

Event-Driven APIs:

Building Real-Time Interfaces



Uli Hitzel
API Evangelist
Axway



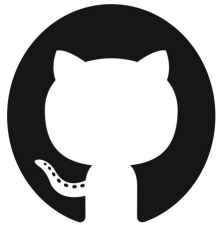
Sydney, September 2019



Developer Advocate



/in/uhitzel



/u1i/slides





cosmicrenaissance

1,257 posts

194 followers

615 following

Just some guy, really.

+65

City life, scenery, food.

Camera: Samsung Galaxy S7/S9



{ api }

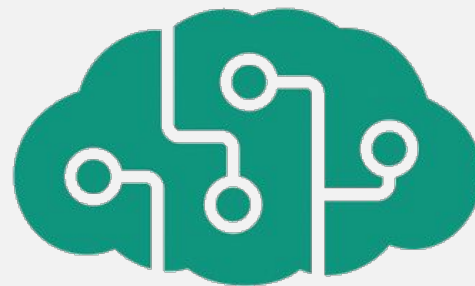
[illegible]



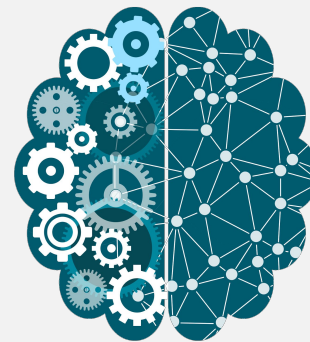
+



+



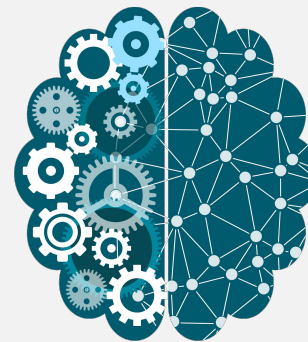
 Microsoft
Cognitive Services



Text: "a close up of a pizza"

Confidence: 0.9246481371443355

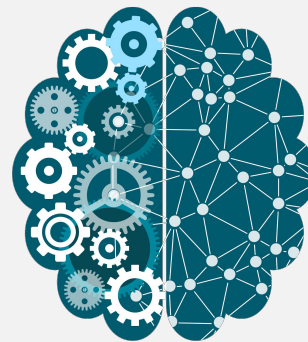
Tags: "pizza" "food" "table" "sitting" "cheese" "plate" "top" "piece" "slice" "toppings"
"close" "eaten" "sauce" "covered" "wooden" "large" "pepperoni" "white" "pan"



Text: "a cup of coffee"

Confidence: 0.8527612488821656

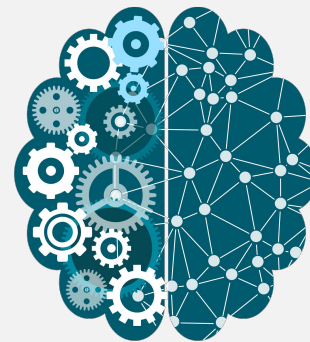
Tags: "cup" "table" "coffee" "food" "sitting" "drink" "plate" "sandwich" "breakfast"
"spoon" "glass" "close" "donut" "hot" "white" "phone" "soup"



Text: "a wooden table next to a window"

Confidence: 0.8365439206950018

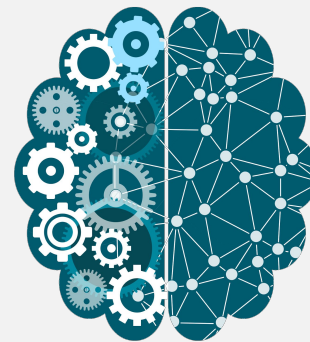
Tags: "table" "indoor" "window" "room" "sitting" "chair" "small" "desk" "wooden"
"laptop" "lit" "top" "light" "counter" "computer" "living" "kitchen"



Text: "a sunset over the ocean"

Confidence: 0.8065587199349568

Tags: "outdoor" "water" "beach" "sunset" "ocean" "sun" "man" "pier" "track"
"board" "top" "large" "red" "standing" "surfing" "walking" "boat" "holding" "flying"



Text: "a bunch of bananas"

Confidence: 0.37608016747604295

Tags: "animal" "covered" "sitting" "old" "side" "large" "water" "lot" "many" "top"
"table" "bunch" "street" "fire" "bird" "snow" "parking" "group" "river" "ocean" "field"

