

The background features a dark, textured surface with several glowing, parallel lines in shades of green and blue that create a sense of depth and movement. These lines are set against a backdrop of faint, concentric circles and a grid of small dots, suggesting a digital or network environment.

# Connectors - part II

With Microsoft Power platform

By Chris Noring



- Creating and consuming a connector
- Creating connector from Open API
- Triggers
- Policies

# Agenda

# Microsoft Power platform



Power apps, build traditional apps



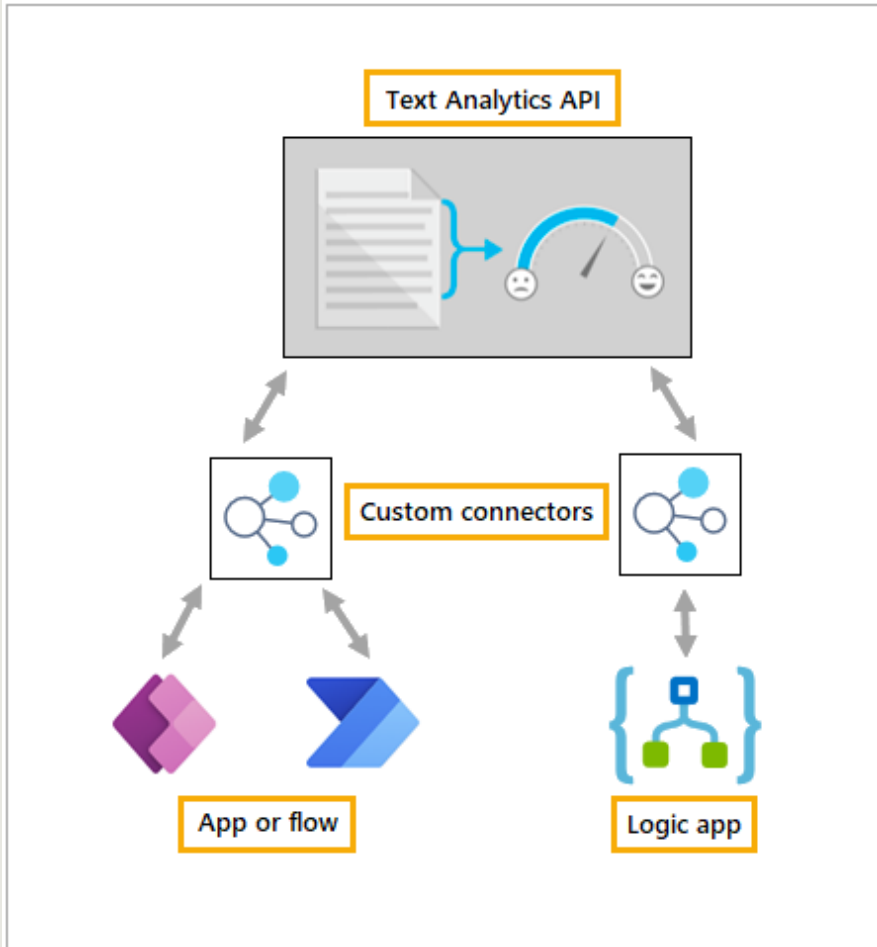
Power BI, business analytics,  
create dashboards



