

json2puml

Make data visible and understandable

Jens Fudickar, April 2022

API's and DATA

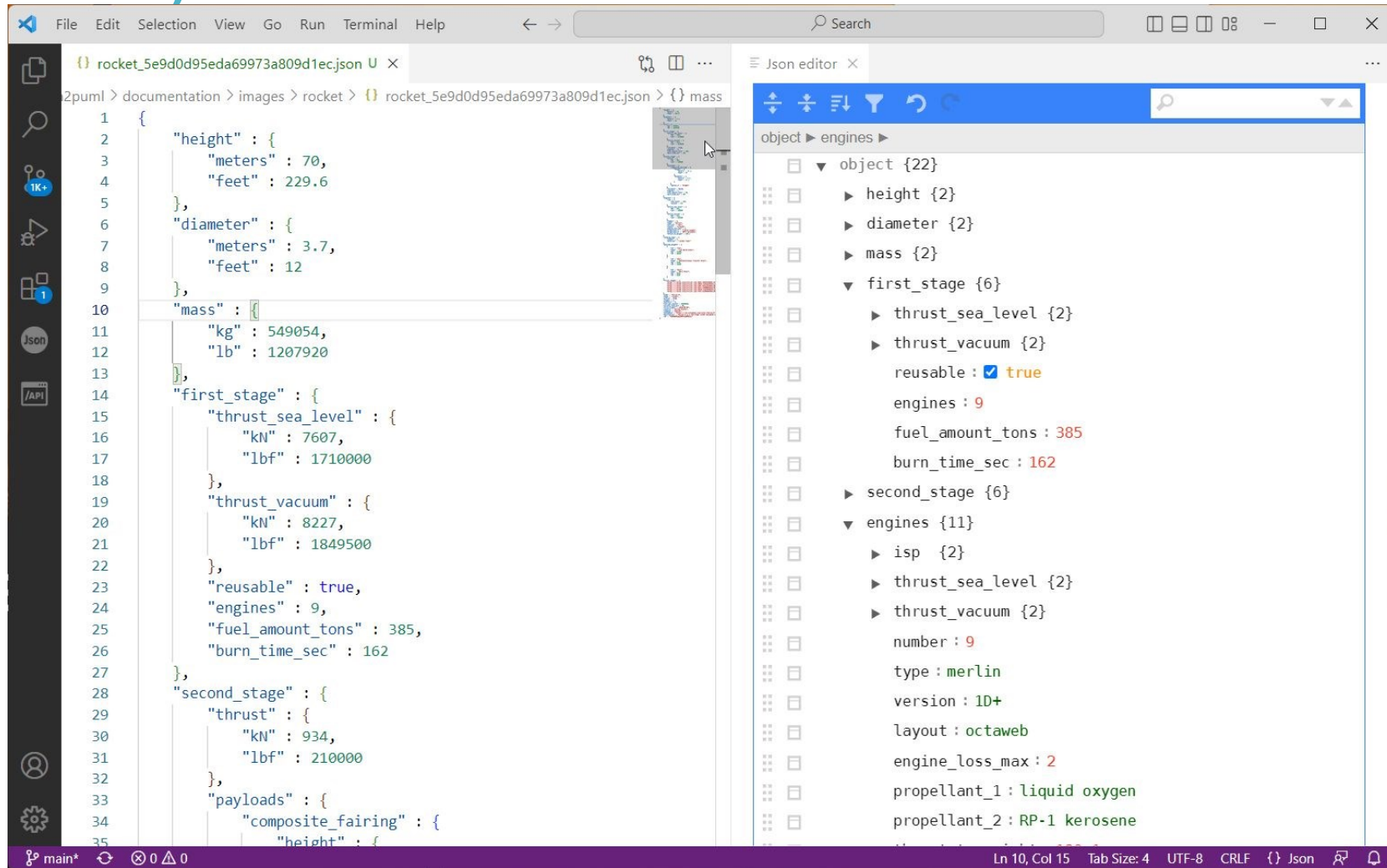
Working with API's leads to working with JSON data