Agenda

1 – It's all about Developer Experience

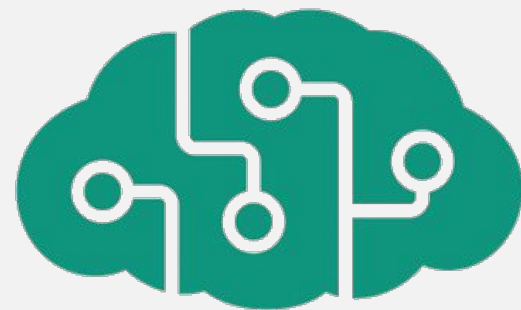
2 – Event-Driven APIs


3 – Demo



You've just seen a great Example

- API abstracts something really complex
- Quick Results
- Easy to use → More Users → More Sample Data → AI learns → Better Product



 Microsoft
Cognitive Services

```

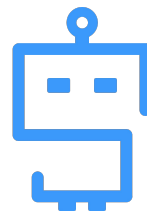
1  swagger: "2.0"
2  info:
3    description: "This is a sample server
      out more about Swagger at [http://swagger.io]
      or on [irc.freenode.net, #swagger]
      this sample, you can use the api to
      authorization filters."
4    version: "1.0.0"
5    title: "Swagger Petstore"
6    termsOfService: "http://swagger.io/terms/"
7    contact:
8      email: "apiteam@swagger.io"
9    license:
10     name: "Apache 2.0"
11     url: "http://www.apache.org/licenses/LICENSE-2.0.html"
12  host: "petstore.swagger.io"
13  basePath: "/v2"
14  tags:
15  - name: "pet"
16    description: "Everything about your pets"
17    externalDocs:
18      description: "Find out more"
19      url: "http://swagger.io"
20  - name: "store"
21    description: "Access to Petstore orders"
22  - name: "user"
23    description: "Operations about user"
24  externalDocs:
25    description: "Find out more about Swagger"
26    url: "http://swagger.io"

```

Design First

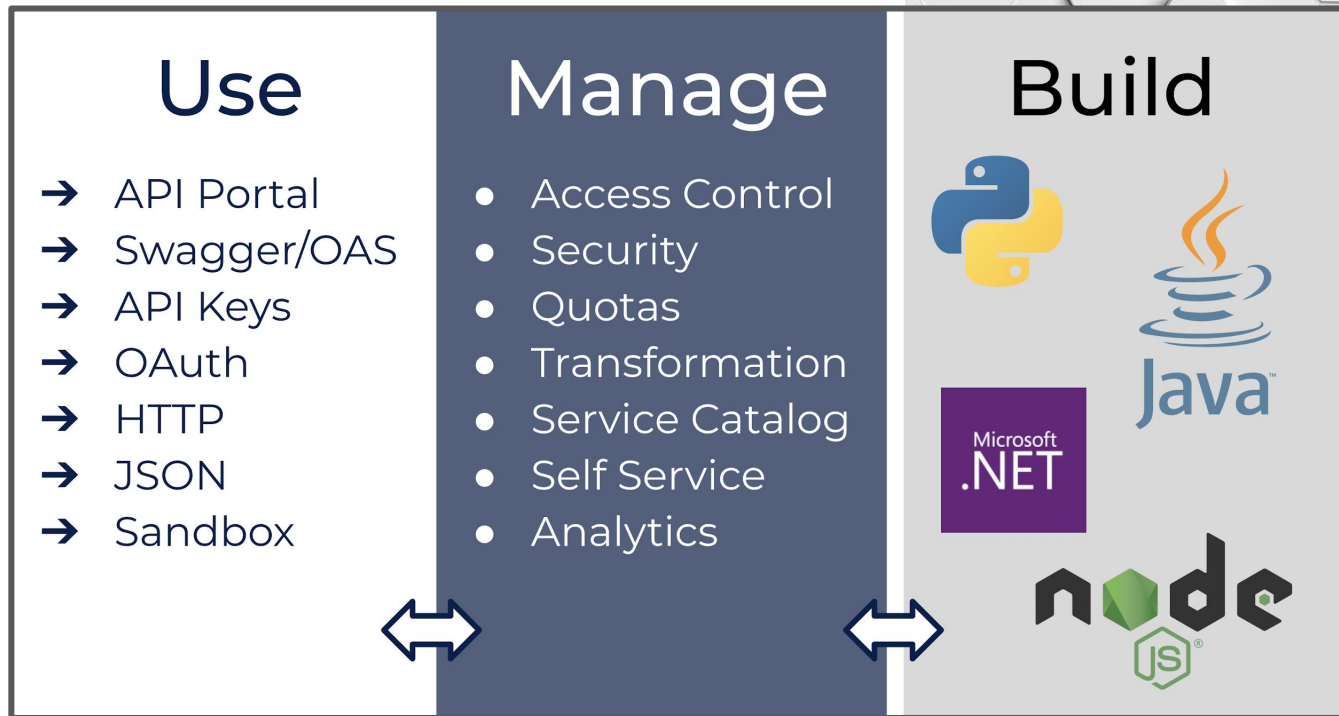


- What can I do?
- What do I get?
- Data Models
- Authentication
- License



Divide & Conquer

2





Think like a
Developer.

