

json2puml

Make data visible and understandable

Jens Fudickar, April 2022

API's and DATA

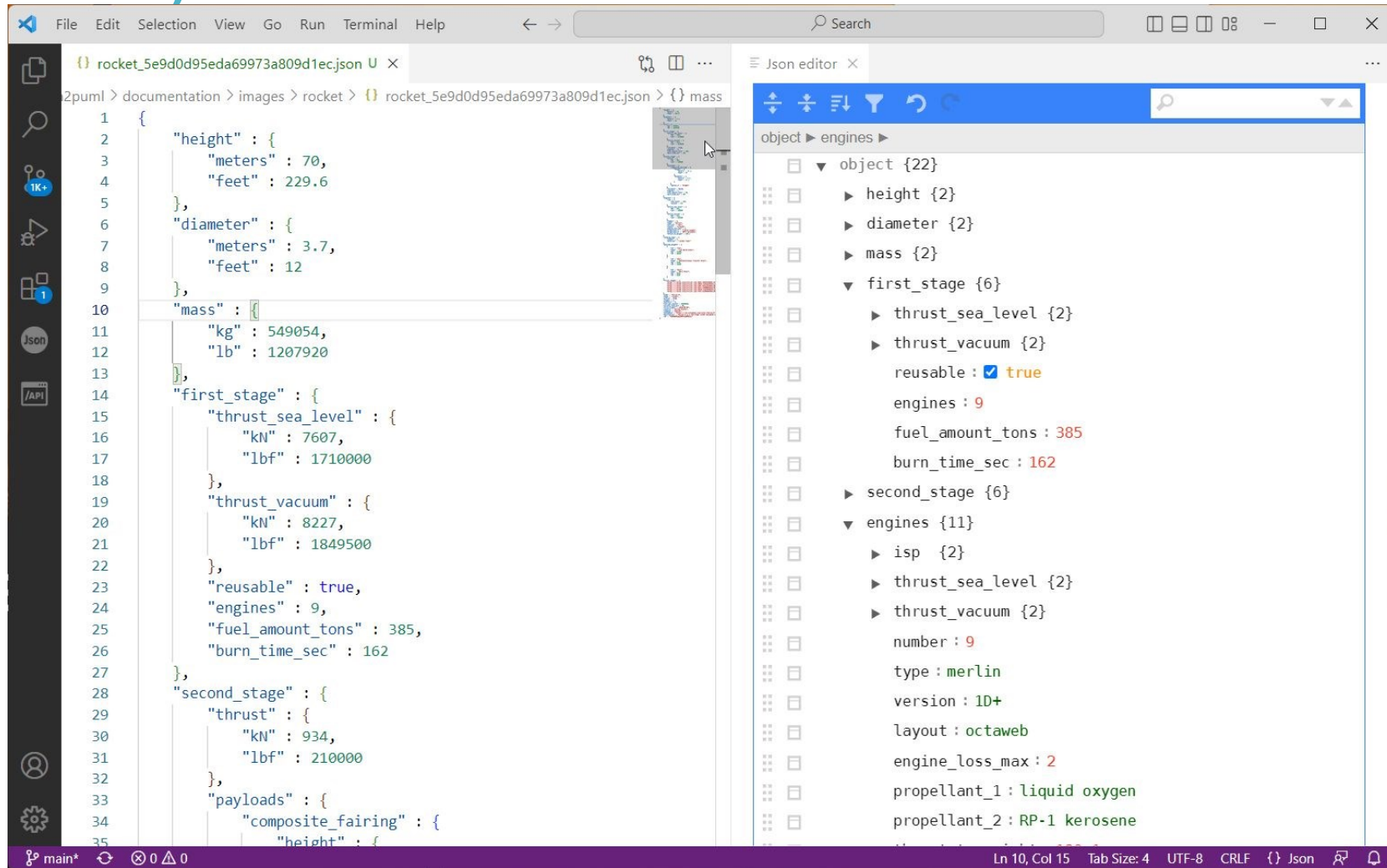
Working with API's leads to working with JSON data