A typical answer of a REST API call can look like this:

```
{
  "height": {
    "meters": 70,
    "feet": 229.6
  },
  "diameter": {
    "meters": 3.7,
    "feet": 12
  },
  "mass": {
    "kg": 549054,
    "lb": 1207920
  },
  "first_stage": {
    "thrust_sea_level": {
      "kN": 7607,
      "lbf": 1710000
    },
    "thrust_vacuum": {
      "kN": 8227,
      "lbf": 1849500
    },
    "reusable": true,
    "engines": 9,
    "fuel_amount_tons": 385,
    "burn_time_sec": 162
  },
  "second_stage": {
    "thrust": {
      "kN": 934,
      "lbf": 210000
    },
    "payloads": {
      "composite_fairing": {
        "height": {
          "meters": 13.1,
          "feet": 43
        },
        "diameter": {
          "meters": 5.2,
          "feet": 17.1
        }
      },
      "option_1": "dragon",
      "reusable": false,
      "engines": 1,
      "fuel_amount_tons": 90,
      "burn_time_sec": 397,
      "engines": {
        "isp": {
          "sea_level": 288,
          "vacuum": 312
        },
        "thrust_sea_level": {
          "kN": 845,
          "lbf": 190000
        },
        "thrust_vacuum": {
          "kN": 914,
          "lbf": 205500
        },
        "number": 9,
        "type": "merlin",
        "version": "1D+",
        "layout": "octaweb",
        "engine_loss_max": 2,
        "propellant_1": "liquid oxygen",
        "propellant_2": "RP-1 kerosene",
        "thrust_to_weight": 180.1,
        "landing_legs": {
          "number": 4,
          "material": "carbon fiber"
        },
        "payload_weights": [
          {
            "id": "leo",
            "name": "Low Earth Orbit",
            "kg": 22800,
            "lb": 50265
          },
          {
            "id": "gto",
            "name": "Geosynchronous Transfer Orbit",
            "kg": 8300,
            "lb": 18300
          },
          {
            "id": "mars",
            "name": "Mars Orbit",
            "kg": 4020,
            "lb": 8860
          }
        ],
        "flickr_images": [
          "https://farm1.staticflickr.com/929/28787338307_3453a11a77_b.jpg",
          "https://farm4.staticflickr.com/3955/32915197674_eee74d81bb_b.jpg",
          "https://farm1.staticflickr.com/293/32312415025_6841e30bf1_b.jpg",
          "https://farm1.staticflickr.com/623/23660653516_5b6cb301d1_b.jpg",
          "https://farm6.staticflickr.com/5518/31579784413_d853331601_b.jpg",
          "https://farm1.staticflickr.com/745/32394687645_a9c54a34ef_b.jpg"
        ],
        "name": "Falcon 9",
        "type": "rocket",
        "active": true,
        "stages": 2,
        "boosters": 0,
        "cost_per_launch": 50000000,
        "success_rate_pct": 98,
        "first_flight": "2010-06-04",
        "country": "United States",
        "company": "SpaceX",
        "wikipedia": "https://en.wikipedia.org/wiki/Falcon_9",
        "description": "Falcon 9 is a two-stage rocket designed and manufactured by SpaceX for the reliable and safe transport of satellites and the Dragon spacecraft into orbit.",
        "id": "5e9d0d95eda69973a809d1ec"
      }
    }
  }
}
```

How to visualize / understand this ?