Power automate, automate your  
business flows

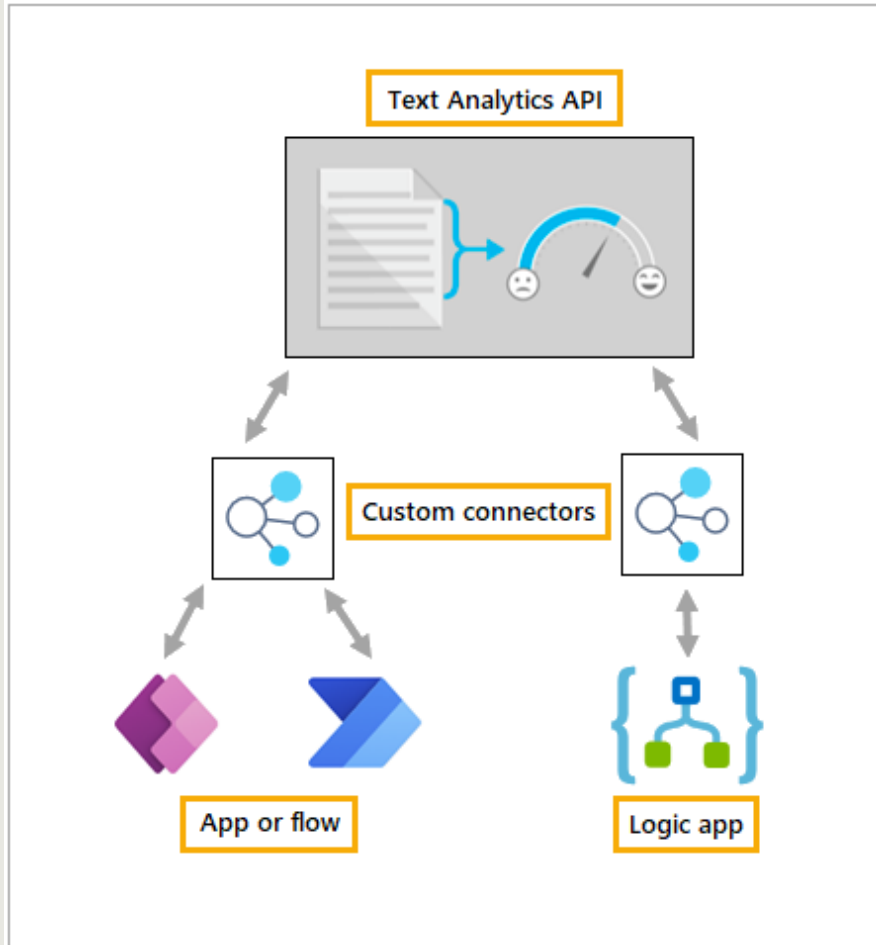


Power virtual agents.  
Intelligent virtual agents, bots and  
more



# What's a connector?

Connector wraps an API



# Connector can be consumed by many services

Power apps

Power automate/flow

Logic apps

# Connector creation process

1

## Shorter route:

- Generate from Open API spec file
- Use in Power platform (Power Apps, Automate or Logic apps)
- Optional, Publish, usable by the whole world

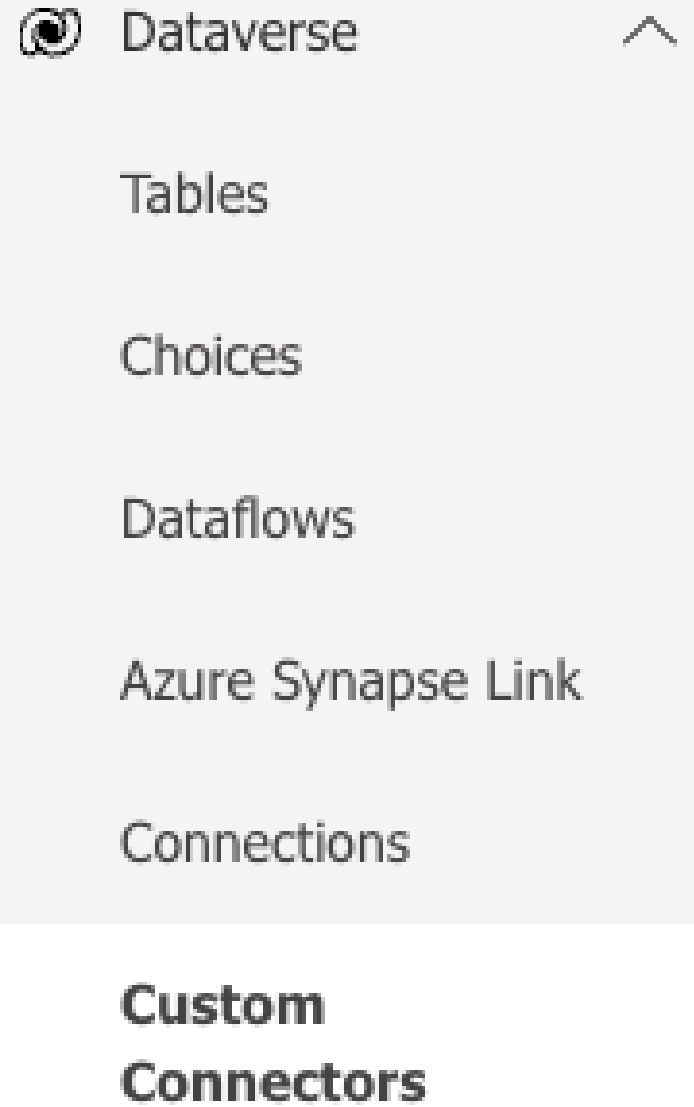
2

## Longer route:

- **Create API**
- Input URL, set up actions
- Use in Power platform
- Optional, Publish, usable by the whole world

# Creating a connector

- Add URL
- Add security
- Add actions, with responses (for every action)
- Create!



