# JavaScript Object Notation (JSON)

JSON is a **data interchange format**.

**Data interchange** simply refers to the process of transferring data between systems and applications etc. **Format** refers to the structure / syntax of the data.

JSON originates from JavaScript and uses the same syntax used in the JavaScript language. However, JSON is language independent and widely used in various programming environments, including Python, C++ and C#.

A major strength of the JSON format is that it is not only easy for humans to read and write, but also easy for machines to parse and generate.

JSON is built on two universal data structures:

1. **A Collection of Key/Value Pairs**

In JavaScript this is called an object (which is different from an object used in OOP in languages like C++ or C#). A JavaScript object is like a dictionary (in C# or Python), a map (in C++) or an associative array (in Bash). In other words, it is a collection of key/value pairs.

*\*Keys in JSON objects must be unique.*

{

"string": "Hello, World!",

"number": 123,

"object": { "key": "value" },

"array": [1, 2, 3],

"booleanTrue": true,

"booleanFalse": false,

"nullValue": null

}

1. **An Ordered List of Values**

In most languages, this is realised as an array (various languages), vector (C++), list (C# and Python) or sequence.

[

"apple",

"banana",

"cherry"

]

The data types supported by JSON include JSON objects and arrays, integers and floating-point values, strings (denoted by "double quotes"), Boolean values (`true` or `false` - these are not capitalised like in Python), or `null`.

JSON data is often stored in `.json` files, which are plain text files. The `.json` extension is used to identify the type of content it contains, which adheres to the JSON format.

The following are some libraries used to parse JSON in various programming languages:

* **Python Standard Library `json` Module**

import json

* **C++ `nlohmann/json` Library**

#include <nlohmann/json.hpp>

* **C# `System.Text.Json` Namespace**

using System.Text.Json;