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

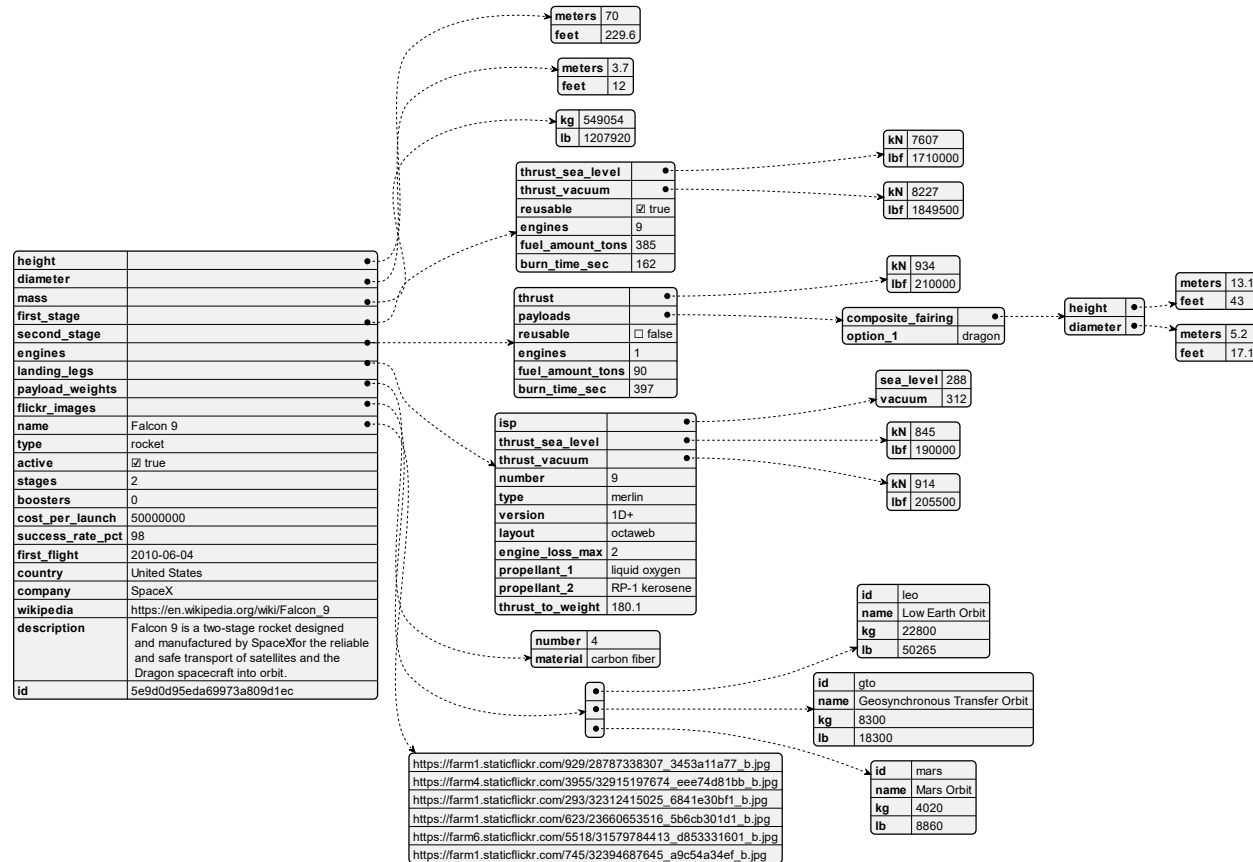


This makes the live easier but it is still complex 😊

Confluence and Standard PlantUML

PlantUML supports an OOTB visualisation of JSON.