Think Use
Cases.



Event-Driven User Interfaces

1 EUR = 1.091245 USD

1 EUR = 1.091315 USD

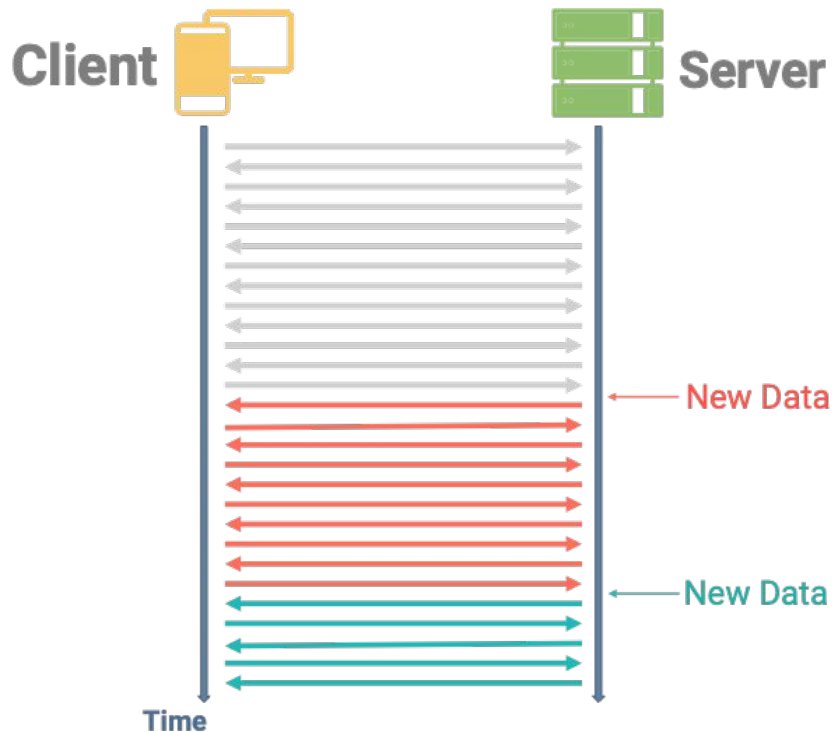
1 USD = 0.65501 GBP

1 USD = 0.655055 GBP

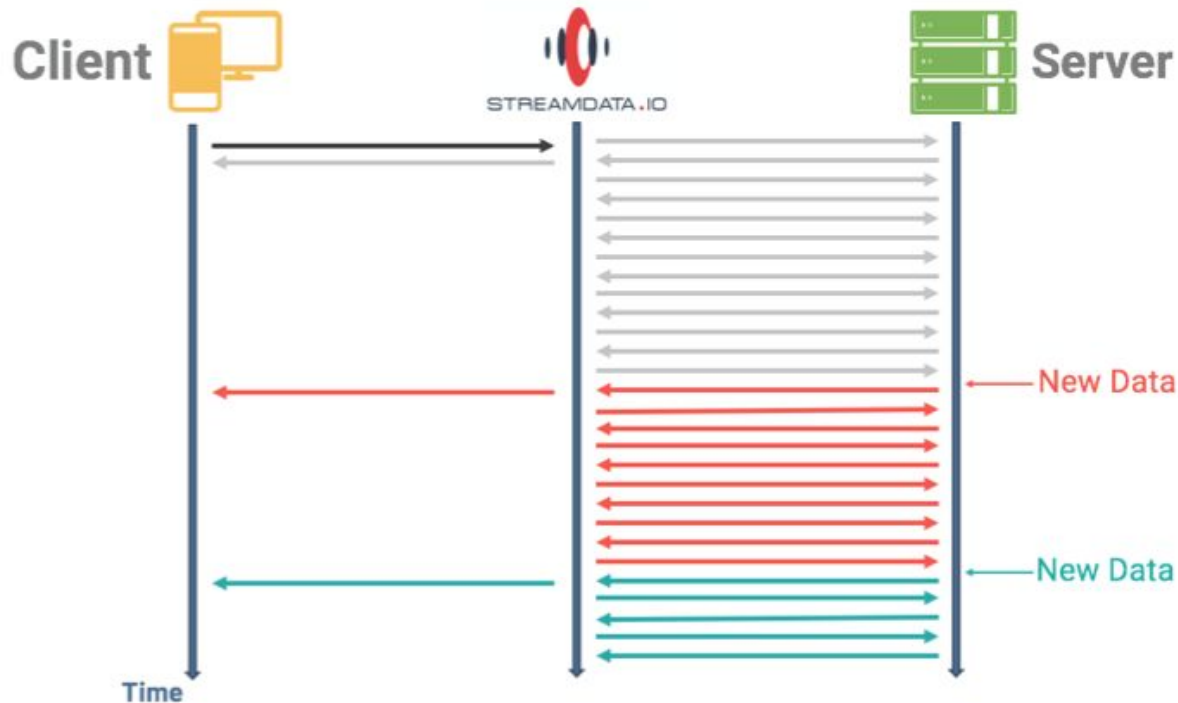
