

# SPS Solutions

## Starwars API

---

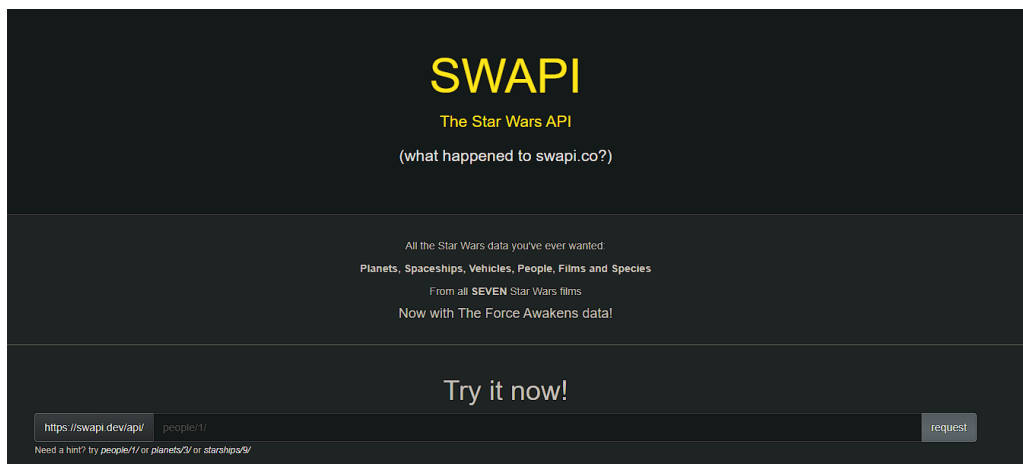
### Documentacion paso a paso de la practica tecnica

1- Analizar el requisito del usuario:

“que necesita” y el “como lo necesita”  
en este caso, por medio de llamadas a la API “Swapi” Obtendremos la informacion de los personajes de la saga, y enviaremos esta en formato CSV.

Que necesita? - Informacion de los personajes de Starwars

Como lo necesita? - Toda la informacion en un formato CSV



2- Analizar los recursos actuales:

Necesitamos dar un vistazo a la API, para saber con que es lo que vamos a trabajar, y el primer inconveniente es que toda la informacion de los personajes no puede ser recabada en una sola llamada, esta se encuentra “paginada”, por lo cual tendremos que idear la forma de extraerlos.

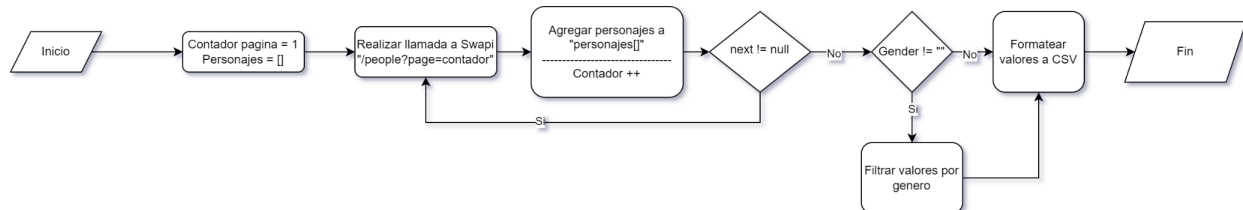
```

{
  "count": 82,
  "next": "https://swapi.dev/api/people/?page=2",
  "previous": null,
  "results": [
    {
      "name": "Luke Skywalker",
      "height": "172",
      "mass": "77",
      "hair_color": "blond",
      "skin_color": "fair",
      "eye_color": "blue",
      "birth_year": "198BY",
      "gender": "male",
      "homeworld": "https://swapi.dev/api/planets/1/",
      "films": [
        "https://swapi.dev/api/films/1/",
        "https://swapi.dev/api/films/2/"
      ]
    }
  ]
}

```

Dado un analisis de los datos podremos aplicar la siguiente logica, ya que si buscamos el recurso `/people` y `/people?page=1` nos da el mismo resultado inicial nos facilita el uso de variables auxiliares y mas por lo tanto.