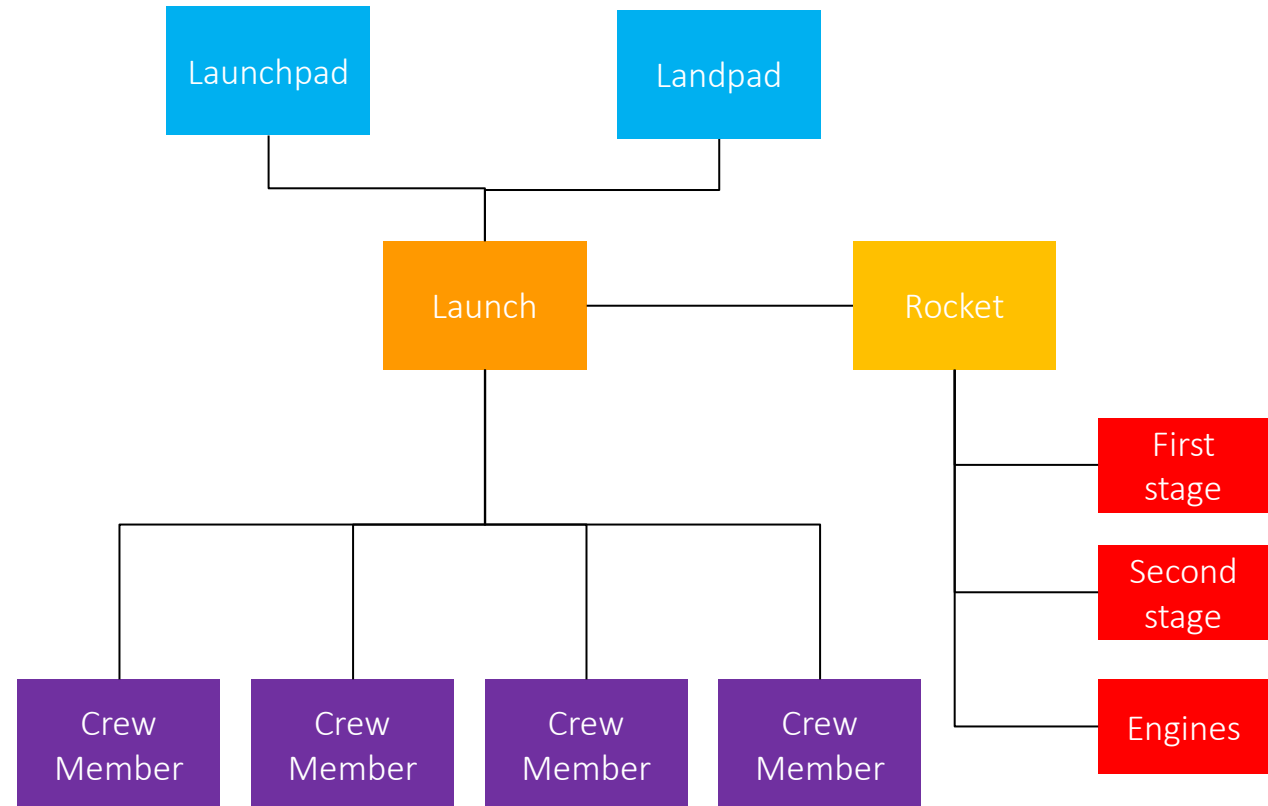
```
@startjson
<json>
@endjson
```



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** allows to convert JSON data into PlantUML based diagrams.
- Json2puml** has an understanding of how data is structured 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.

rocketrocket_5e9d0d95eda69973a809d1ec.compact.puml

rocket\rocket_5e9d0d95eda69973a809d1ec.compact.puml

rockefrocket_5e3d3d35eda59373a809d1ec.default.puml

rockefrocket_5e3d3d35eda59373a809d1ec.default.puml

json2puml	v2.0.7.59
Generated at	26.01.2023 00:00:10
Definition File	samples\space\space_x_definition.json
Input List File	samples\space\space_x_inputlist_launches.json
Definition Option	default

