InfinyOn

Data Wrangling using Fluvio's WASM-powered SmartModules

Sebastian Imlay

InfinyOn

Introduction

- Me
  - @sebimlay on 🐦
  - @simlay@hachyderm.io on 🐘
  - simlay on GitHub
- Code/presentation
  - https://github.com/simlay/presentations
- Work on Fluvio at InfinyOn

InfinyOn

Overview

High Performer SUMMER 2022

- Describe Fluvio
  - Consumer Streams
  - Producer Streams
  - Using WASM SmartModules
- Demo
  - Ingest/transform data from Github API using a Fluvio Connector
  - Consume Filter and display data via WASM

InfinyOn

High Performer SUMMER 2022

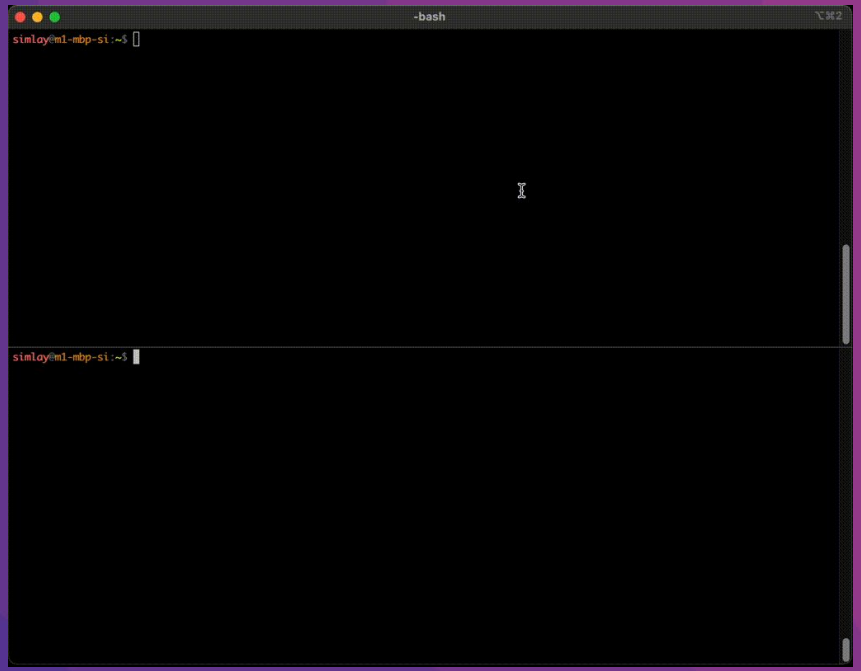
What is Fluvio

- Produce and consume on a topic
- Low latency record streaming
- Written in Rust
- Client Libraries
  - Python
  - Java
  - Nodejs
  - Browser JavaScript (compile to wasm)

InfinyOn

High Performer
SUMMER
2022

Fluvio CLI Demo

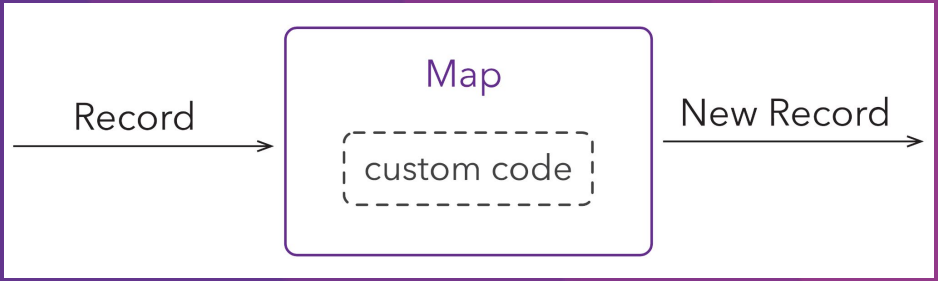


InfinyOn

High Performer SUMMER 2022

Fluvio SmartStreams

- A WebAssembly Module with a special entry point
  - Aggregate/Filter/FilterMap/ etc.
- Applied on the server of a Consumer Stream