Realizaremos una llamada a la api hasta que el valor en `next` que nos indica que aun hay valores, regrese con el valor de null, hasta ese momento, es cuando sabremos que obtuvimos todos los valores de los personajes



### 3- Diseño archivo RAML:

Una vez dada con la logica de resolucion de este problema, podemos empezar con el diseño de la API con todos sus recursos ya identificados, en este caso, solo es necesario realizar uno para los personajes como se indica, pero, por buenas practicas, decidi agregar un status/health-check para visualizar el estado de la API.

```

/personajes:
  get:
    description: este recurso es principalmente utilizado para obtener TODOS los personajes de la franquicia
    queryParameters:
      gender:
        required: false
        type: string
        enum: [male,female,n/a]
        description: Filtrar los personajes por genero (male, female o ninguno de estos 2)
    responses:
      200:
        body:
          application/csv:
            type: csvrespuesta

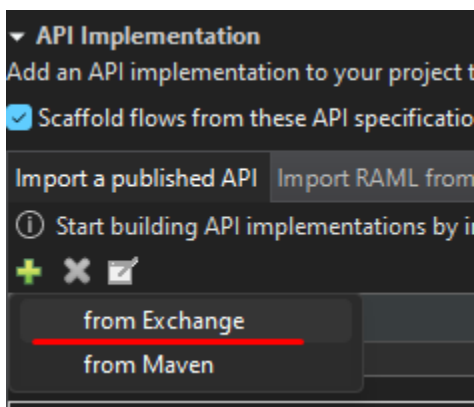
/status-check:
  description: verificar si la API se encuentra activa
  get:
    responses:
      200:
        body:
          application/json:
            example:
              {
                "message": "La api se encuentra corriendo sin problemas!"
              }

```

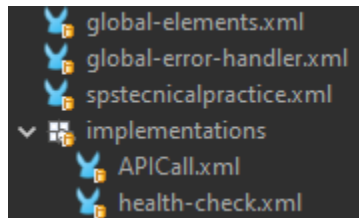
Nota: dentro del QueryParameter decidi agregar un Enum para los generos, ya que al parecer solo se distinguen en 3 dentro la informacion de la API, asi evitaremos algun dato malicioso por parte del usuario.

#### 4- Crear la solución en Anypoint platform

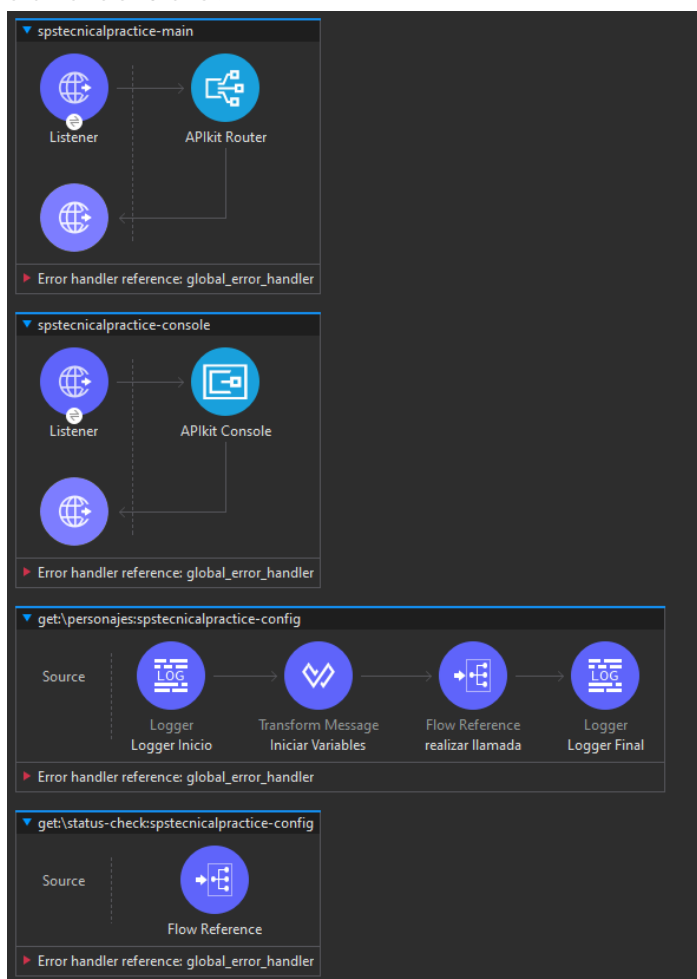
Se crea un nuevo proyecto en base a un archivo RAML creado desde Exchange, este creara toda la configuracion necesaria para poder iniciar con el proyecto, una vez teniendo este, se inicia con el paso de administrar la arquitectura de los archivos