Objectformat	Objectformat
hardware	hardware_details

```

engine = engine, # 9600/5000/3000/2000/1500/1000/500/250/125/62.5/31.25/15.625/7.8125/3.90625/1.953125/0.9765625/0.48828125/0.244140625/0.1220703125/0.06103515625/0.030517578125/0.0152587890625/0.00762939453125/0.003814697265625/0.0019073486328125/0.00095367431640625/0.000476837158203125/0.0002384185791015625/0.00011920928955078125/0.000059604644775390625/0.0000298023223876953125/0.00001490116119384765625/0.000007450580596923828125/0.0000037252902984619140625/0.00000186264514923095703125/0.000000931322574615478515625/0.0000004656612873077392578125/0.00000023283064365386962890625/0.000000116415321826934814453125/0.0000000582076609134674072265625/0.00000002910383045673370361328125/0.000000014551915228366851806640625/0.0000000072759576141834259033203125/0.00000000363797880709171295166015625/0.000000001818989403545856475830078125/0.0000000009094947017729282379150390625/0.00000000045474735088646411895751953125/0.000000000227373675443232059478759765625/0.0000000001136868377216160297393798828125/0.00000000005684341886080801486968994140625/0.000000000028421709430404007434844970703125/0.0000000000142108547152020037174224853515625/0.00000000000710542735760100185871124267578125/0.000000000003552713678800500929355621337890625/0.0000000000017763568394002504646778106689453125/0.00000000000088817841970012523233890533347265625/0.000000000000444089209850062616169452666736328125/0.0000000000002220446049250313080847263333681640625/0.00000000000011102230246251565404236316668408203125/0.000000000000055511151231257827021181583334041015625/0.0000000000000277555756156289135105907916670205078125/0.00000000000001387778780781445675529539583351025390625/0.0000000000000069388939039072283776476979167551025390625/0.000000000000003469446951953614188823848958377551025390625/0.00000000000000173472347597680709441192447918877551025390625/0.0000000000000008673617379883403547059622395943877551025390625/0.000000000000000433680868994170177352981119797193877551025390625/0.00000000000000021684043449708508867649055989859693877551025390625/0.0000000000000001084202172485425443382452799492984693877551025390625/0.000000000000000054210108624271272169122639974649234693877551025390625/0.00000000000000002710505431213563608456131998732461734693877551025390625/0.0000000000000000135525271560678180422806599936623086734693877551025390625/0.000000000000000006776263578033909021140329996831154336734693877551025390625/0.00000000000000000338813178901695451057016499841557716836734693877551025390625/0.0000000000000000016940658945084772552850824997077885841836734693877551025390625/0.000000000000000000847032947254238627642541249853894292091836734693877551025390625/0.00000000000000000042351647362711931382127062492694714604591836734693877551025390625/0.0000000000000000002117582368135596569106353124634735730229591836734693877551025390625/0.000000000000000000105879118406779828455317656233173786511479591836734693877551025390625/0.00000000000000000005293955920338991422765882811658689255573979591836734693877551025390625/0.00000000000000000002646977960169495711382941405829346277869899591836734693877551025390625/0.000000000000000000013234889800847478556914707029146731389349499591836734693877551025390625/0.000000000000000000006617444900423739278457353514573656946724999591836734693877551025390625/0.0000000000000000000033087224502118696392286767572868284733624999591836734693877551025390625/0.0000000000000000000016543612251059348196143383786434142168124999591836734693877551025390625/0.00000000000000000000082718061255296740980716918932170710840624999591836734693877551025390625/0.000000000000000000000413590306276483704903584594660853554203124999591836734693877551025390625/0.0000000000000000000002067951531382418524517922973304267771015624999591836734693877551025390625/0.00000000000000000000010339757656912092612289614866504288885078124999591836734693877551025390625/0.000000000000000000000051698788284560463061448074332521444425390624999591836734693877551025390625/0.0000000000000000000000258493941422802315307240371662607222126953124999591836734693877551025390625/0.0000000000000000000000129246970711401157653620185
```