InfinyOn

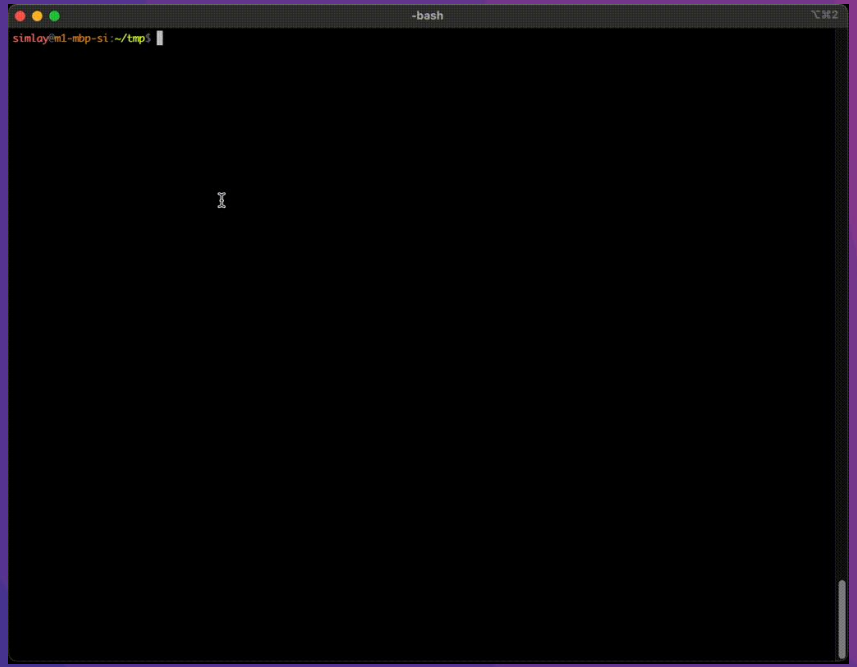
High Performer SUMMER 2022

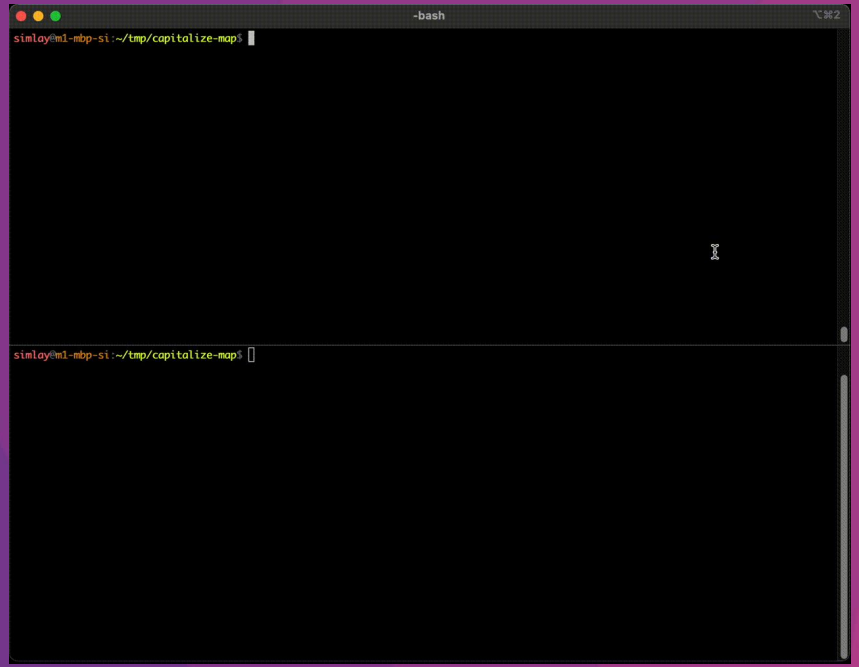
Use SMDK

- SMDK - Smart Module Development Kit
  - A nice wrapper around cargo-generate
  - Makes use of Fluvio Hub (a repository for SmartModules)

InfinyOn

SmartModule CLI Demo





InfinyOn

High Performer SUMMER 2022

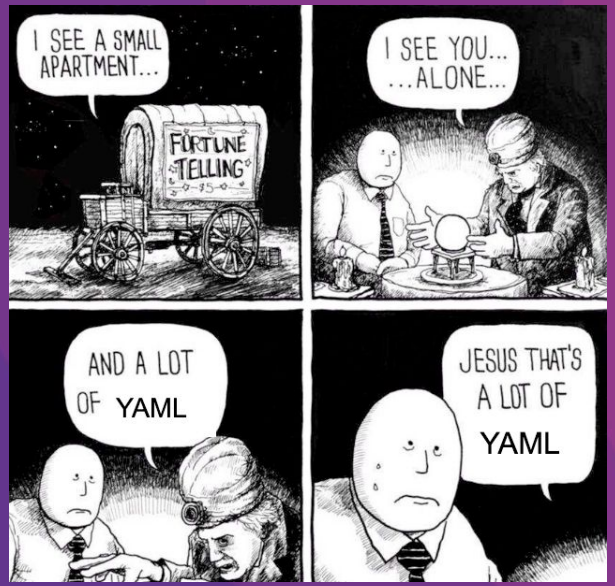
Capitalize Smart Module

```rust
use fluvio_smartmodule::{smartmodule, Result, Record, RecordData};

#[smartmodule(map)]
pub fn map(record: &Record) -> Result<(Option<RecordData>, RecordData)> {
    let key = record.key.clone();

    let value = std::str::from_utf8(record.value.as_ref())?.to_uppercase();

    Ok((key, value.into()))
}
```