(Online) JSON Editor

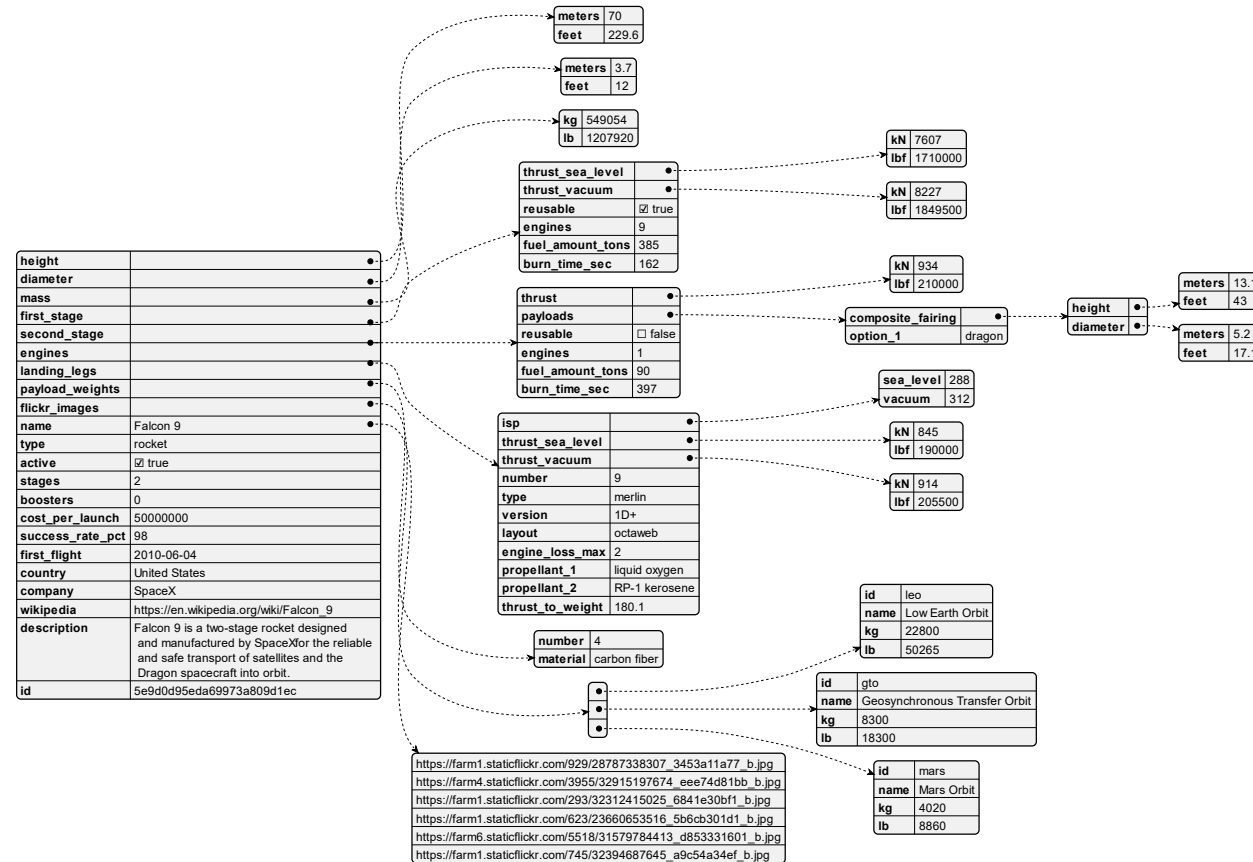


This makes the live easier but it is still complex 😊

Confluence and Standard PlantUML

PlantUML supports an OOTB visualisation of JSON.

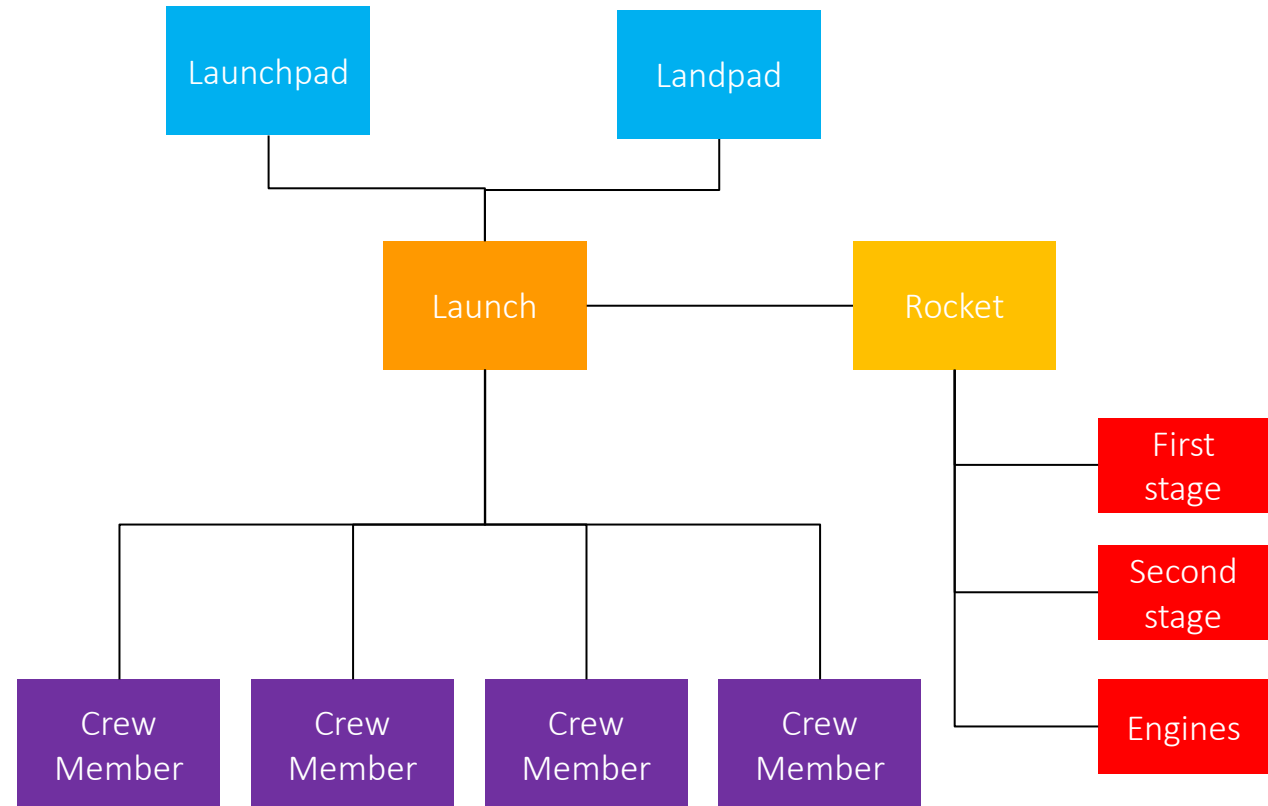
```
@startjson
<json>
@endjson
```



This improves also, but there is no knowledge about the data models behind

Drawing by Hand

It's nice, and it's giving you a task.
But it's not effective, not accurate
and not fast enough.