# Input URL


1. General > 2. Security > 3. Definition > 4. Code (Preview) > 5. Test

Swagger Editor ✓ Update connector ✕ Clc

### General information

Add an icon and short description to your custom connector. Your host and base URL will be automatically generated from the swagger file.

### General information



Upload connector icon  
Supported file formats are PNG and JPG. (< 1MB)

↑ Upload

Icon background color

Description

Connect via on-premises data gateway [Learn more](#)

Scheme \*

HTTPS  HTTP

Host \*

Base URL



# Add security

1. General > **2. Security** > 3. Definition > 4. Code (Preview) > 5. Test

Swagger Editor ✓ Update connector ✕ Close

### Security

Choose the authentication type and fill in the required fields to set the security for your custom connector.  
[Learn more](#)

#### Authentication type

Choose what authentication is implemented by your API \*

No authentication

No authentication

Basic authentication

API Key

OAuth 2.0

# Definition, adding actions

1. General > 2. Security > **3. Definition** > 4. Code (Preview) > 5. Test Swagger Editor Update connector Close

**Actions (3)**  
Actions determine the operations that users can perform. Actions can be used to read, create, update or delete resources in the underlying connector.

- 1 People ...
- 2 Planets ...
- 3 **Person** ...

**References (0)**  
References are reusable parameters used by both actions and triggers.

**Policies (0)**  
Policies are used to change the behavior of actions and triggers through configuration. You can use one or more policies from a set of predefined templates.

**General**

Summary [Learn more](#)

Description [Learn more](#)

Operation ID \*  
This is the unique string used to identify the operation.

Visibility [Learn more](#)  
 none  advanced  internal  important

**Request**  
It defines the pre-requirements needed in order to make a request. Describes a single operation parameter. A unique parameter is defined by a combination of a name and location.

Verb \*  
The verb describes the operations available on a single path.  
**GET**

# Consume a connector (ex Canvas app)

01

Add as data source

02

Invoke as a function using Fx

03

Handle response and connect to a gallery



**DEMO,  
create a connector  
from a custom API**

## FUSION TEAM DEVELOPMENT PROCESS



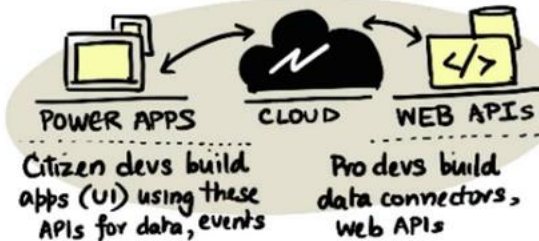
IT/PRO DEVS + CITIZEN DEVS

WORK TOGETHER TO CREATE APPS THAT SOLVE BUSINESS NEEDS



POWER APPS IS A LOW CODE DEV APPROACH ALL !! CAN USE

- ✓ FOCUS ON YOUR AREA OF EXPERTISE OR COMFORT
- ✓ ASK FOR HELP WHERE YOU NEED IT (data)
- ✓ REPLACE OR ITERATE LEGACY APPS (fast)



# Open API Spec

- API description format for REST APIs
- Describe your entire API, including:
  - Available endpoints ( /users )
  - Operations on each endpoint ( GET /users , POST /users )
  - Operation parameters Input and output for each operation

The image shows the Swagger Editor interface. On the left, a code editor displays the OpenAPI specification for the Swagger Petstore API. The specification includes the following details:

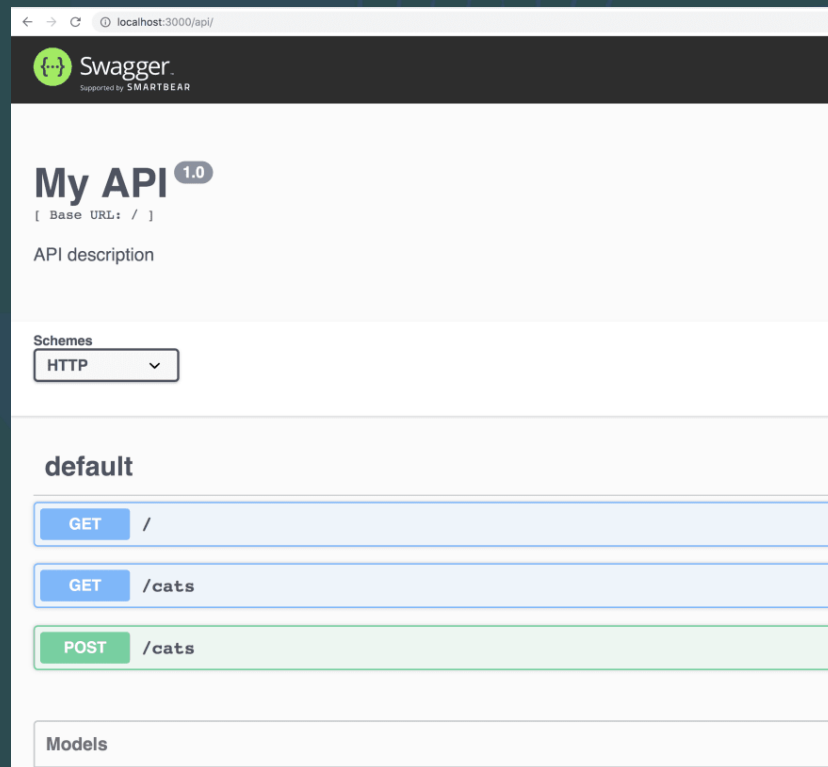
```
1 openapi: "3.0.0"
2 info:
3   version: 1.0.0
4   title: Swagger Petstore
5   license:
6     name: MIT
7 servers:
8   - url: http://petstore.swagger.io/v1
9 paths:
10  /pets:
11    get:
12      summary: List all pets
13      operationId: listPets
14      tags:
15        - pets
16      parameters:
17        - name: limit
18          in: query
19          description: How many items to return at one
20            time (max 100)
21          required: false
22          schema:
23            type: integer
24            format: int32
25      responses:
26        '200':
27          description: A paged array of pets
28          headers:
29            x-next:
30              description: A link to the next page of
31                responses
32              schema:
33                type: string
34            content:
```

On the right, the Swagger UI displays the API's metadata and endpoints. The title is "Swagger Petstore" with version "1.0.0" and "OAS3" specification. The license is MIT. The server URL is "http://petstore.swagger.io/v1". The "pets" endpoint is expanded, showing three operations:

- GET** /pets List all pets
- POST** /pets Create a pet
- GET** /pets/{petId} Info for a specific pet

Below the endpoints, there is a "Models" section which is currently collapsed.

