



Microsoft Dynamics 365
and Power Platform Community Conference

MAY 25 – 27, 2026
Portorož, Slovenia, Europe

How to NOT depend on 3rd parties when consuming APIs

Tobias Fenster
General Manager & Chief Engineer,
4PS by Hilti



What's the problem?

We need to integrate with more and more systems via APIs

We also integrate our ERP / CRM / whatever into more and more other systems

During development and automated tests, we often face challenges

- Availability might not be easy
- Data often is an issue
- Scaling isn't easy: What if 3 pipelines run at the same time?

“Mocking” can be a solution: Instead of the real ERP / CRM or 3rd party system, you have a “mock”, which is a system that behaves in the same way

- For an API, that means it accepts the same (relevant) requests and send the same responses

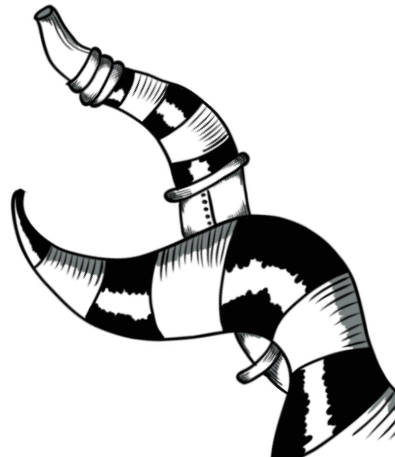
What is **WIREFMCK**?

<https://github.com/wiremock/wiremock>:

WireMock is a popular open-source tool for API mock testing with over 5 million downloads per month. It can help you to create stable test and development environments, isolate yourself from flaky 3rd parties and simulate APIs that don't exist yet.

Key features:

- Mock APIs by defining them via JSON or code
- Record API calls and create mocks from that
- Stateful behavior simulation