One Solution: json2puml

Json2puml is a command line tool developed to generate PlantUML files based JSON files (TMF based).

Json2puml has an understanding of how data is structured (in TMF) and simplifies and visualises the outcome.

json2puml has the possibility to combine the JSON results of multiple API calls into one result set.

json2puml is highly configurable to generate outcomes in different detailed levels.

json2puml is free to use for everyone.

SpaceX REST API

Open Source REST API for launch, rocket, core, capsule, starlink, launchpad, and landing pad data.

<https://github.com/r-spacex/SpaceX-API>

What is this?

It's an unofficial public API to receive various information about all Space X rocket launches.

Space X API – get /rocket

One API result formatted with three different options.

rocketrocket_5e9d0d95eda69973a809d1ec.compact.puml

R rocket Falcon 9	
rocket - 5e9d0d95eda69973a809d1ec	
attribute	value
id	5e9d0d95eda69973a809d1ec
name	Falcon 9
type	rocket
active	true
stages	2
boosters	0
cost per launch	50000000
success rate pct	98
first flight	2010-06-04
country	United States
company	SpaceX

json2puml	v2.0.7.59
Generated at	26.01.2023 00:50:08
Definition File	samples\spacex\spacex_definition.json
Input List File	samples\spacex\spacex_inputlist_launches.json
Definition Option	compact

Objectformat	hardware
--------------	----------

rocketrocket_5e9d0d95eda69973a809d1ec.compact.puml

Option : compact

E engines	
engines - engines_5e9d0d95eda69973a809d1ec_3	
attribute	value
number	9
type	merlin
version	1D+
nozzle	octothrust
engine_test_name_2	
propellant_1	liquid oxygen
propellant_2	RP-1 kerosene
thrust_sea_level	1800

thrust	
attribute	value
sea_level	289
vacuum	312

thrust_sea_level	
attribute	value
kg	845
lb	1895000

thrust_vacuum	
attribute	value
kg	914
lb	2055000

json2puml	v2.0.7.59
Generated at	26.01.2023 00:00:10
Definition File	samples\spacex\spacex_definition.json
Input List File	samples\spacex\spacex_inputlist_launches.json
Definition Option	default

Objectformat	Objectformat
hardware	hardware-details

Option : default

rocketrocket_5e9d0d95eda69973a809d1ec.default.puml

E engines	
engines - engines_5e9d0d95eda69973a809d1ec_3	
attribute	value
number	9
type	merlin
version	1D+
nozzle	octothrust
engine_test_name_2	
propellant_1	liquid oxygen
propellant_2	RP-1 kerosene
thrust_sea_level	1800

thrust	
attribute	value
sea_level	289
vacuum	312

thrust_sea_level	
attribute	value
kg	845
lb	1895000

thrust_vacuum	
attribute	value
kg	914
lb	2055000

F first stage	
first_stage - first_stage_5e9d0d95eda69973a809d1ec_1	
attribute	value
boosters	1
costs	10
cost amount	385
burn time sec	152

thrust	
attribute	value
sea_level	1717000
vacuum	1717000

thrust_sea_level	
attribute	value
kg	1717000
lb	1717000

thrust_vacuum	
attribute	value
kg	1717000
lb	1717000

S second stage	
second_stage - second_stage_5e9d0d95eda69973a809d1ec_2	
attribute	value
boosters	1
costs	10
cost amount	385
burn time sec	107

thrust	
attribute	value
sea_level	1717000
vacuum	1717000

thrust_sea_level	
attribute	value
kg	1717000
lb	1717000

thrust_vacuum	
attribute	value
kg	1717000
lb	1717000

P payload	
payload - payload_second_stage_5e9d0d95eda69973a809d1ec_1	
attribute	value
boosters	1
costs	10
cost amount	385
burn time sec	107

thrust	
attribute	value
sea_level	1717000
vacuum	1717000

thrust_sea_level	
attribute	value
kg	1717000
lb	1717000

thrust_vacuum	
attribute	value
kg	1717000
lb	1717000

C composite stage	
composite_fairing - composite_fairing_payload_second_stage_5e9d0d95eda69973a809d1ec_1_1	
attribute	value
height	12.1
diameter	3.7
mass	12.1
volume	12.1