Los archivos quedaron de la siguiente manera \*todo en base a mi experiencia



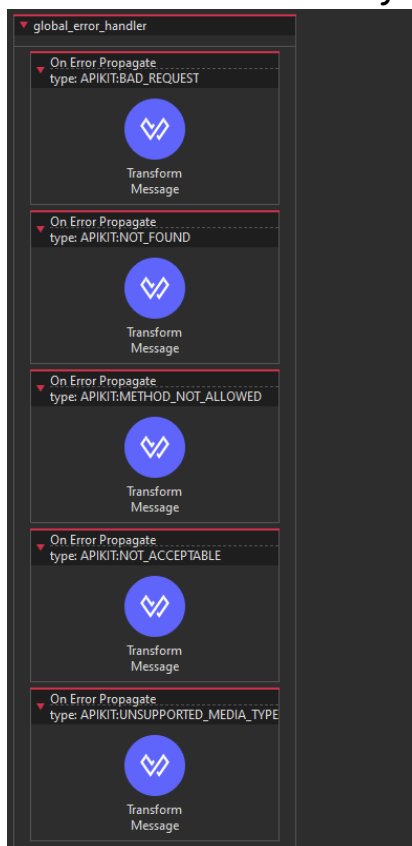
- spstecnicalpractice.xml - archivo main de la solución, dentro de este archivo XML se encuentran los APIKitRouter y Listeners principales de la solución



- global-elements.xml - este es un archivo XML el cual es creado para poder contener todas las configuraciones principales y secundarias que pueden llegar a ser reutilizados dentro de otros módulos

Global Configuration Elements	
Type	Name
 Configuration (Configuration)	Configuration
 Router (Configuration)	spstecnicalpractice-config
 HTTP Listener config (Configuration)	HTTP_Listener_config
 HTTP Request configuration (Configuration)	HTTPS_Swapi_Call_Configuration

- global-error-handler.xml - este se utiliza para definir un manejo centralizado de errores en una aplicación Mule, este archivo permite configurar cómo se deben manejar los errores que ocurren en cualquier parte de la aplicación, lo que facilita la gestión y la coherencia en el manejo de errores en todo el proyecto



solo es necesario hacer referencia a este dentro de las configuraciones globales y dentro del archivo main sobre el listener principal o todos aquellos flujos que requieran de este

- Implementations(folder) - dentro de este folder se anexan todas las implementaciones (recursos) de la API a realizar, en este caso se encuentra la implementacion del recurso principal "personajes" y la implementacion del recurso de monitoreo de estado de la API

## 5- implementacion de la logica

Dentro de este paso solo queda aplicar la logica escrita arriba por medio de un diagrama de clases, la cual consta de la siguiente manera

### spstecnicalpractice.xml

1- se inicializan todas las variables necesarias para la siguiente logica, siendo estas:

- personajes[] -un arreglo que contendra todos los objetos de personajes extraidos de la API
- Contador - un contador que ira incrementando para facilitar el movimiento entre paginaciones (inicia en 1)
- gender - una variable que almacenara el valor del QueryParameter genero ingresado por el usuario, el cual quedara como vacio "" para evitar cualquier problema de la logica cuando el usuario no indique su valor

2- se realiza una llamada a un flujo especificado en el archivo APICall.xml que es la implementacion de la logica del recurso

### APICall.xml

1- se realiza la llamada al host

(<https://swapi.dev/api/people?page={contador}>) donde "{contador}" siendo la variable inicializada en el paso pasado.

2- se realiza el mapeo de datos de los personajes extraidos del archivo actual para ingresarse a la variable personajes[] y aumentando la variable contador

```
%dw 2.0
output application/json
---
{
  "personajes": vars.personajes.personajes ++ (payload.results map ((item, index) ->
    {
      "name": item.name,
      "height":item.height,
      "mass": item.mass,
      "hair_color": item.hair_color,
      "skin_color": item.skin_color,
      "eye_color": item.eye_color,
      "birth_year": item.birth_year,
      "gender": item.gender
    }
  ))
}
```

3- se realiza una evaluacion donde la llamada de la API principal nos indica si aun quedan paginas con informacion sobre los personajes, siendo el caso donde este aun tenga valor, regresara a iniciar el mismo

flujo el que se encuentra, realizando los mismos pasos anteriores solo que, sobre la siguiente pagina de personajes