second_stage - second_stage_Se9d0d95e4d69973a809d1ec_2

attribute	value
reusable	false
engine	1
fuel amount tons	90
burn time sec	257

thrust

attribute	value
th	934
ref	2.100000

relations from

object	property type
object Falcon 9	rocket

relations to

property type	object
rocket	object Falcon 9

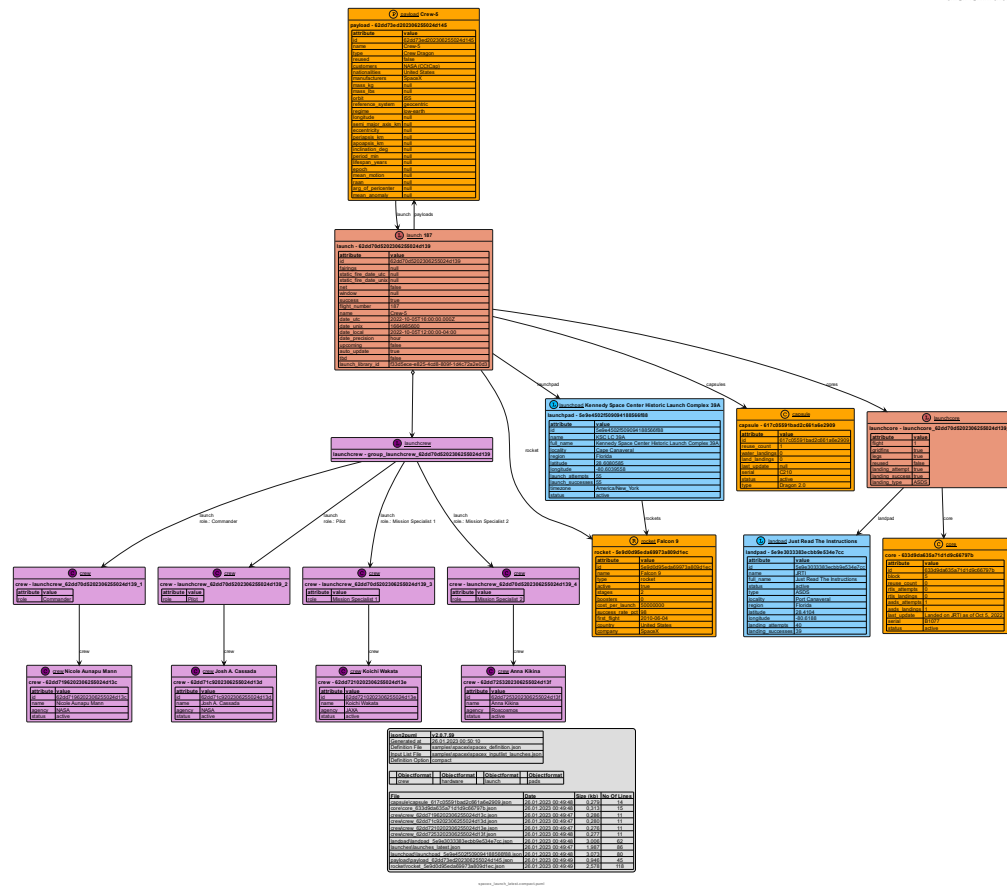
The diagram shows a database schema with two tables:

- composite_fairing**: Contains attributes *radius* (value 13.5), *meters* (value 63), and *height* (value 63).
- relations from**: Contains attributes *object* (value 1), *property* (value 1), and *value* (value 1).