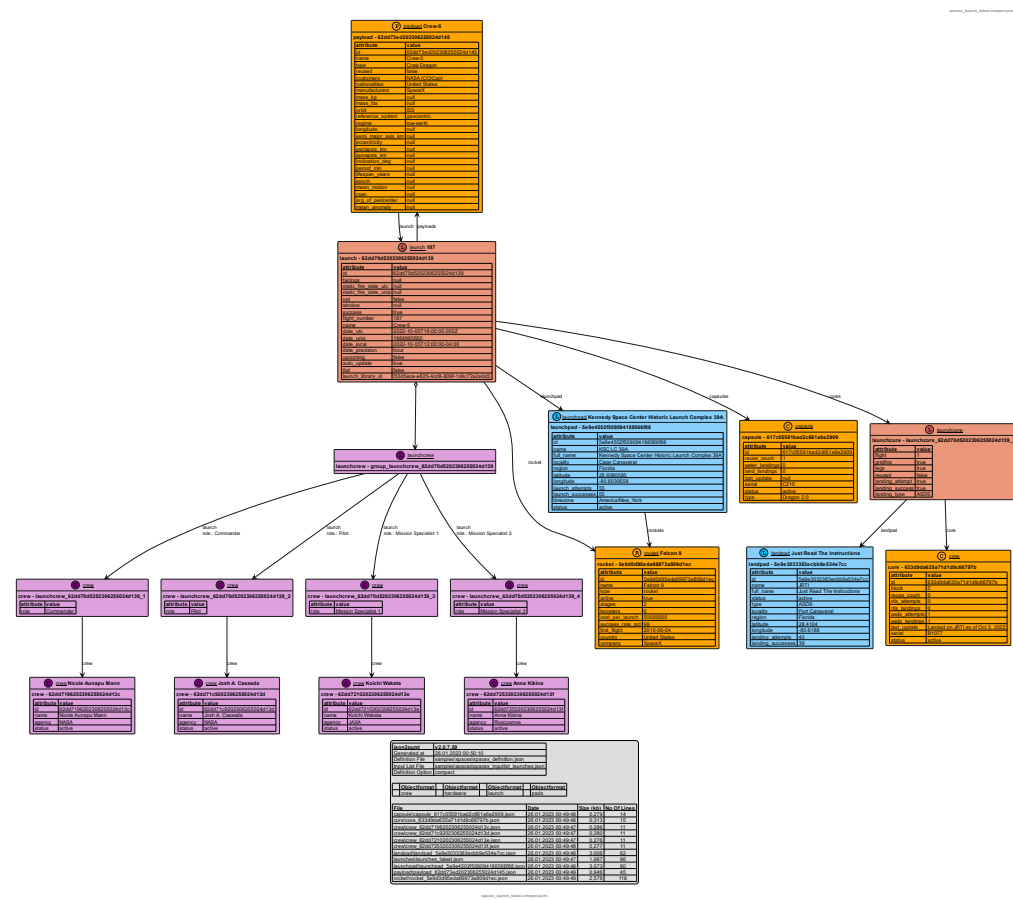
json2puml	v2.0.7.59
Generated at	26.01.2023 23:26:32
Definition File	samples\spacex\spacex_definition.json
Input List File	samples\spacex\spacex_inputlist_launches.json
Definition Option	full