# JavaScript REST API (Nest.js)

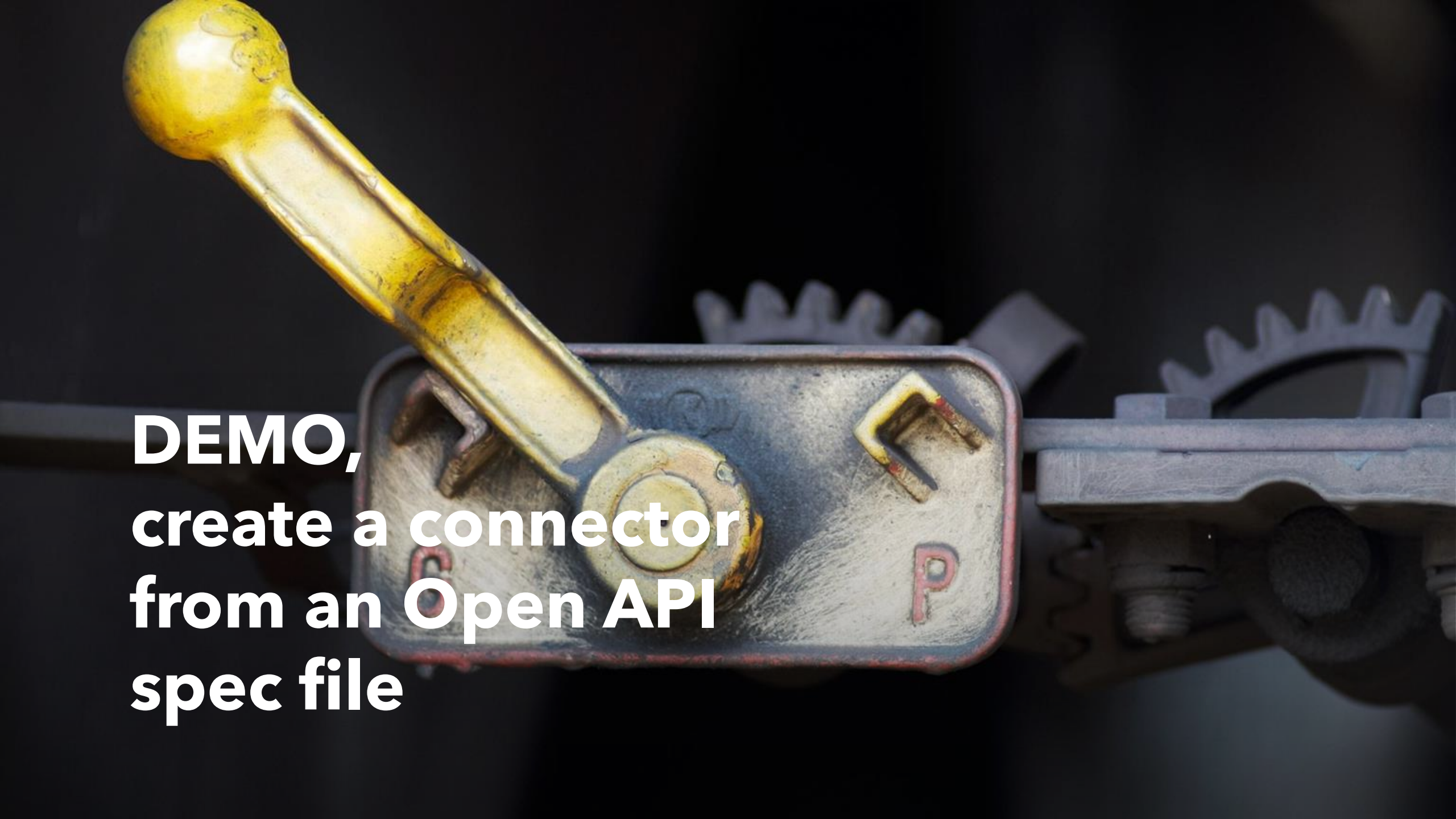


```
import { Controller, Get, Post, Body } from '@nestjs/common';
import { CatsService } from './cats.service';
import { Cat } from './cat';

@Controller('cats')
export class CatsController {
  constructor(private srv: CatsService) {}

  @Get()
  getCats() {
    return this.srv.getCats();
  }

  @Post()
  createCat(@Body() cat: Cat) {
    this.srv.createCat(cat);
  }
}
```



**DEMO,  
create a connector  
from an Open API  
spec file**



EXPLORER

Get Started

README.md M

swagger.json

openapi.json X

.gitignore

main.py M

index.htm

🔍 📄 🗑️ ⋮

OPEN EDITORS

Get Started

README.md M

swagger.json

X openapi.json

.gitignore

main.py M

index.html ~/Documents/dev/proje...

requirements.txt

swagger.png

FAST-DEMO

&gt; \_\_pycache\_\_

&gt; .vscode

&gt; fast-env

.gitignore

main.py M

openapi.json

README.md M

requirements.txt

swagger.json

swagger.png

&gt; OUTLINE

&gt; TIMELINE

&gt; METADATA

{} openapi.json &gt; {} paths &gt; {} /items/

```
1  {
2    "openapi": "3.0.2",
3    "info": {
4      "title": "FastAPI",
5      "version": "0.1.0"
6    },
7    "paths": {
8      "/items/": {
9        "get": {
10         "summary": "Read Items",
11         "operationId": "read_items_items__get",
12         "parameters": [
13           {
14             "required": false,
15             "schema": {
16               "title": "Page",
17               "type": "integer",
```