1 EUR = 135.522 JPY

1 EUR = 135.509 JPY

Approach: Continuous Polling



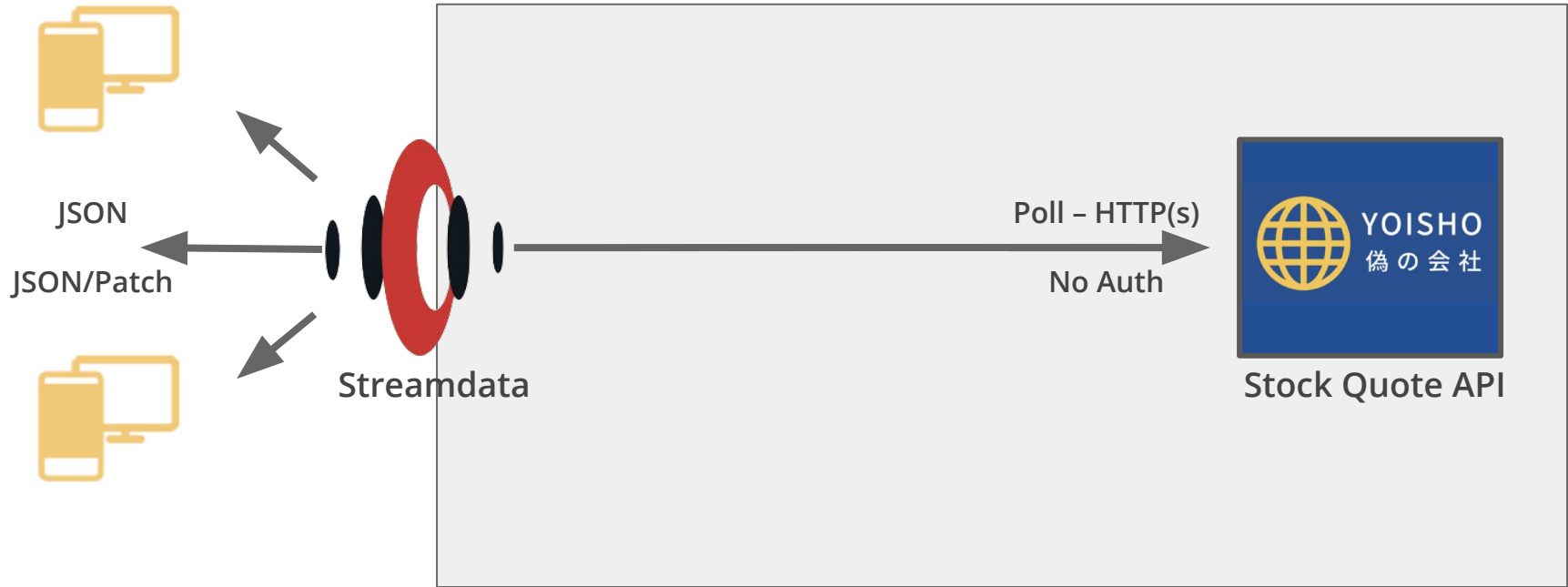
Better: Event-Driven





- Financial Sector Clients
- Airlines & Travel Systems
- Smarter Cities
- Media / News