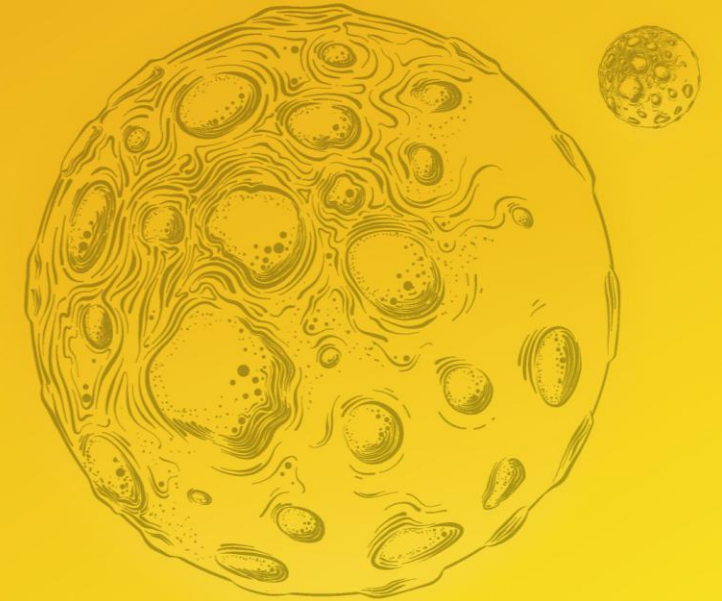


Microsoft Dynamics 365 and Power Platform Community Conference

MAY 25 – 27, 2026
Portorož, Slovenia, Europe

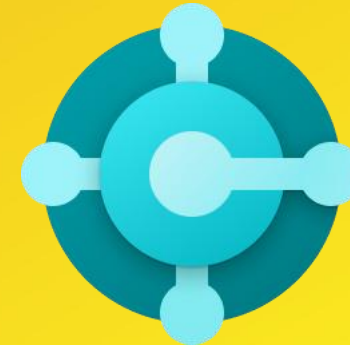
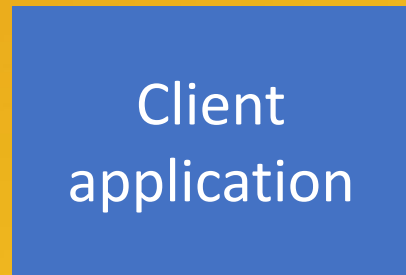
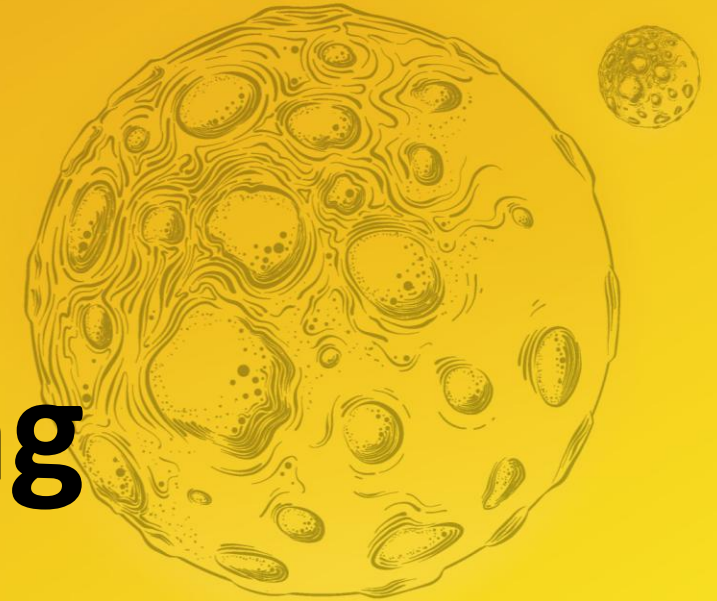
Demo

Of course, hello world!



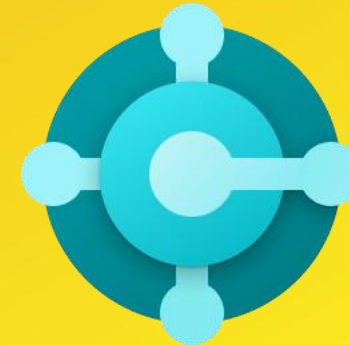
WireMock for BC mocking

Scenario 1: Use WireMock to mock BC



WireMock for BC mocking

Scenario 1: Use WireMock to mock BC



Example: Get companies, CRUD actions for customers

Scenario 1: Mock BC

For reading, we need to mock responses to:

- GET /companies
- GET /companies(<id>)/customers
- GET /companies(<id>)/customers(<id>)

Done with “standard” WireMock in Java using recorded and optimized JSON files

Client is tiny C# Console application

- Dev and test

Advanced topics: Stateful behavior

Scenario:

- A new customer is created
- Only then can it be retrieved and updated
- There must be a different response before and after the update if the customer is retrieved
- The customer can only be deleted after it was created, but also after it was modified
- Once it is deleted, it can't be retrieved

Done via a “scenarioName”, “requiredScenarioState” and “newScenarioState”

Advanced topics: Stateful behavior

Mock responses to:

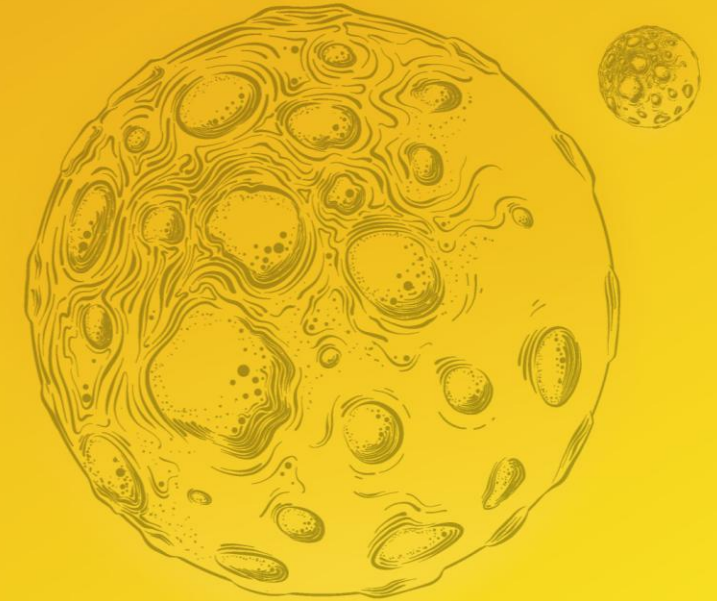
- GET /companies(<id>)/customers(<id>) – multiple different ones
- POST /companies(<id>)/customers
- PATCH /companies(<id>)/customers(<id>)
- DELETE /companies(<id>)/customers(<id>)

Demo

Mock BC

What have we seen:

- Set up WireMock for recording
- Record the required steps
- Optimize: Rename, dynamic values, header handling
- Stateful scenarios
- Usage of the mocked backend during development and test development