PROBLEMS 4

TERMINAL

AZURE

JUPYTER

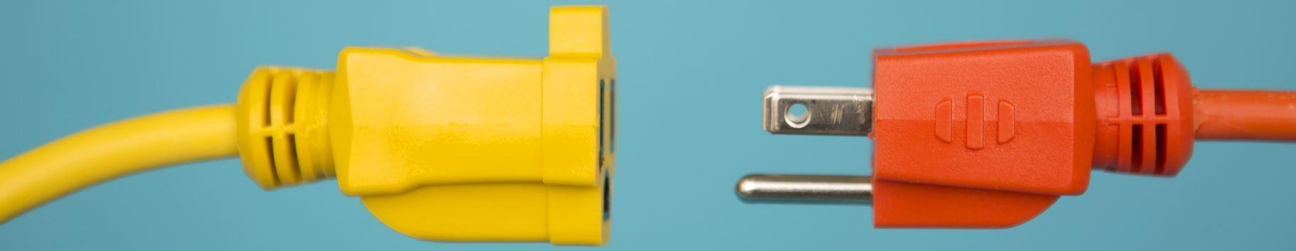
OUTPUT

DEBUG CONSOLE

.NET INTERACTIVE

zsh + 🗑️ ⬆️ ✕

○ (base) → fast-demo git:(main) X

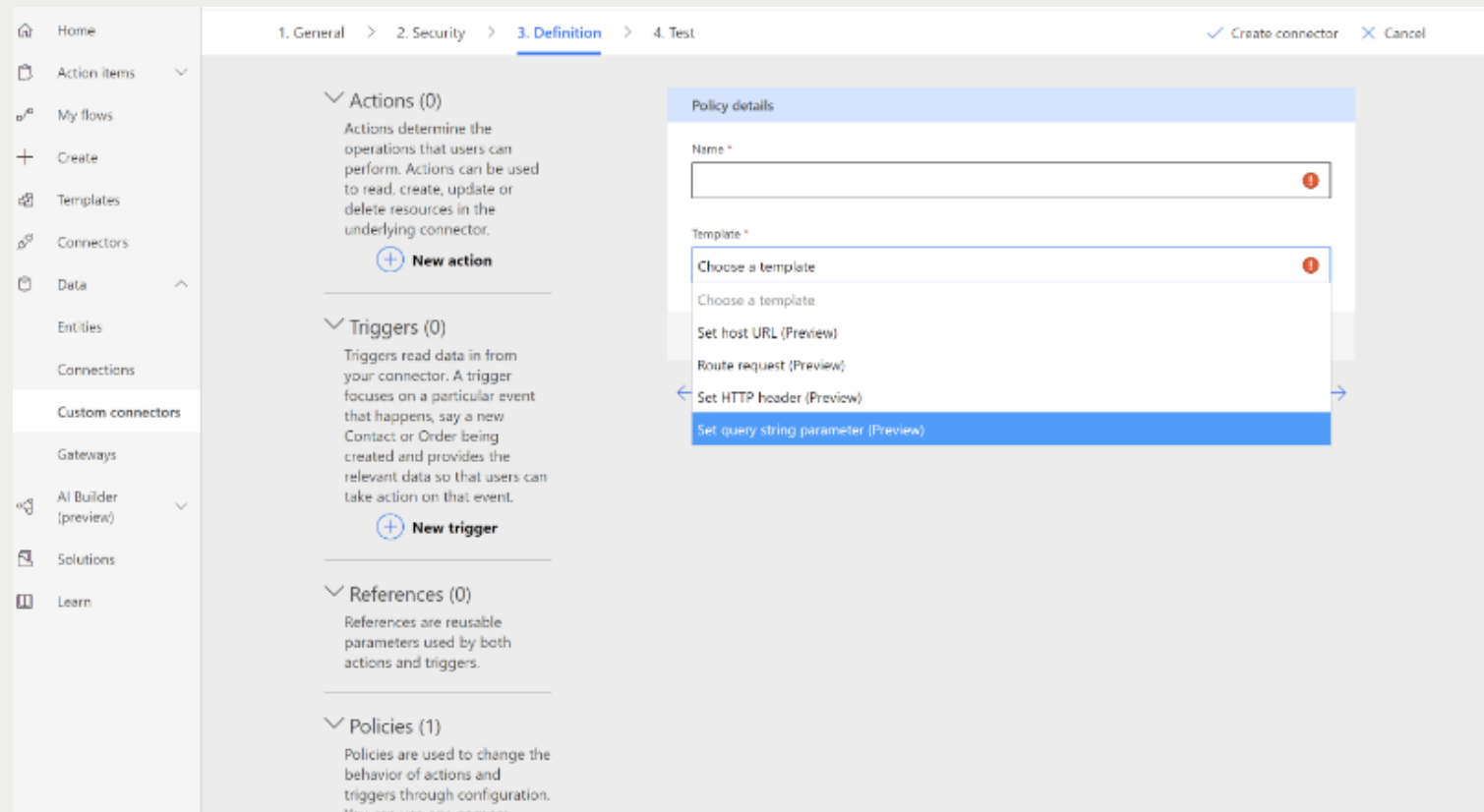


- Policies can be used to **modify** the behaviour of connectors at runtime
- For example, policies are used to enforce throttling limits on API calls to route calls to different endpoints, and so on

# Connector policies

# Adding a policy

- Add to definition page
- Select a template



The screenshot displays the Azure API Management console interface. The breadcrumb navigation at the top indicates the current step is '3. Definition'. The left-hand navigation pane shows the 'Custom connectors' section expanded. The main content area is divided into sections for 'Actions (0)', 'Triggers (0)', 'References (0)', and 'Policies (1)'. A 'Policy details' dialog box is open, featuring a 'Name' field and a 'Template' dropdown menu. The dropdown menu is expanded, showing a list of available templates: 'Choose a template', 'Set host URL (Preview)', 'Route request (Preview)', 'Set HTTP header (Preview)', and 'Set query string parameter (Preview)'. The 'Set query string parameter (Preview)' option is currently selected and highlighted in blue. The dialog also includes 'Create connector' and 'Cancel' buttons at the top right.

# Policy example

- If route is hit
- BEFORE: /GetOrders?pageSize=1000, then  
AFTER: override query parameter to  
pageSize=5 **limits response**
- No code, policy by configuration!

Name \*

limit output

Template \* [Learn more](#)

Set query string parameter

Adds or updates value of request query string parameter

Operations

List of actions and triggers to which the policy will apply to. If no operation is selected, this policy will apply to all operations.

GetOrders

Query parameter name \*

Specifies the name of the query parameter to be set.

pageSize

Query parameter value \*

Specifies the value of the query parameter to be set.

5

Action if query parameter exists


Specifies what action to take when the query parameter is already specified.

override

# Triggers



## Why a trigger

- a system needs to respond to the changes in the underlying data or services
- 

## What's a trigger

- A condition, when condition is true, do something

## What do I use it for, scenarios

- When a record is created," or when a certain event takes place in the service that is defined by the custom connector, such as "When alarm is raised.

# Trigger types



## Polling trigger

**timed activity** that initiates a call to the service API on a regular, configurable interval to determine if new data is available

