Real-Time in PHP

Introducing the Mercure protocol

Drupal mountain camp 2022

Rainer Friederich

25. June 2022

About

Rainer Friederich

I’m a software engineer at Wondrous in Basel.

Real-time in web apps

Why does it matter?

01

Expectations

Users expect real-time UX. Without it your app appears broken to younger users.

02

Enablement

Some features require real-time communication capabilities.

03

Prevention

Real-time capabilities allow the application to warn the user upfront of actions to increase the UX.

Intro

What this talk is about

What is the state of real-time capabilities and PHP

The mercure protocol in its standard implementation

The mercure protocol and Drupal

Mercure vs its competitors

Wrap-up

What is the state of real-time capabilities and PHP

Intro

Real-time capabilities in PHP

- Theoretically real-time is possible with PHP
- It just does not scale and behave well and therefore the available libraries are limited
- ReactPHP + PHP 8.1 fibers are not fully ready for mainstream usage in big monolithic applications.

Framework-X : https://framework-x.org/

Intro

Why is PHP such a bad fit for real-time communication?

- Bootstrapping has to be done for each request, also if there is an active websocket connection.
- Websockets use UDP broadcasting while PHP was built for the HTTP request & response principle

Some available options

For a Drupal site

01

Ajax request

02

Long polling

03

Direct websockets or SSE

04

Build standalone app

05

Socket.io

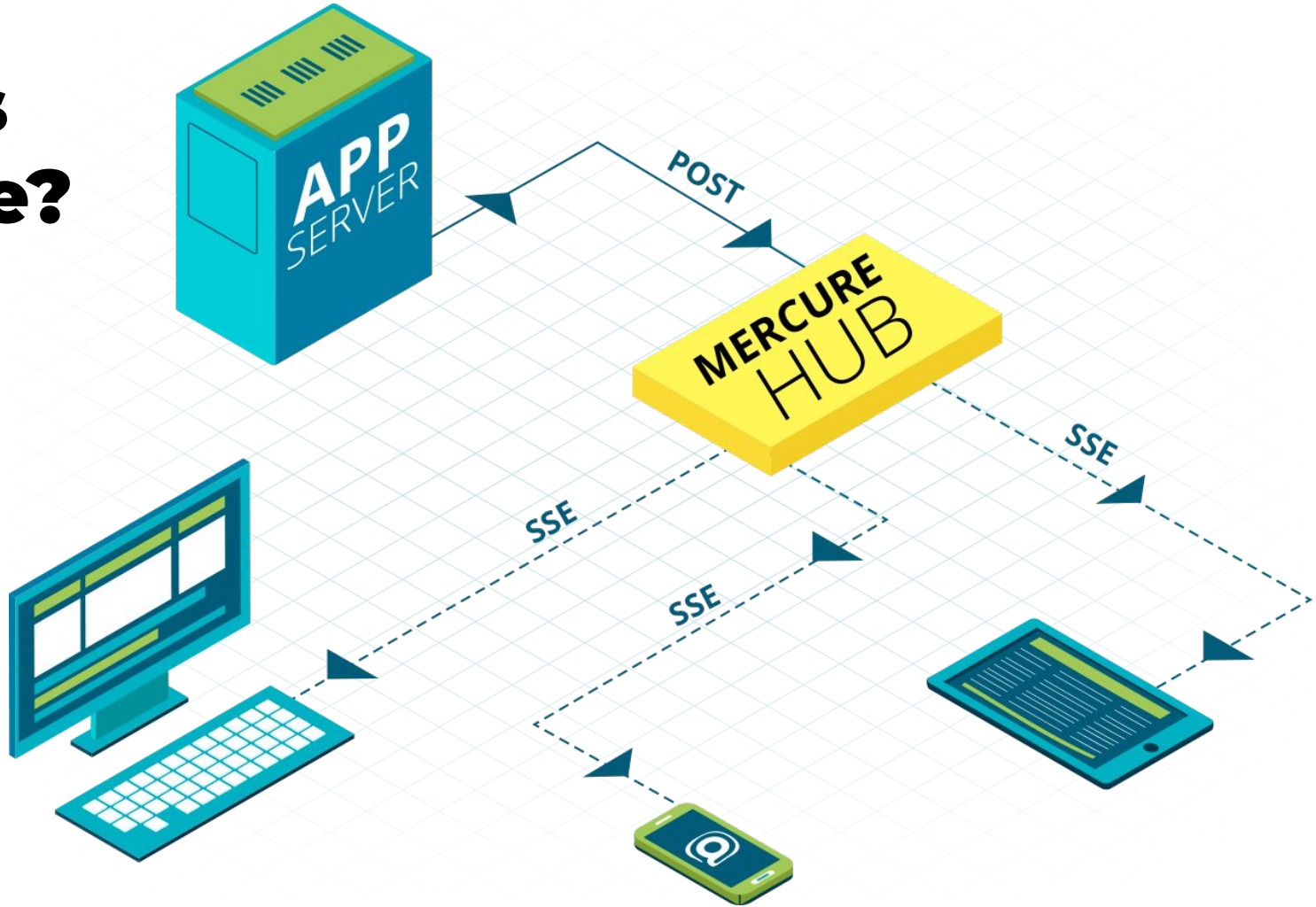
06

Mercure

Real time guide : https://www.leggetter.co.uk/real-time-web-technologies-guide/
PHP Mercure : https://github.com/bpolaszek/freddie
Socket.io: https://github.com/socketio/socket.io

The mercure protocol in its standard implementation

What is Mercure?