Use Case – Setup





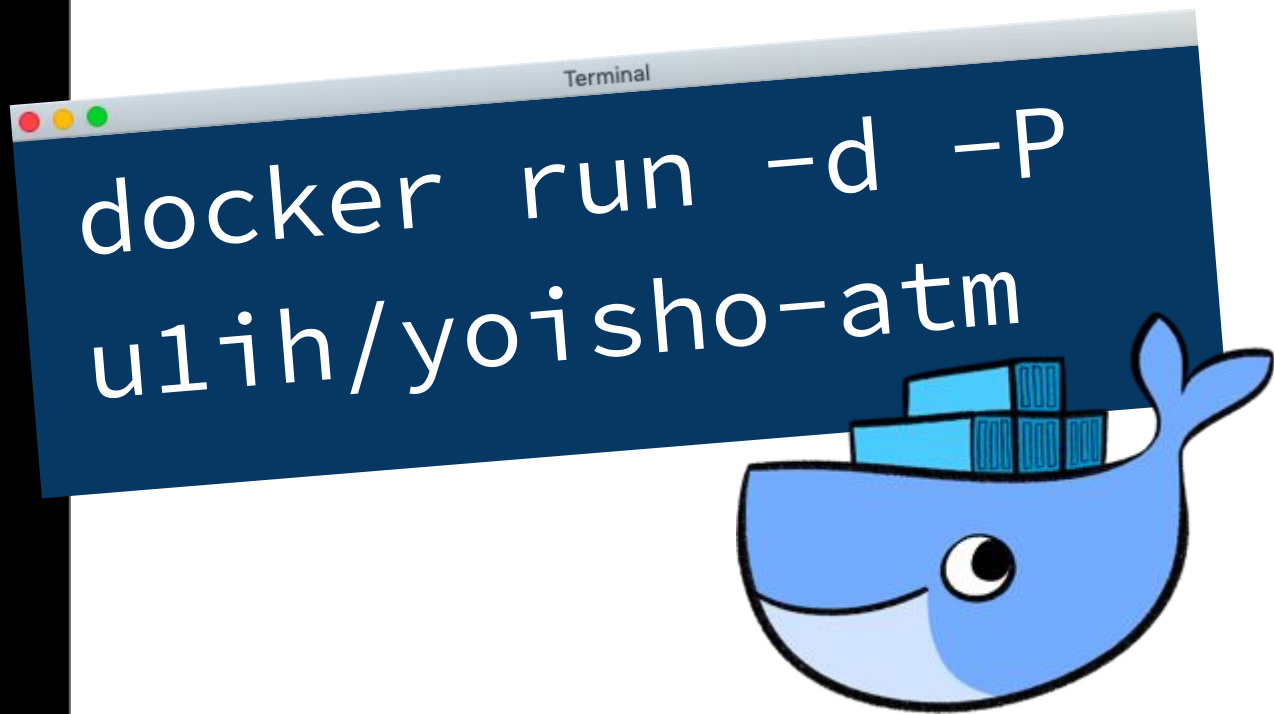
YOISHO
偽の会社

SOAP & REST
Webservices / APIs
for a fictional
japanese bank. Easy
to run docker
images that expose
endpoints you can
use for demos and
testing.



View On GitHub

Open Banking APIs



Open Banking - Live Endpoints

Webservices and REST APIs that expose bank related data services with dynamic content, use them for testing and demos. Available over [HTTP](#) and [HTTPS](#)

[Full Documentation](#)



ATM

Full CRUDL REST interface (Create, Read, Update, Delete, List), /v1 and /v2 endpoints with respective Swagger specs

[API Version 1 - Swagger](#)

[API Version 1 - Swagger Editor](#)

[API Version 2 - Swagger](#)

[API Version 2 - Swagger Editor](#)

[Sample Request](#)

Currency Exchange

REST/JSON, 1 parameter. Response is different each time.

[Swagger](#)

[Swagger Editor](#)

[Sample Request](#)

Assets (SOAP)

Bank Assets & Debt- SOAP/XML, 2 methods, dynamic output.

[WSDL](#)

Open Banking - Live Endpoints

Webservices and REST APIs that expose bank related data services with dynamic content, use them for testing and demos. Available over [HTTP](#) and [HTTPS](#)

[Full Documentation](#)



Fixed Deposit

Calculator - REST/JSON, 2 parameters, semantic error handling, complex output

[Swagger](#)

[Swagger Editor](#)

[Sample Request](#)

Investment

/v1 and /v2 endpoints with respective Swagger specs. Static responses.