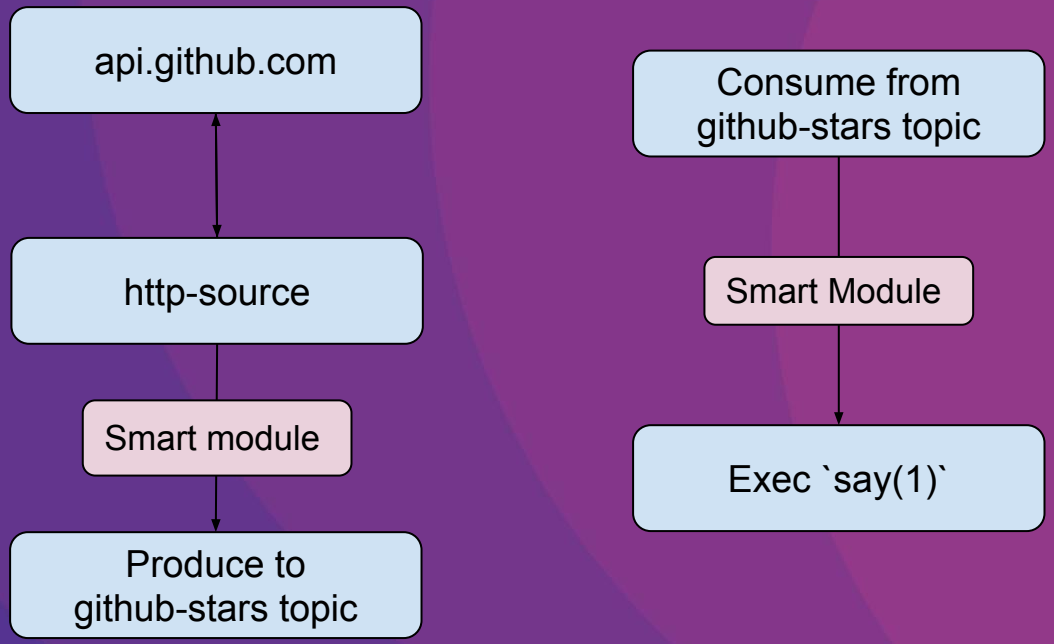
InfinyOn

Fluvio Connectors

- A pre-built fluvio client meant for low-code/no-code setup
- Run on Fluvio Cloud nicely



Demo Overview



InfinyOn

Input Connector

High Performer SUMMER 2022

```yaml
name: github-stars-input
type: http-source
version: '0.4.1'
topic: github-stars
create_topic: true
parameters:
  endpoint: 'https://api.github.com/repos/infinyon/fluvio'
  method: GET
  interval: '1s'
  header: 'Authorization: token <YOUR GITHUB TOKEN>'
transform:
  - uses: simlay/github-stars-aggregate@0.1.0
```

InfinyOn

High Performer SUMMER 2022

SmartModules

```rust
use fluvio_smartmodule::{smartmodule, Record, RecordData, Result as EyreResult};
mod model;
use model::GithubStars;

#[smartmodule(aggregate)]
pub fn aggregate(accumulator: RecordData, current: &Record) -> EyreResult<RecordData> {
    let mut accumulated_stars: GithubStars =
        serde_json::from_slice(accumulator.as_ref()).unwrap_or_default();

    let current_stars: GithubStars = serde_json::from_slice(current.value.as_ref())?;
    accumulated_stars.star_update =
        accumulated_stars.stargazers_count != current_stars.stargazers_count;
    accumulated_stars.stargazers_count = current_stars.stargazers_count;

    let accumulated_stars: RecordData = accumulated_stars.try_into()?;
    Ok(accumulated_stars)
}
```

```rust
use fluvio_smartmodule::{smartmodule, Record, RecordData, Result};

use model::GithubStars;

#[smartmodule(filter_map)]
fn filter_map(record: &Record) -> Result<Option<(Option<RecordData>, RecordData)>> {
    let stars = serde_json::from_slice::<GithubStars>(record.value.as_ref())?;

    if stars.star_update {
        let stars = format!("Fluvio Github Star count is now {}", stars.stargazers_count);
        Ok(Some((record.key.clone(), stars.into())))
    } else {
        Ok(None)
    }
}
```

Put it together!

```
simlay@m1-mbp-si:~/tmp$ fl
```

InfinyOn

Future Work?

High Performer SUMMER 2022

- Make use of WebAssembly Interface Types?
  - Could be used for shared smart module libraries
- Compile fluvio client to wasm-wasi for a cross-platform client?
  - wasi-sockets

InfinyOn

High Performer
SUMMER
2022

Questions?