Specs and features

01

Official Symfony

02

Official web standard

03

DX and UX friendly

04

E2E encryption possible

05

Works with Rest/JSON and GraphQL

06

Performance

06

Security and privacy

Default implementation in api-platform

```php
<?php
// api/src/Entity/Book.php

namespace App/Entity;

use ApiPlatform\Core\Annotation\ApiResource;

#[ApiResource(mercure: true)]
class Book
{
  // ...
}
```

api-platform: https://api-platform.com/

The mercure protocol and Drupal

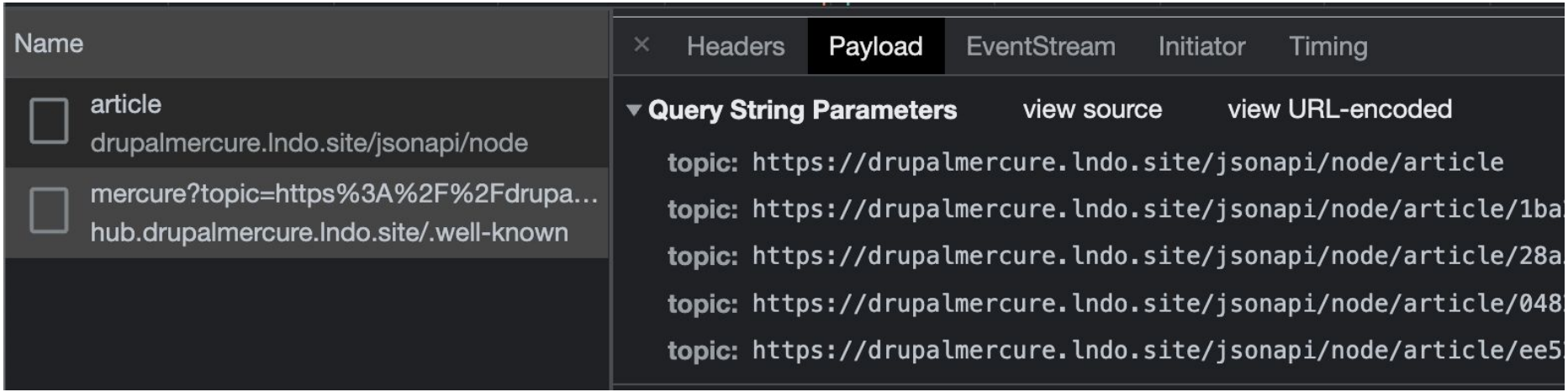

drupal/.lando.yml
services:
mercure:
app_mount: false
services:
command: caddy run -config /etc/caddy/Caddyfile.dev
environment:
MERCURE_PUBLISHER_JWT_KEY: '!ChangeMe!'
MERCURE_SUBSCRIBER_JWT_KEY: '!ChangeMe!'
SERVER_NAME: ':80'
GLOBAL_OPTIONS: 'auto_https off'
image: dunglas/mercure
ports:
- 1337:80
type: compose