Objectformat	Objectformat
hardware	hardware-details

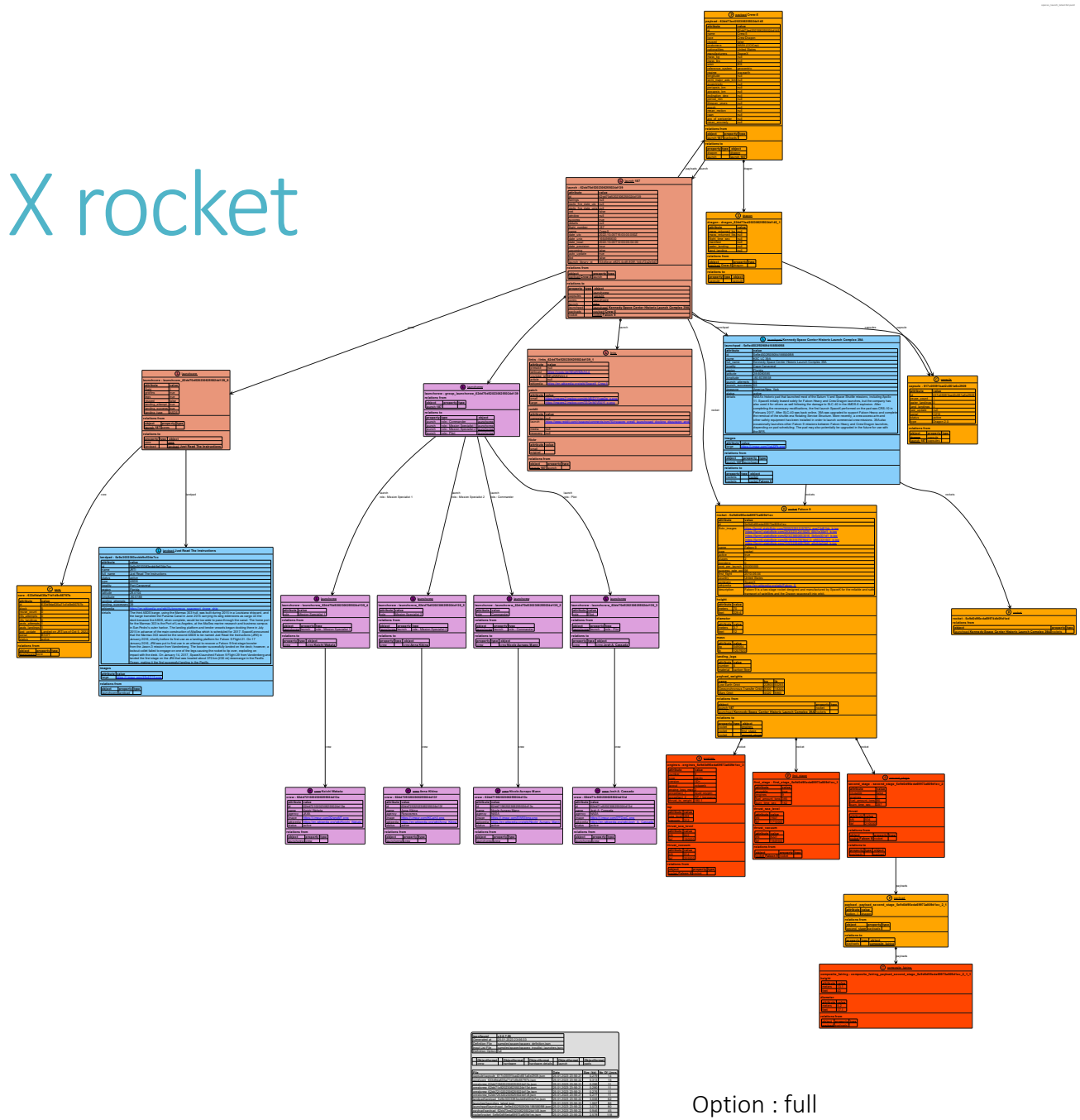
Option : full

rocketrocket_5e9d0d95eda69973a809d1ec.full.puml

All data regarding the latest launch of a Space X rocket



Option : compact



Option : full

SWAPI - The Star Wars API

<https://swapi.dev/>

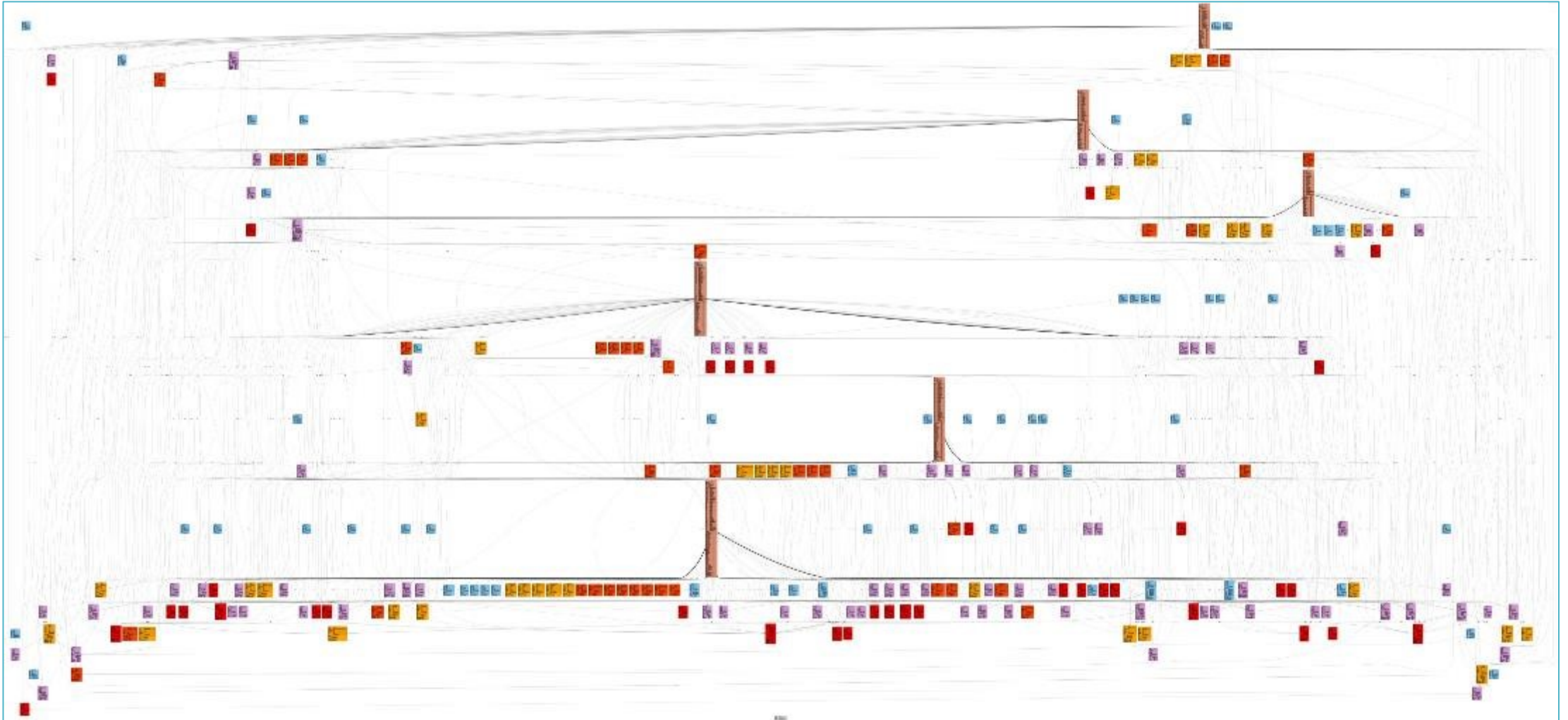
What is this?