[API Version 1 - Swagger](#)

[API Version 1 - Swagger Editor](#)

[API Version 2 - Swagger](#)

[Swagger Editor](#)

[API Version 2 - Sample Request](#)

Stock Quote

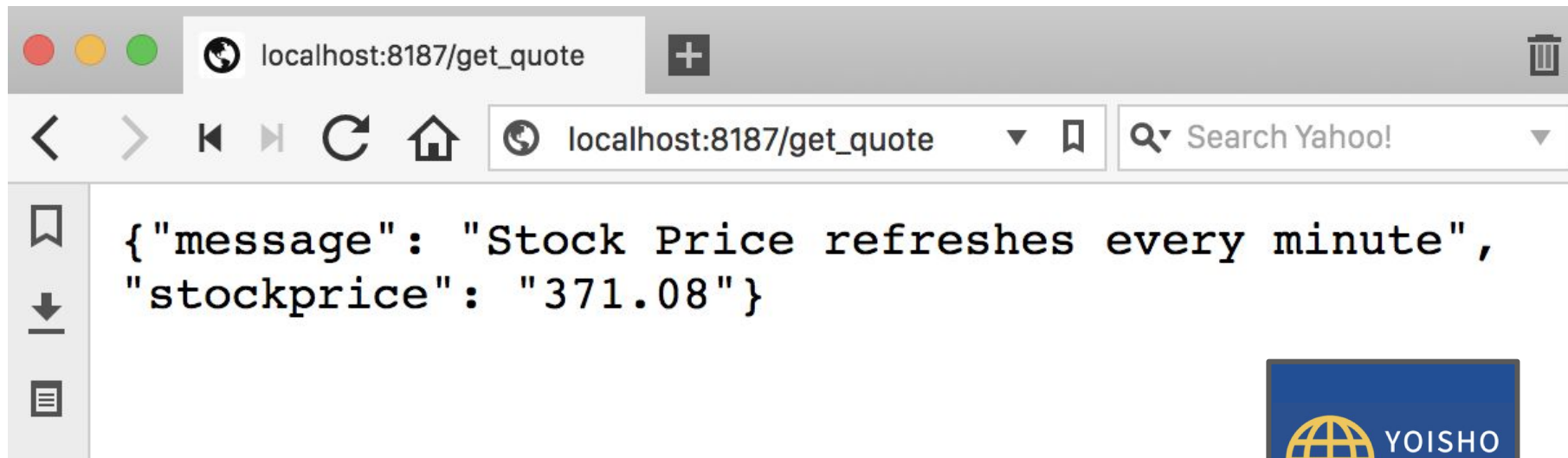
Returns the stock price. Data updates once every minute, it's also sending additional Cache-Control headers.

[Swagger](#)

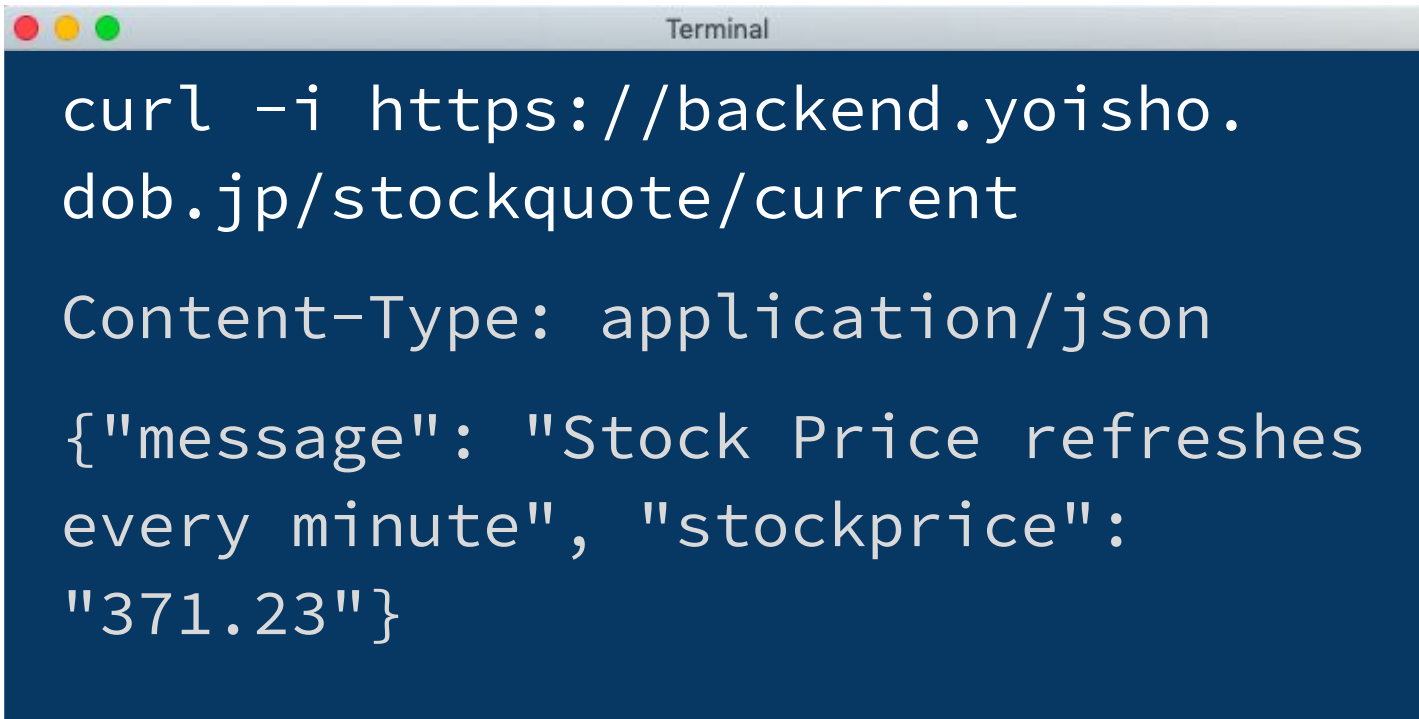
[Swagger Editor](#)