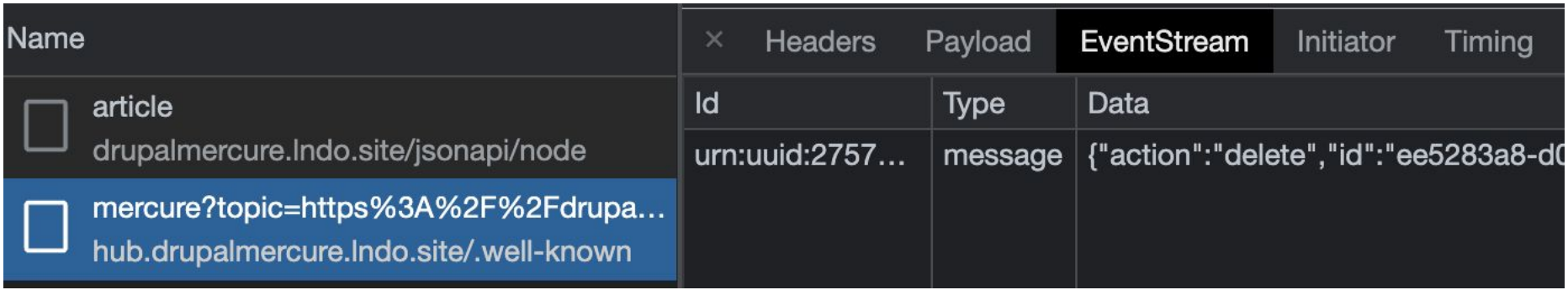

drupal/web/sites/default/services.yml
parameters:
  mercure:
    hubs:
      default:
        url: 'http://hub.drupalmercure.lndo.site/.well-known/mercure'
        jwt:
          secret: '!ChangeMe!'

```php
<?php
// drupal/web/modules/custom/article_updates/src/ArticleUpdateService.php

namespace Drupal\article_updates;

use Drupal\node\NodeInterface;
use Symfony\Component\Mercure\HubInterface;
use Symfony\Component\Mercure\Update;

class ArticleUpdatesService {
  public function __construct(private HubInterface $mercure) {}

  public function deleteUpdate(NodeInterface $node): void {
    $topic = 'https://drupalmercure.lndo.site/jsonapi/node/' . $node->getType() . '/' . $node->uuid();

    $update = new Update($topic, json_encode(
      [
        'action' => 'delete',
        'id' => $node->uuid(),
        'data' => []
      ]
    ));

    $this->mercure->publish($update);
  }
}
```

```php
<?php
// drupal/web/modules/custom/article_updates/article_updates.module

function article_updates_entity_delete(EntityInterface $entity): void {
  if (!$entity instanceof NodeInterface || $entity->bundle() !== 'article') {
    return;
  }

  Drupal::service('article_updates_service')->deleteUpdate($entity);
}
```

```
// client/js/main.js

const mercureBaseUrl = 'http://hub.drupalmercure.lndo.site/.well-known/mercure';
const apiBaseUrl = 'https://drupalmercure.lndo.site'
const mercureUrl = new URL(mercureBaseUrl);
mercureUrl.searchParams.append('topic', apiBaseUrl + '/jsonapi/node/article');

axios.get(apiBaseUrl + '/jsonapi/node/article')
  .then(function (response) {
    response.data.data.forEach(function(article) {
      mercureUrl.searchParams.append(
        'topic',
        apiBaseUrl + '/jsonapi/node/article/' + article.id
      );
    })

    const eventSource = new EventSource(mercureUrl);

    eventSource.onmessage = function(event) {
      const responseData = JSON.parse(event.data)
      switch (responseData.action) {
        case "delete":
          document.getElementById(responseData.id).remove();
          break;
      }

    }
  })
```


Name
article
drupalmercure.lndo.site/jsonapi/node
mercure?topic=https%3A%2F%2Fdrupa…
hub.drupalmercure.lndo.site/.well-known
Headers
Payload
EventStream
Initiator
Timing
Query String Parameters
view source
view URL-encoded
topic: https://drupalmercure.lndo.site/jsonapi/node/article
topic: https://drupalmercure.lndo.site/jsonapi/node/article/1ba
topic: https://drupalmercure.lndo.site/jsonapi/node/article/28a
topic: https://drupalmercure.lndo.site/jsonapi/node/article/048
topic: https://drupalmercure.lndo.site/jsonapi/node/article/ee5


Name
article
drupalmercure.lndo.site/jsonapi/node
mercure?topic=https%3A%2F%2Fdrupa…
hub.drupalmercure.lndo.site/.well-known
Headers
Payload
EventStream
Initiator
Timing
Id
Type
Data
urn:uuid:2757…
message
{"action":"delete","id":"ee5283a8-d0

Mercure vs its competitors

SSE Gotchas

01

No bi-directional communication

02

No binary support

03

Maximum concurrent connections

SSE Wins

01

Pareto solution

02

Performance

03

Consistency

04

Based on web standards

05

Open source

06

Infrastructure as code

When to use alternatives

01

P2P updates

02

Eventual consistency

03

Bi-directional communication

04

External solution prefered

05

Different programming language is used.

06

Conflict solving (gdocs) needed

07

Native mobile notifications needed

Wrap-up

Wrap-up

Conclusion

01

It does not always needs to be websockets and pusher protocol

02

Real-time updates are possible with PHP without SaaS additions

03

For Drupal and Symfony SSE and Event source together with Mercure are a very good fit.

Wrap-up

Mercure in action

If you want to try it out check out the public repo on my github which contains all the code you have seen in the presentation and more.

https://github.com/yobottehg/drupal-mercure

Useful links

Ready to use Kubernetes setup with Symfony and Mercure:
https://github.com/api-platform/api-platform/releases

Mercure specifications:
https://mercure.rocks/spec

Mercure Drupal module:
https://www.drupal.org/project/mercure

Mercure Symfony component:
https://symfony.com/doc/current/mercure.html

Thanks for listening, Questions?