The Star Wars API, or "swapi" (Swah-pee) is the world's first quantified and programmatically-accessible data source for all the data from the Star Wars canon universe!

We've taken all the rich contextual stuff from the universe and formatted into something easier to consume with software. Then we went and stuck an API on the front so you can access it all!


All Data

6 Films, 82 Persons, 60 Planets, 37 Species, 36 Star ships, 39 Vehicles



Examples

output:summary/full/summary.full.deathstar.puml


films
A New Hope

attribute	value
title	A New Hope
episode_id	4
director	George Lucas
producer	Gary Kurtz, Rick McCallum
release_date	1977-05-25
created	2014-12-10T14:23:31.880000Z
edited	2014-12-20T19:49:45.256000Z
uri	https://swapi.dev/api/films/1/

Relations From

object	property	type
starships	Death Star	films

Relations To

property	type	object
starships		starships Death Star

starships films

S

starships

Death Star

attribute	value
name	Death Star
model	DS-1 Orbital Battle Station
manufacturer	Imperial Department of Military Research, Sienar Fleet Systems
cost_in_credits	1000000000000
length	120000
max_atmosphering_speed	n/a
crew	342,963
passengers	843,342
cargo_capacity	1000000000000
consumables	3 years
hyperdrive_rating	4.0
MGLT	10
starship_class	Deep Space Mobile Battlesation
created	2014-12-10T16:36:50.509000Z
edited	2014-12-20T21:26:24.783000Z
url	https://swapi.dev/api/starships/9/