[Sample Request](#)

API Backend: Stock Quote



Example with cURL

A screenshot of a macOS Terminal window. The title bar is light gray with three colored window control buttons (red, yellow, green) on the left and the word "Terminal" in the center. The main area has a dark blue background with white text. The text shows a cURL command being executed, followed by the headers and the JSON response from the server.

```
curl -i https://backend.yoisho.dob.jp/stockquote/current

Content-Type: application/json

{"message": "Stock Price refreshes every minute", "stockprice": "371.23"}
```

Creating a Streaming API

Add New API

URL

GET

Polling Frequency ⓘ


seconds ▾

HTTP Headers ⓘ

+ Add Http Header

Query Params ⓘ

+ Add Query Param

 **streamdata.io**
by axway

Cancel

+ Add

We get a 'proxied' API endpoint

`https://streamdata.motwin.net/https://backend.yoisho.dob.jp/stockquote/current?X-Sd-Token=ZTx
lMHZmNzgtN2UxZS00NDZhLWEzM1QtZT
MxMmFHMzJl7DQy`

Consuming the new API

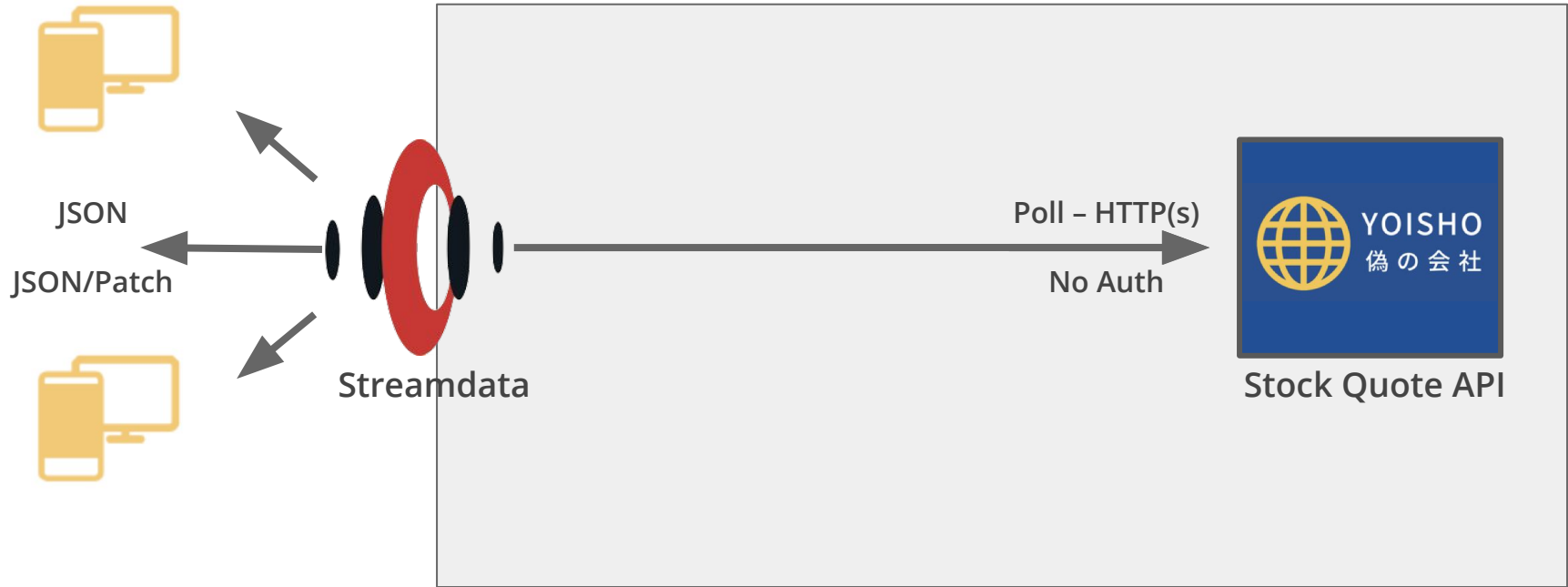


```
Terminal
curl https://streamdata[...]
Content-Type: text/event-stream
event: data
data: {"stockprice": "371.77"}
event: patch
data: [{"op": "replace", "path": "/stock price", "value": "371.74"}]
```

