native values**

When possible, property values SHOULD use native JSON datatypes such as numbers (integer, decimal and floating point) and booleans (true and false).

Note: JSON has a single numeric type, so using native representation of numbers can lose precision

**4. Assume arrays are unordered**

JSON specifies that the values in an array are ordered, however in many cases arrays are also used for values which are unordered.

**5. Use well-known identifiers when describing data**

By sticking to basic JSON data expression, and providing a JSON-LD Context, all keys used within a JSON document can have unambiguous meaning, as they bind to URLs which describe their meaning.

By adding an@context entry, the previous example can now be interpreted as JSON-LD.

Example 2: Example JSON-LD identifying a person

{

"@context": "http://schema.org",

"name": "Barack Obama",

"givenName": "Barack",

"familyName": "Obama",

"jobTitle": "44th President of the United States"

}

**6. Nest referenced inline objects**

When multiple related entity descriptions are provided inline, related entities SHOULD be nested.

Example: Nested relationships

{

"@context": "http://schema.org",

"id": "http://www.wikidata.org/entity/Q76",

"type": "Person",

"name": "Barack Obama",

"givenName": "Barack",

"familyName": "Obama",

"jobTitle": "44th President of the United States",

"spouse": {

"id": "http://www.wikidata.org/entity/Q13133",

"type": "Person",

"name": "Michelle Obama",

"spouse": "http://www.wikidata.org/entity/Q76"

}

}