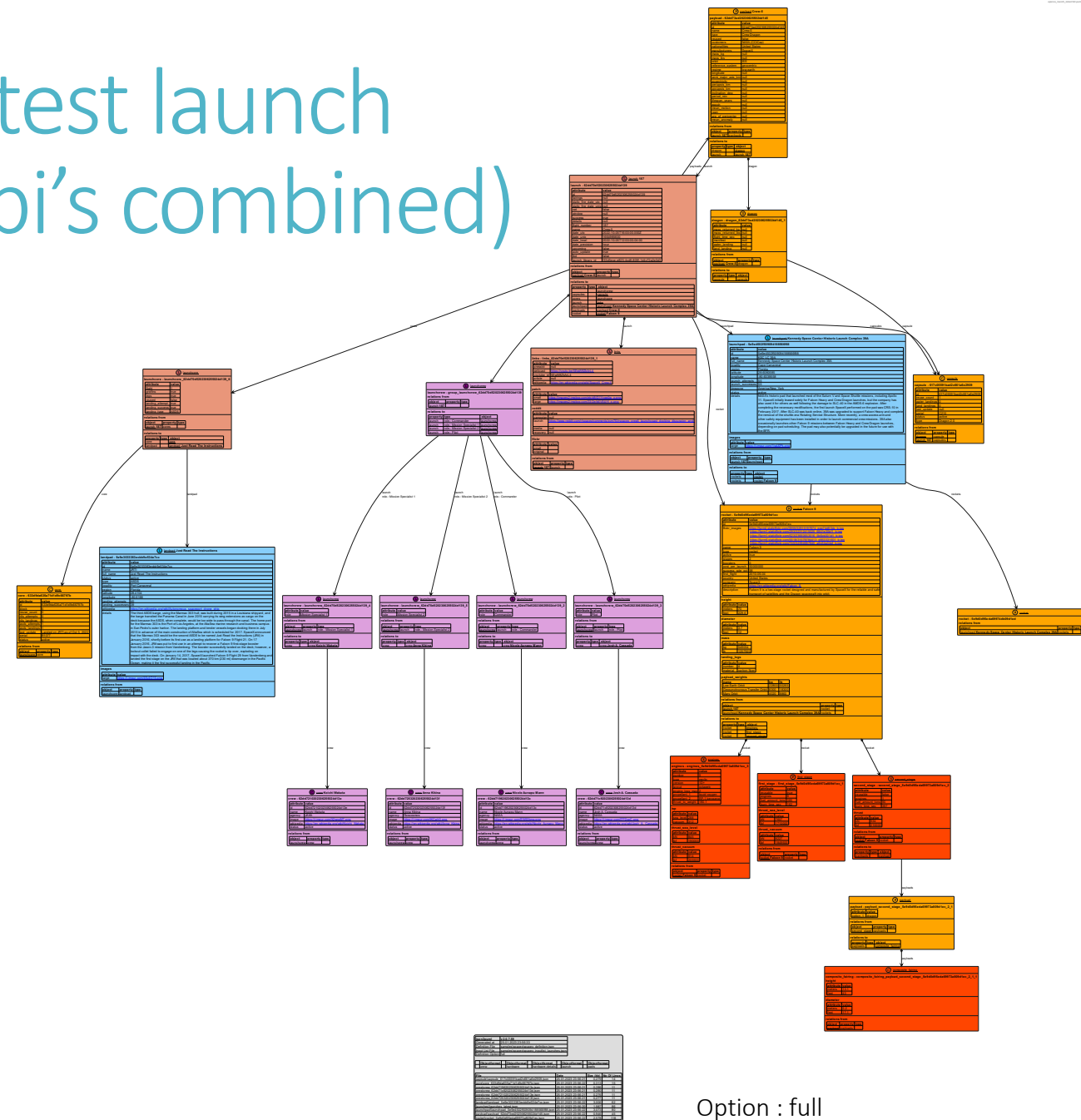
Option : full

Option : full

All data regarding the latest launch of a Space X rocket (8 Api's combined)