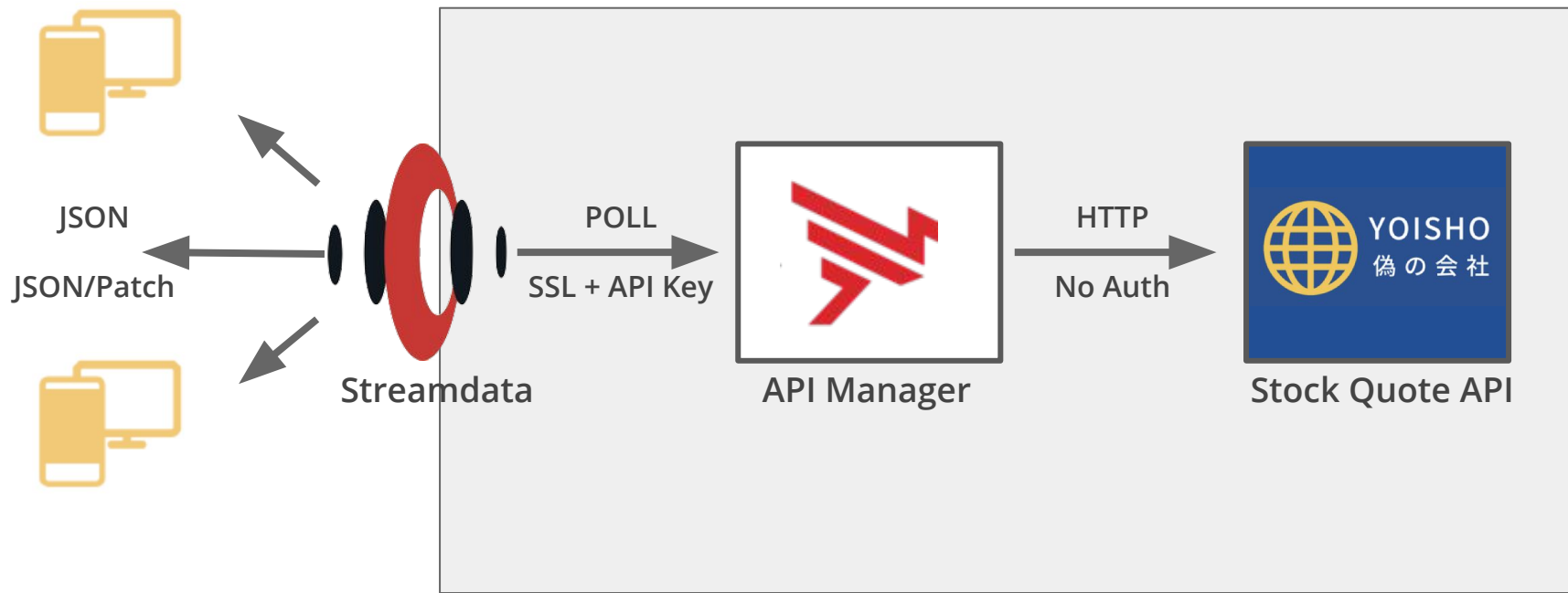
The image shows a terminal window with a dark blue background. The title bar of the window is light gray and contains the word "Terminal" in the center. On the left side of the title bar are three colored window control buttons: red, yellow, and green. The terminal displays the output of a curl command. The first line is the command itself. The second line is the "Content-Type" header, which is highlighted with a red rectangular box. The third line is "event: data". The fourth line is a JSON object representing stock price data. The fifth line is "event: patch", where the word "patch" is highlighted with a red rectangular box. The sixth line is another JSON array representing an update operation. A red arrow points from a red box labeled "Server-Sent-Events" to the "Content-Type" header. Another red arrow points from the same box to the "patch" event.

Server-Sent-Events

Use Case – Setup



Add Security



Your Turn!

- **Use:** Uli's Open Banking APIs
- **Try it out:** streamdata.io
- **Join & Share:** Participate at Meetups & Get Involved



/u1i

