JSON Review

JavaScript Object Notation

JSON

- JavaScript Object Notation (JSON) is a text format for the serialization of structured data.
  - Minimal
  - Textual
  - Subset of JavaScript

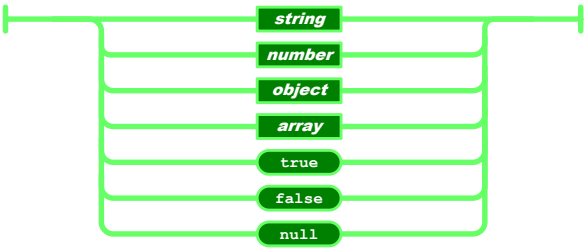
JSON

- A Subset of ECMA-262 Third Edition (ie, JavaScript)
- Features of JSON
  - Language Independent.
  - Text-based.
  - Light-weight.
  - Easy to parse.
- JSON can represent four primitive value types and two structured types

JSON Data Representations

- Primitive Value Types
  - Strings
  - Numbers
  - Booleans
  - **`null`**
- Structured Types
  - Objects
  - Arrays

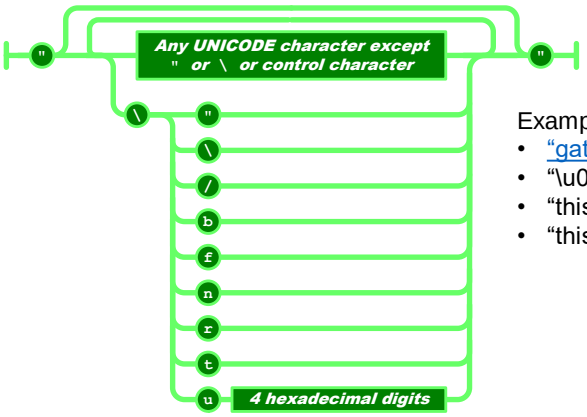
JSON Data “Railroad Diagram”



Strings

- Sequence of 0 or more Unicode characters
  - Ex: “” or “the string”
- No separate character type
  - A character is represented as a string with a length of 1
  - Ex: “a” is the character a
- **Enclosed** in "double quotes"
- Backslash escapement
  - Ex: “This string contains a double quote \””

String



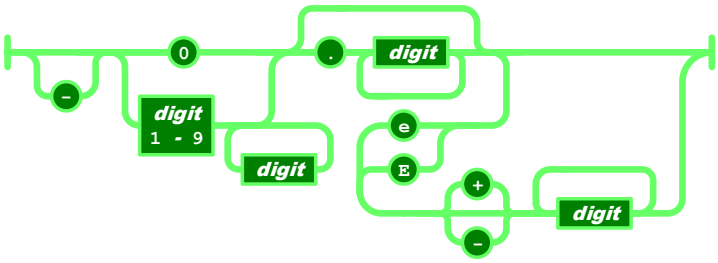
Examples:

- "gatesw@Microsoft.com"
- "\u0041"
- "this is a newline \n"
- "this is UNICODE"

Numbers

- Integer
- Real (contains decimal point)
- Scientific
- No octal or hex
- No **`NaN`** or **`Infinity`**
  - Use **`null`** instead

Number



Examples:
- 100
- 100.0
- 1.0E+2
- 1e-2

Booleans

- **`true`**
- **`false`**

`Note: these are lowercase`…

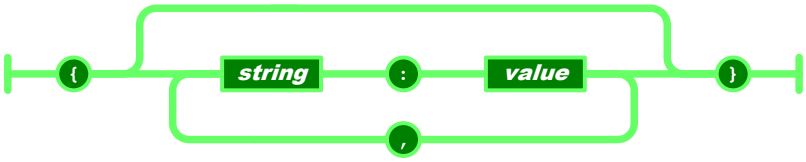
`null`

- A value that isn't anything.
- Like SQL, JSON's null is the absence of data

Object

- Objects are unordered containers of key/value pairs
- Objects are wrapped in { } (curly braces)
  - : (colon) separates keys and values
  - , (comma) separates key/value pairs
- Keys are double-quoted strings
- Values are one of the valid JSON value types

Object



Example:

```
{
    "Image": {
      "Width":  800,
      "Height": 600,
      "Title": "View from 15th Floor",
      "Thumbnail": {
        "Url":"http://www.e.com/image/48",
         "Height": 125,
         "Width":  100
      },
      "Animated" : false,
      "IDs": [116, 943, 234, 38793]
    }
}
```

Array

- Arrays are ordered sequences of values
- Arrays are enclosed in [] (square brackets)
- , (comma) separates values
- The JSON standard is silent on indexing
  - Ie, an implementation can start array indexing at 0 (Python) or 1 (Excel)

Array

Example:

```
["Sunday", "Monday", "Tuesday",
"Wednesday", "Thursday",
"Friday", "Saturday"]
```

```
[
    [0, -1, 0],
    [1, 0, 0],
    {“zero”:0, “one”:}
]
```

Arrays vs Objects

- Use objects when the key names are arbitrary strings.
  - Not to be confused with the term Associative Array (dictionaries in Python) from other languages
- Use arrays when the key names are sequential integers
  - Deserialization of JSON arrays is not language independent
- Deserialization of JSON data is not language independent
  - Varies from language to language
- Deserialization refers to the ingestion of JSON data types into the native data types of the language being used. When using Python:
  - a JSON Object becomes (as you might guess) a dictionary
  - a JSON Array becomes a list