Option : compact



Option : full

SWAPI - The Star Wars API

<https://swapi.dev/>

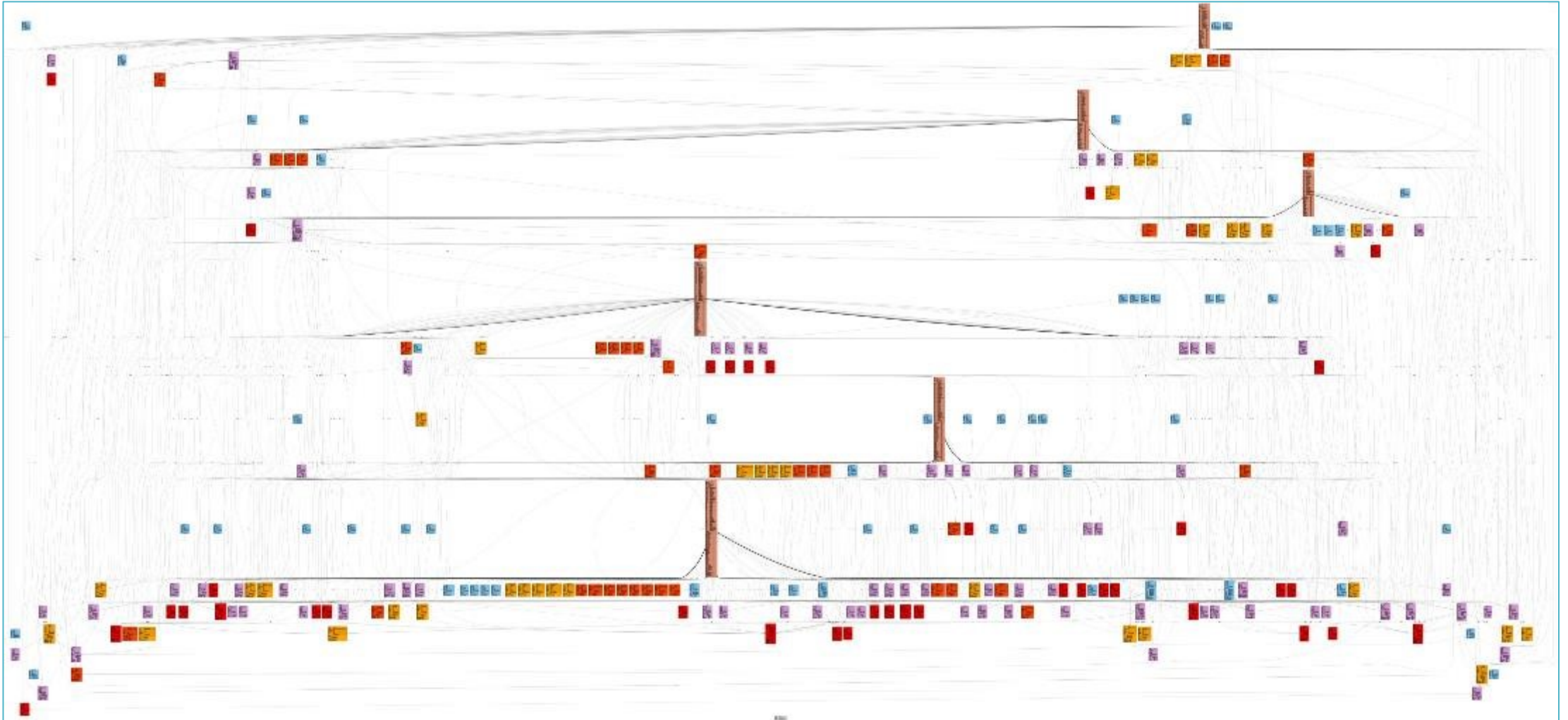
What is this?

