

Working with JSON



Raphael Alampay

DEVELOPER

@happyalampay github.com/ralampay

Overview



What is JSON?

Reading from JSON Sources

- Text
- Files

Generating JSON

JSON Fundamentals

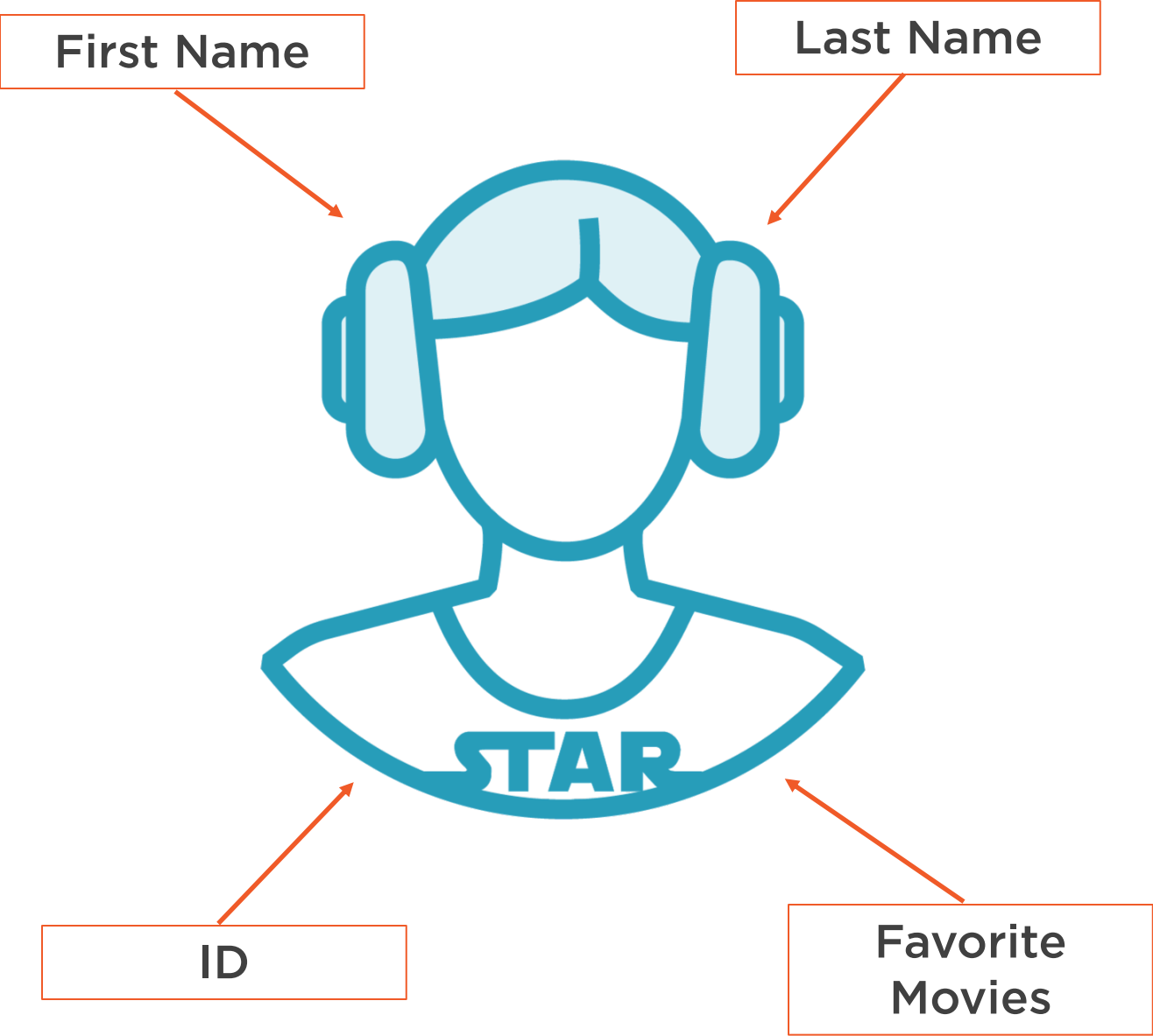
Javascript Object Notation

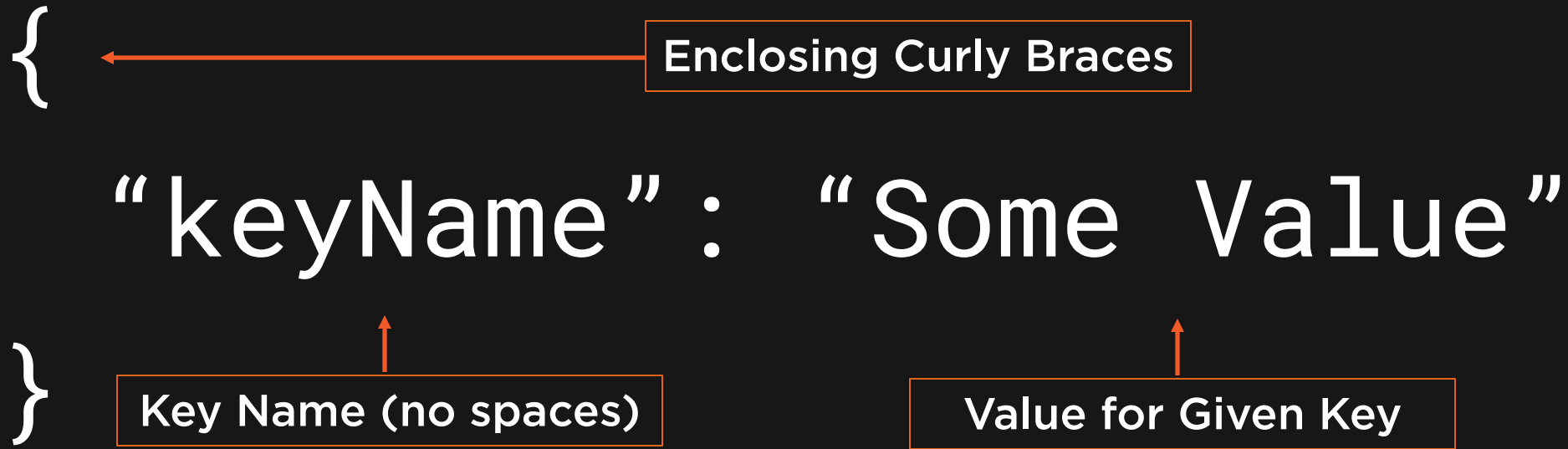
JSON files are just txt
files!

Represents complex
structured data using
key value pairs

What is JSON?

Model Data





The diagram illustrates the syntax of a JSON object. It shows a pair of curly braces enclosing a key-value pair. The opening brace is annotated with a box labeled "Enclosing Curly Braces". The key "keyName" is annotated with a box labeled "Key Name (no spaces)". The value "Some Value" is annotated with a box labeled "Value for Given Key".

```
{  
  "keyName": "Some Value"  
}
```

Syntax

Representing Data as an Object

```
{
```

```
  "id": 1,
```

Numerical Value



```
  "firstName": "Leah",
```

String Value

```
  "lastName": "Organa"
```

String Value

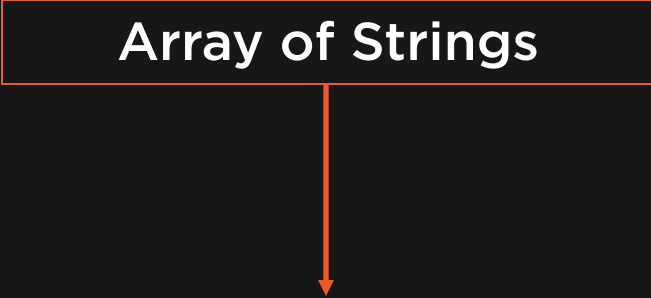
```
}
```

Syntax

Scalar Values: Numerical / String / Boolean

Each Key Value pair is separated by a “,”

```
{  
  "id": 1  
  "firstName": "Leah",  
  "lastName": "Organa",  
  "favoriteMovies": ["Star Wars", "Star Trek"]  
}
```



A diagram consisting of a rectangular box with an orange border containing the text "Array of Strings". A vertical orange arrow points downwards from the center of the box to the array value ["Star Wars", "Star Trek"] in the JSON object above.

Syntax

Array Values


```
{
```

```
  "id": 1
```

```
  "firstName": "Leah",
```

```
  "lastName": "Organa",
```

```
  "address": { "name": "Home", "location": "Alderaan" }
```

```
  "favoriteMovies": ["Star Wars", "Star Trek"]
```

```
}
```

Object with it's own set of key value pairs



Syntax

Objects as Values

```
{
```

```
  "id": 1
```

```
  "firstName": "Leah",
```

```
  "lastName": "Organa"
```

```
  "favoriteMovies": [ { "name": "Star Wars" }, { "name": "Star Trek" } ]
```

```
}
```