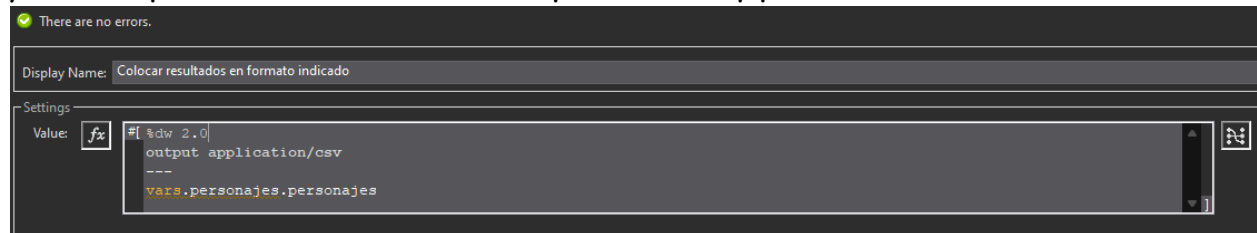


4- Cuando la variable "next" en la llamada alcanza el valor de null, este pasa al siguiente subflujo para tratar la logica de genero, preguntando si existe algun valor dentro de la variable "gender"

en cualquier caso positivo, este realiza una funcion de filter sobre los datos con el genero especificado (ejemplo: male)

```
%dw 2.0
output application/json
---
{
    personajes: vars.personajes.personajes filter ((personaje) -> (personaje.gender == "male"))
}
```

5- dar formato a los datos en tipo CSV para la presentacion como lo desea el usuario, esto se puede realizar facilmente agreando dicho valor, y solo especificando el formato que seria "application/CSV"



Resultados:



**GET** <http://localhost:8081/swapi/v1/personajes>

**Params** • Authorization Headers (7) Body Scripts Tests Settings

Query Params

<input type="checkbox"/>	Key	Value
<input type="checkbox"/>	gender	male
	Key	Value

**Body** Cookies Headers (3) Test Results

Pretty	Raw	Preview	Visualize	Text	
1	name,height,mass,hair_color,skin_color,eye_color,birth_year,gender				
2	Luke Skywalker,172,77,blond,fair,blue,19BBY,male				
3	C-3P0,167,75,n/a,gold,yellow,112BBY,n/a				
4	R2-D2,96,32,n/a,white\, blue,red,33BBY,n/a				
5	Darth Vader,202,136,none,white,yellow,41.9BBY,male				
6	Leia Organa,150,49,brown,light,brown,19BBY,female				
7	Owen Lars,178,120,brown\, grey,light,blue,52BBY,male				
8	Beru Whitesun lars,165,75,brown,light,blue,47BBY,female				
9	R5-D4,97,32,n/a,white\, red,red,unknown,n/a				
10	Biggs Darklighter,183,84,black,light,brown,24BBY,male				
11	Obi-Wan Kenobi,182,77,auburn\, white,fair,blue-gray,57BBY,male				
12	Anakin Skywalker,188,84,blond,fair,blue,41.9BBY,male				
13	Wilhuff Tarkin,180,unknown,auburn\, grey,fair,blue,64BBY,male				
14	Chewbacca,228,112,brown,unknown,blue,200BBY,male				
15	Han Solo,180,80,brown,fair,brown,29BBY,male				
16	Greedo,173,74,n/a,green,black,44BBY,male				
17	Jabba Desilijic Tiure,175,1\,358,n/a,green-tan\, brown,orange,600BBY,hermaphrodite				
18	Wedge Antilles,170,77,brown,fair,hazel,21BBY,male				
19	Jek Tono Porkins,180,110,brown,fair,blue,unknown,male				
20	Yoda,66,17,white,green,brown,896BBY,male				
21	Palpatine,170,75,grey,pale,yellow,82BBY,male				

SPS Solutions / llamada api local

GET

http://localhost:8081/swapi/v1/personajes?gender=male

Params

Authorization

Headers (7)

Body

Scripts

Tests

Settings

Query Params

<input checked="" type="checkbox"/>	Key	Value
<input checked="" type="checkbox"/>	gender	male
	Key	Value

Body

Cookies

Headers (3)