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

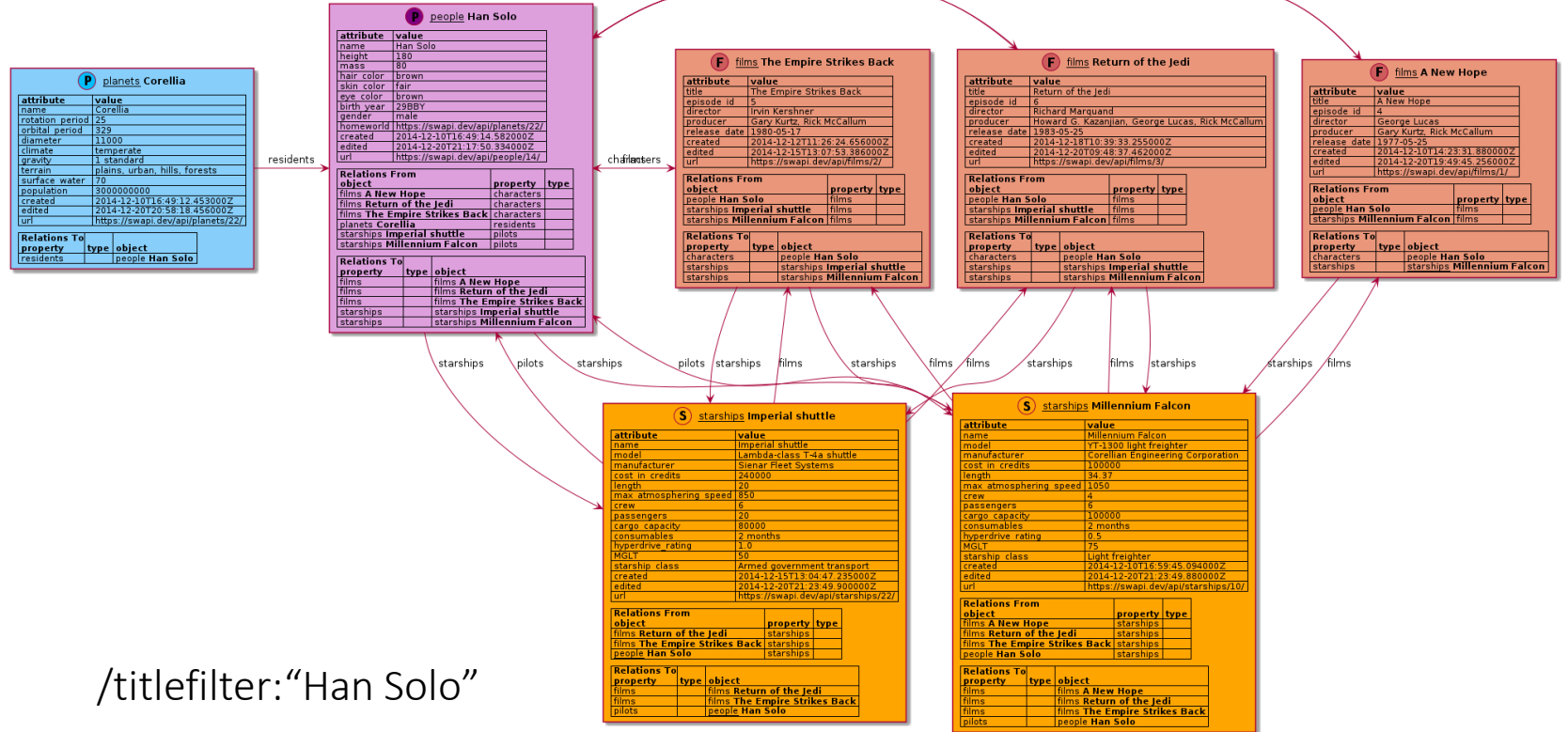
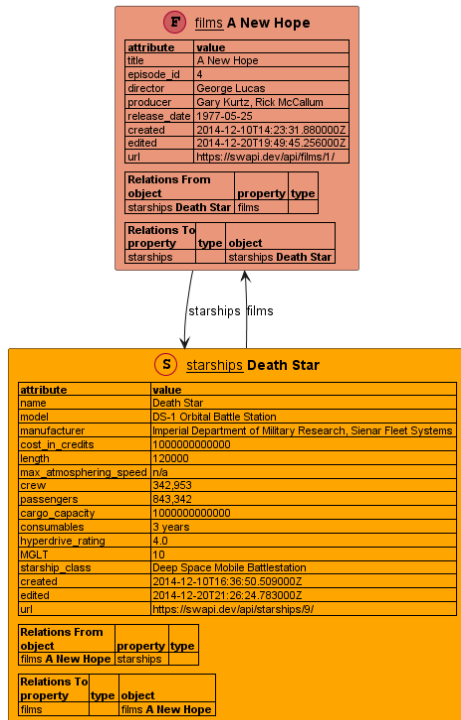
6 Films, 82 Persons, 60 Planets, 37 Species, 36 Star ships, 39 Vehicles



Examples

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

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






/titlefilter: "Han Solo"

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	
<div><div><div></div><div></div><div></div></div><div><div>Objectformat</div><div>films</div><div>starships</div></div></div>			
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"

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 PlantUml command line utility.

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>