WireMock for 3rd-party mocking

Scenario 2: Use WireMock to mock 3rd-party applications



WireMock for 3rd-party mocking

Scenario 2: Use WireMock to mock 3rd-party applications



API Call



3rd party
backend



Example: CRUD actions for PetStore application (OpenAPI demo)

Scenario 2: Mock PetStore API

Mock responses to:

- GET /pet/<id> - fixed and dynamic
- POST /pet
- PUT /pet
- DELETE /pet/<id>

Done with WireMock.NET using recording and generated code

Client is BC

- Dev and test

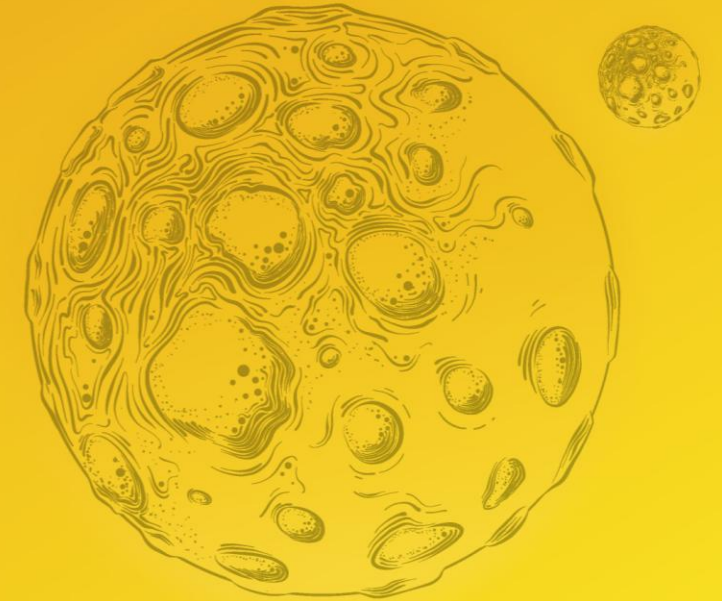


Microsoft Dynamics 365 and Power Platform Community Conference

MAY 25 – 27, 2026
Portorož, Slovenia, Europe

Demo

Mock PetStore



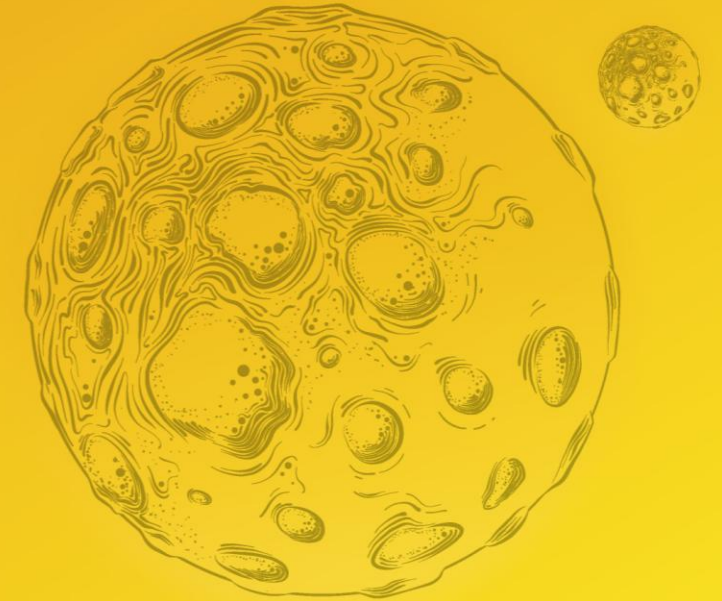


Microsoft Dynamics 365 and Power Platform Community Conference

MAY 25 – 27, 2026
Portorož, Slovenia, Europe

Demo

PetStore Client in BC



Wiremock Inspector

WireMockInspector v1.0.30.0

http://localhost:9095

Requests Mappings Scenarios Settings

All

2026-05-25 13:11:16.724	PUT	/api/v3/pet	200
2026-05-25 13:11:12.907	GET	/api/v3/pet/1000	200
2026-05-25 13:11:09.367	POST	/api/v3/pet	200
2026-05-25 13:11:05.655	GET	/api/v3/pet/1000	404