Test Results

Pretty

Raw

Preview

Visualize

Text

1

name,height,mass,hair\_color,skin\_color,eye\_color,birth\_year,gender

2

Luke Skywalker,172,77,blond,fair,blue,19BBY,male

3

Darth Vader,202,136,none,white,yellow,41.9BBY,male

4

Owen Lars,178,120,brown\, grey,light,blue,52BBY,male

5

Biggs Darklighter,183,84,black,light,brown,24BBY,male

6

Obi-Wan Kenobi,182,77,auburn\, white,fair,blue-gray,57BBY,male

7

Anakin Skywalker,188,84,blond,fair,blue,41.9BBY,male

8

Wilhuff Tarkin,180,unknown,auburn\, grey,fair,blue,64BBY,male

9

Chewbacca,228,112,brown,unknown,blue,200BBY,male

10

Han Solo,180,80,brown,fair,brown,29BBY,male

11

Greedo,173,74,n/a,green,black,44BBY,male

12

Wedge Antilles,170,77,brown,fair,hazel,21BBY,male

13

Jek Tono Porkins,180,110,brown,fair,blue,unknown,male

14

Yoda,66,17,white,green,brown,896BBY,male

15

Palpatine,170,75,grey,pale,yellow,82BBY,male

16

Boba Fett,183,78.2,black,fair,brown,31.5BBY,male

17

Bossk,190,113,none,green,red,53BBY,male

18

Lando Calrissian,177,79,black,dark,brown,31BBY,male

19

Lobot,175,79,none,light,blue,37BBY,male

20

Ackbar,180,83,none,brown mottle,orange,41BBY,male

21

Arvel Crynyd,unknown,unknown,brown,fair,brown,unknown,male

22

Wicket Systri Warrick,88,20,brown,brown,brown,8BBY,male

23

Nien Nunb,160,68,none,grey,black,unknown,male

24

Qui-Gon Jinn,193,89,brown,fair,blue,92BBY,male

25

Nute Gunray,191,90,none,mottled green,red,unknown,male

26

Finis Valorum,170,unknown,blond,fair,blue,91BBY,male

27

Jar Jar Binks,196,66,none,orange,orange,52BBY,male

28

Roos Tarpals,224,82,none,grey,orange,unknown,male

29

Rugor Nass,206,unknown,none,green,orange,unknown,male

30

Ric Olié,183,unknown,brown,fair,blue,unknown,male

GET http://localhost:8081/swapi/v1/personajes?gender=female

Params • Authorization Headers (7) Body Scripts Tests Settings

#### Query Params

<input checked="" type="checkbox"/>	Key	Value
<input checked="" type="checkbox"/>	gender	female
<input type="checkbox"/>	Key	Value

Body Cookies Headers (3) Test Results

Pretty Raw Preview Visualize Text 

```

1 name,height,mass,hair_color,skin_color,eye_color,birth_year,gender
2 Leia Organa,150,49,brown,light,brown,19BBY,female
3 Beru Whitesun lars,165,75,brown,light,blue,47BBY,female
4 Mon Mothma,150,unknown,auburn,fair,blue,48BBY,female
5 Padmé Amidala,185,45,brown,light,brown,46BBY,female
6 Shmi Skywalker,163,unknown,black,fair,brown,72BBY,female
7 Ayla Secura,178,55,none,blue,hazel,48BBY,female
8 Adi Gallia,184,50,none,dark,blue,unknown,female
9 Cordé,157,unknown,brown,light,brown,unknown,female
10 Luminara Unduli,170,56.2,black,yellow,blue,58BBY,female
11 Barriss Offee,166,50,black,yellow,blue,40BBY,female
12 Dormé,165,unknown,brown,light,brown,unknown,female
13 Zam Wesell,168,55,blonde,fair\, green\, yellow,yellow,unknown,female
14 Taun We,213,unknown,none,greys,black,unknown,female
15 Jocasta Nu,167,unknown,white,fair,blue,unknown,female
16 R4-P17,96,unknown,none,silver\, red,red\, blue,unknown,female
17 Shaak Ti,178,57,none,red\, blue\, white,black,unknown,female
18 Sly Moore,178,48,none,pale,white,unknown,female
19

```

SPS Solutions / llamada api local

GET <http://localhost:8081/swapi/v1/personajes?gender=n/a>

Params • Authorization Headers (7) Body Scripts Tests Settings

Query Params

<input checked="" type="checkbox"/>	Key	Value
<input checked="" type="checkbox"/>	gender	n/a
	Key	Value

Body Cookies Headers (3) Test Results

Pretty Raw Preview Visualize Text

```
1 name,height,mass,hair_color,skin_color,eye_color,birth_year,gender
2 C-3P0,167,75,n/a,gold,yellow,112BBY,n/a
3 R2-D2,96,32,n/a,white\, blue,red,33BBY,n/a
4 R5-D4,97,32,n/a,white\, red,red,unknown,n/a
5
```

6- subir solucion a GIT

Nota:

Gracias por la oportunidad de realizar esta practica, espero que se encuentren satisfechos con mis resultados, cualquier otra cosa espero saber mas de ustedes, muchas gracias.