Relations From

object	property	type
films	A New Hope	starships

Relations To

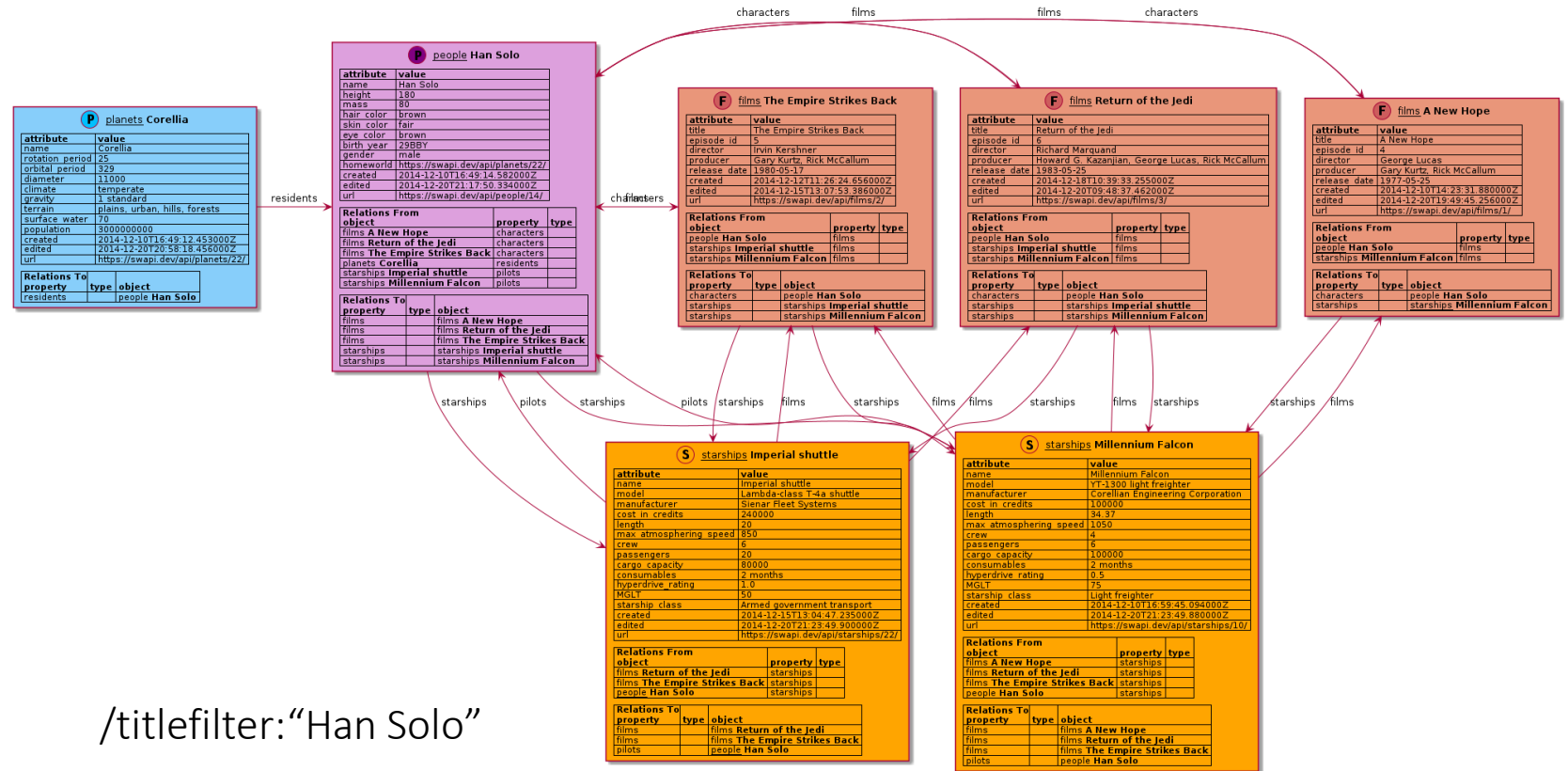
property	type	object
films		films A New Hope

json2puml		v1.1.10.20	
Generated at	14.04.2022 11:21:55		
Definition File	swapidefinition.json		
Input File	output:summary/full/summary.full.deathstar.json		
Definition Option	full		
Title Filter	Death Star		
Objectformat			
films			
starships			
File	Date	Size (kb)	No Of Lines
swapi-films.json	20.03.2022 17:08:22	21.087	506
swapi-people.json	20.03.2022 17:08:37	66.059	1900
swapi-planets.json	20.03.2022 17:08:43	39.858	1151
swapi-species.json	20.03.2022 17:08:47	29.398	833
swapi-starships.json	20.03.2022 17:08:52	30.779	866
swapi-vehicles.json	20.03.2022 17:08:57	28.724	820

output:summary/full/summary.full.deathstar.puml

/titlefilter:“Death Star”

output:summary/full/summary.full.hansolo.puml



/titlefilter:“Han Solo”

json2puml		v1.1.10.20	
Generated at	14.04.2022 11:21:51		
Definition File	swapidefinition.json		
Input File	output:summary/full/summary.full.hansolo.json		
Definition Option	full		
Title Filter	Han Solo		
Objectformat			
films			
people			
planets			
starships			
File	Date	Size (kb)	No Of Lines
swapi-films.json	20.03.2022 17:08:22	21.087	506
swapi-people.json	20.03.2022 17:08:37	66.059	1900
swapi-planets.json	20.03.2022 17:08:43	39.858	1151
swapi-species.json	20.03.2022 17:08:47	29.398	833
swapi-starships.json	20.03.2022 17:08:52	30.779	866
swapi-vehicles.json	20.03.2022 17:08:57	28.724	820

output:summary/full/summary.full.hansolo.puml

How is it working

json2puml

Has three main components which can be combined

1. Fetch data from a defined set of API's using curl. (Optional)
2. Convert the JSON data into a PlantUml script.
3. Convert the PlantUml script into a SVG or PNG file using the PlantUml jar file.

json2puml

can automate the fetching of data by using data from an API result as an input criteria for the next API call.

E.g. Using the SpaceX API the `get /launch` result includes the ID of the used rocket. This ID can be used for the next `get /rocket` call.

Operating Systems

json2puml can be used on MS Windows and on Linux.
For Linux there are plain executables and docker container available.

json2puml can run as command line tool or as a service application answering on http based REST calls.

Where can you find it:

It's published on Github.com under a GPLv3 license.

<https://github.com/jfudickar/json2puml>