Objects as elements of an array



Syntax

Array of Objects

Demo



Exploring JSON

Reading JSON

Ways to Read JSON

Reading as Text

Treating JSON as simple Text

Programmatically

Representing JSON as an
actual data structure

```
require 'json'
```

```
content = '{"id": 1, "firstName": "Leah",  
"lastName": "Organa"}'
```

```
data = JSON.load(content)
```

```
puts "ID: #{data['id']}"
```

```
puts "First Name: #{data['first_name']}"
```

```
puts "Last Name: #{data['last_name']}"
```

◀ **Require the JSON library**

◀ **Content as string using valid JSON syntax**

◀ **Call JSON.load to parse the JSON string**

◀ **Access the attributes of the object via it's key**

```
require 'json'
```

```
content = '{"id": 1, "firstName": "Leah",  
"lastName": "Organa"}'
```

```
data = JSON.parse(content)
```

```
puts "ID: #{data['id']}"
```

```
puts "First Name: #{data['first_name']}"
```

```
puts "Last Name: #{data['last_name']}"
```

- ◀ Require the JSON library
- ◀ Content as string using valid JSON syntax
- ◀ Call **JSON.parse** to parse the JSON string
- ◀ Access the attributes of the object via it's key

What's the Difference (2.4.x)?

JSON.load

Takes in both a string or file

JSON.parse

Takes in a string only

JSON.load

Reading from File

sample.json

```
{  
  "id": 1  
  "firstName": "Leah",  
  "lastName": "Organa"  
}
```

```
require 'json'

data = JSON.load(File.new("sample.json"))

puts "ID: #{data['id']}"

puts "First Name: #{data['first_name']}"

puts "Last Name: #{data['last_name']}"
```

- ◀ Require the JSON library
- ◀ Content is loaded from a file
- ◀ Access the attributes of the object via it's key

```
require 'json'

content = File.read("sample.json")

data = JSON.parse(content)

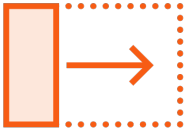
puts "ID: #{data['id']}"

puts "First Name: #{data['first_name']}"

puts "Last Name: #{data['last_name']}"
```

- ◀ Require the JSON library
- ◀ Read the contents of a file
- ◀ Parse the content
- ◀ Access the attributes of the object via it's key

Things to Keep in Mind



Have a different facility for reading a file before loading content as JSON



JSON can get complex so be sure to know the structure beforehand



JSON.load and JSON.parse does the exact same thing but JSON.parse is focused on just handling a string

Demo



Read a JSON file using Ruby

Generating JSON

```
require 'json'

hash = {
  id: 1,
  firstName: "Leah",
  lastName: "Organa"
}

content = hash.to_json
data = JSON.parse(content)

puts "ID: #{data['id']}"
puts "First Name: #{data['firstName']}"
puts "Last Name: #{data['lastName']}"
```

- ◀ Require the JSON library
- ◀ Some data represented as a hash
- ◀ Use **.to_json** to convert a hash to a JSON string before parsing with **JSON.parse**
- ◀ Access the values according to key

```
require 'json'

hash = {
  id: 1,
  firstName: "Leah",
  lastName: "Organa"
}

content = JSON.dump(hash)
data = JSON.parse(content)

puts "ID: #{data['id']}"
puts "First Name: #{data['firstName']}"
puts "Last Name: #{data['lastName']}"
```

- ◀ Require the JSON library
- ◀ Some data represented as a hash
- ◀ Use **JSON.dump** to convert a hash to a JSON string before parsing with **JSON.parse**
- ◀ Access the values according to key

What's the Difference?

.to_json

General Usecases

JSON.dump

Can take a limit argument

```
require 'json'

hash = {
  id: 1,
  firstName: "Leah",
  lastName: "Organa"
}

content = hash.to_json

File.write("test.json", content)
```

- ◀ Require the JSON library
- ◀ Some data represented as a hash
- ◀ Use **.to_json** to convert a hash to string
- ◀ Write the data to a file

```
{ "id": 1, "firstName": "Leah", "lastName": "Organa" }
```

Output of JSON string to File

Dumped to a single line

```
require 'json'

hash = {
  id: 1,
  firstName: "Leah",
  lastName: "Organa"
}

content = JSON.pretty_generate(hash)

File.write("test.json", content)
```

- ◀ Require the JSON library
- ◀ Some data represented as a hash
- ◀ Use **JSON.pretty_generate** to convert a hash to a formatted string
- ◀ Write the data to a file

```
{  
    "id": 1,  
    "firstName": "Leah",  
    "lastName": "Organa"  
}
```

Output of JSON String to File

Proper indentation and new lines

Demo



How to get from Hashes to JSON

Summary



JSON for Complex Structure

Simple Key-Value Pair Syntax

Practice: Data as JSON