Wiremock Inspector

WireMockInspector v1.0.30.0

http://localhost:9095

Requests Mappings Scenarios Settings

All 🔍

✔	2026-05-25 13:10:37.206	0a39eead-d202-4258-9f12-2ab0098afbd3	Perfect matches: 1 Partial matches: 0
PUT	/api/v3/pet		
✔	2026-05-25 13:10:37.205	b1f27775-02d1-4033-96fd-9fc89002106d	Perfect matches: 1 Partial matches: 0
GET	/api/v3/pet/1000		
✔	2026-05-25 13:10:37.205	6664a3f8-a4d0-44fe-ae8-131bf5c077b1	Perfect matches: 1 Partial matches: 0
POST	/api/v3/pet		

Wiremock Inspector

WireMockInspector v1.0.30.0

http://localhost:9095

Requests Mappings Scenarios Settings

All

Pet CRUD Current state: Fluffy Sold

Flow Transitions

```
graph TD; Start(( )) --> Created[Fluffy Created]; Created --> Created; Created --> Sold[Fluffy Sold]; Sold --> Sold; Sold --> Deleted[Fluffy Deleted]; Deleted --> Deleted;
```

The diagram illustrates a state transition flow for a pet scenario. It starts with a green circle representing the initial state, which leads to a green box labeled 'Fluffy Created'. From 'Fluffy Created', there is a self-loop arrow and a downward arrow to a blue box labeled 'Fluffy Sold'. From 'Fluffy Sold', there is a self-loop arrow and a downward arrow to a white box labeled 'Fluffy Deleted'. From 'Fluffy Deleted', there is a self-loop arrow.

Wiremock Inspector

WireMockInspector v1.0.30.0

http://localhost:9095

Requests Mappings Scenarios Settings

All ?

✓	2026-05-25 13:11:16.724	PUT	/api/v3/pet	200
✓	2026-05-25 13:11:12.907	GET	/api/v3/pet/	200
✓	2026-05-25 13:11:09.367	POST	/api/v3/pet	200
✓	2026-05-25 13:11:05.655	GET	/api/v3/pet/	404

Matching status: PerfectMatch
Mapping: 0a39eead-d202-4258-9f12-2ab0098afbd3
Mapping availability: Found

Request Response Code

Show options

C# (default) ClientIP Method Path Url

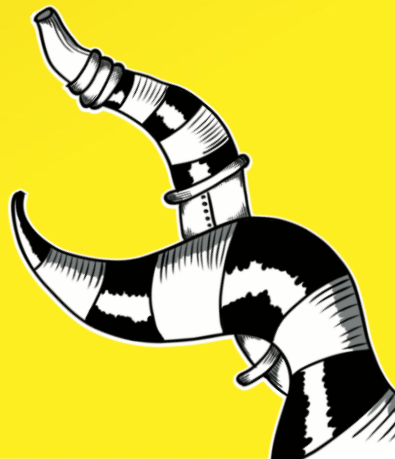
```
1 var mappingBuilder = new MappingBuilder();
2 mappingBuilder
3     .Given(Request.Create()
4         .UsingMethod("PUT")
5         .WithPath("/api/v3/pet")
6         .WithBodyAsJson(new
7             {
8                 id = 1000,
9                 name = "Fluffv"
```

Closing remarks

WireMock can also do a bit of “chaos engineering”: Induce faults and delays

There are of course alternatives like mocking features natively in your ERP / CRM / whatever environment

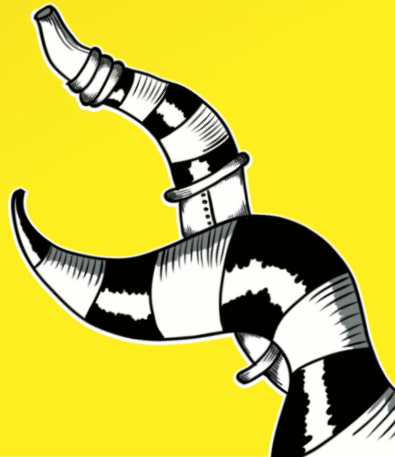
- If you only use that → WireMock is probably not the best fit
- If you are not afraid of a little JSON / Java / C# and/or you have additional clients with a need to mock your backend
→ WireMock is worth a look!



Closing remarks

What I hope you have learned today:

- You don't always need a real backend to develop against APIs
- WireMock works with either JSON or code (C# / Java)
- It supports recording, dynamic returns and stateful scenarios
- The client code does ideally need no changes at all



Questions?



THANK YOU!