**Ex:** Periodically call the voice mailbox and check for new messages



## Webhook trigger

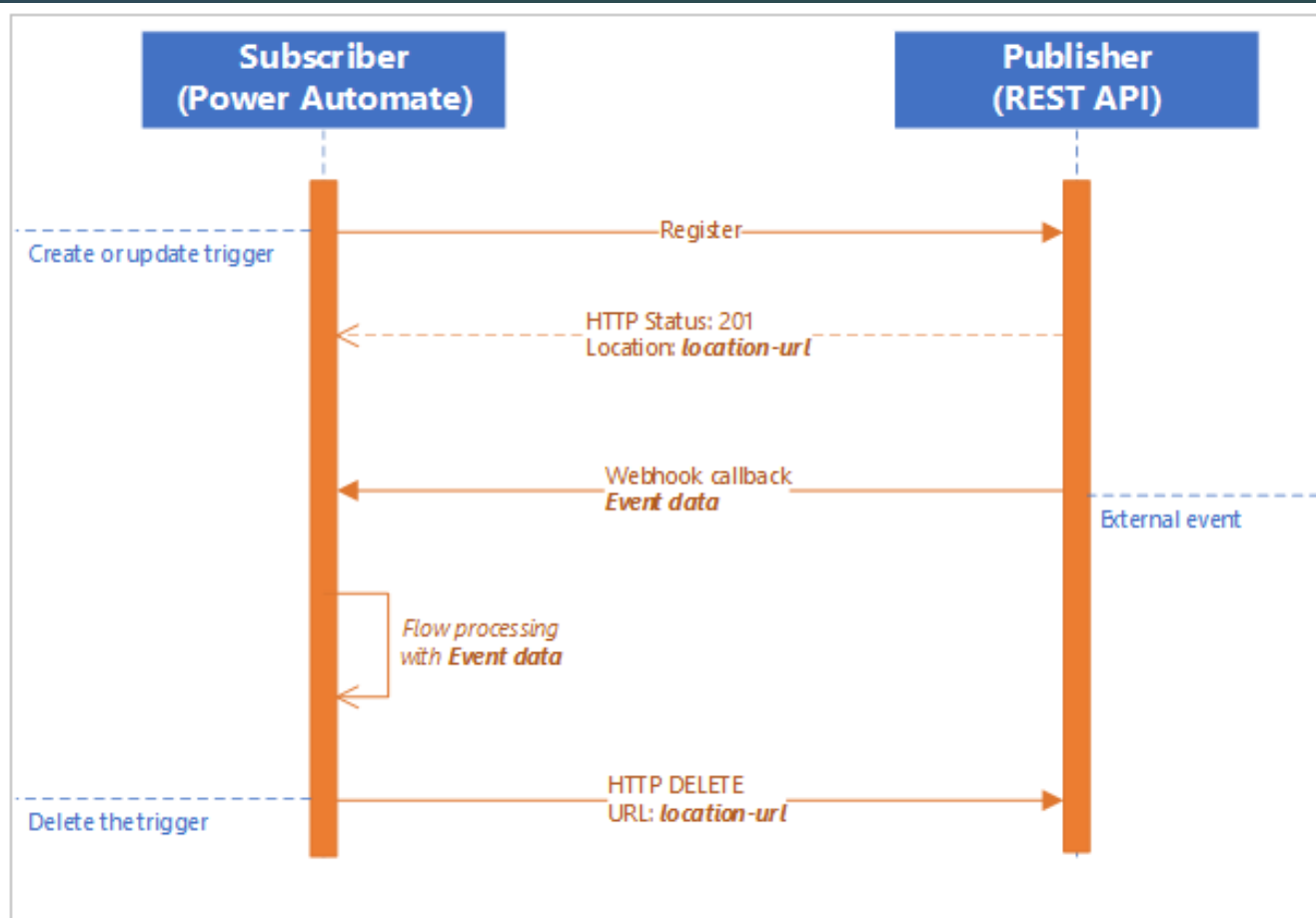
data pushed in your direction as the result of an event

**Ex:** Let your voicemail system **send YOU an email** when a new voice message is received

Property: The service that supports webhook triggers must be able to maintain a list of parties to call back (URL or address) and know "how" to call back .

**Ex:** a list of email addresses and the ability to send a notification email.

# Architecture



# Defining a trigger

1. General > 2. Security > **3. Definition** > 4. Test

Swagger Editor  Create connector  Cancel

> Actions (8)

Triggers (1)

Triggers read data in from your connector. A trigger focuses on a particular event that happens, say a new Contact or Order being created and provides the relevant data so that users can take action on that event.

**1 InvoiceCreated** ...

**+ New trigger**

References (6)

References are reusable parameters used by both actions and triggers.

1 Invoice

2 InvoiceSchema

3 InvoiceType

**General**

Summary [Learn more](#)

When Invoice is Created

Description [Learn more](#)

When Invoice is Created

Operation ID \*

This is the unique string used to identify the operation.

InvoiceCreated

Visibility [Learn more](#)

none  advanced  internal  important

Trigger type [Learn more](#) \*

Webhook  Polling



# Summary

- Microsoft Power Platforms is for everyone, citizen developer as well as code-first developer
- Connectors are wrapped APIs consumable by low code platforms like Power Apps, Power Automate and Logic apps
- Code-first developers can build APIs and generate connectors
- Connectors can be created from scratch as well as Open API specs
- Connectors can be consumed by Power Automate, Power apps and Logic apps
- Policies and Triggers are more advanced features you could leverage as well