

Look Ma', I made a website!

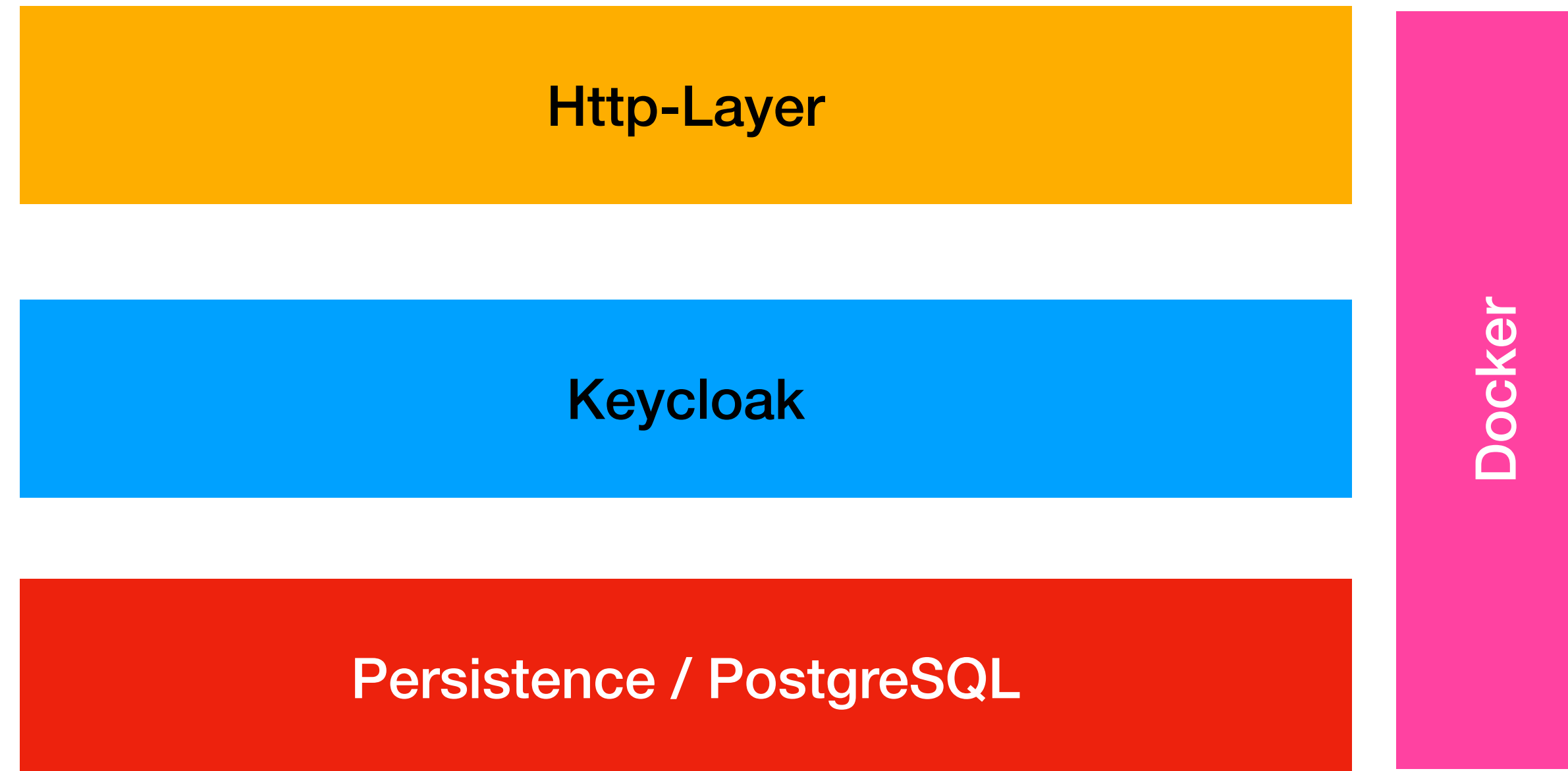
Keycloak, Meetup, Quarkus, Wordpress

Dominik Dorn, 3.10.2022

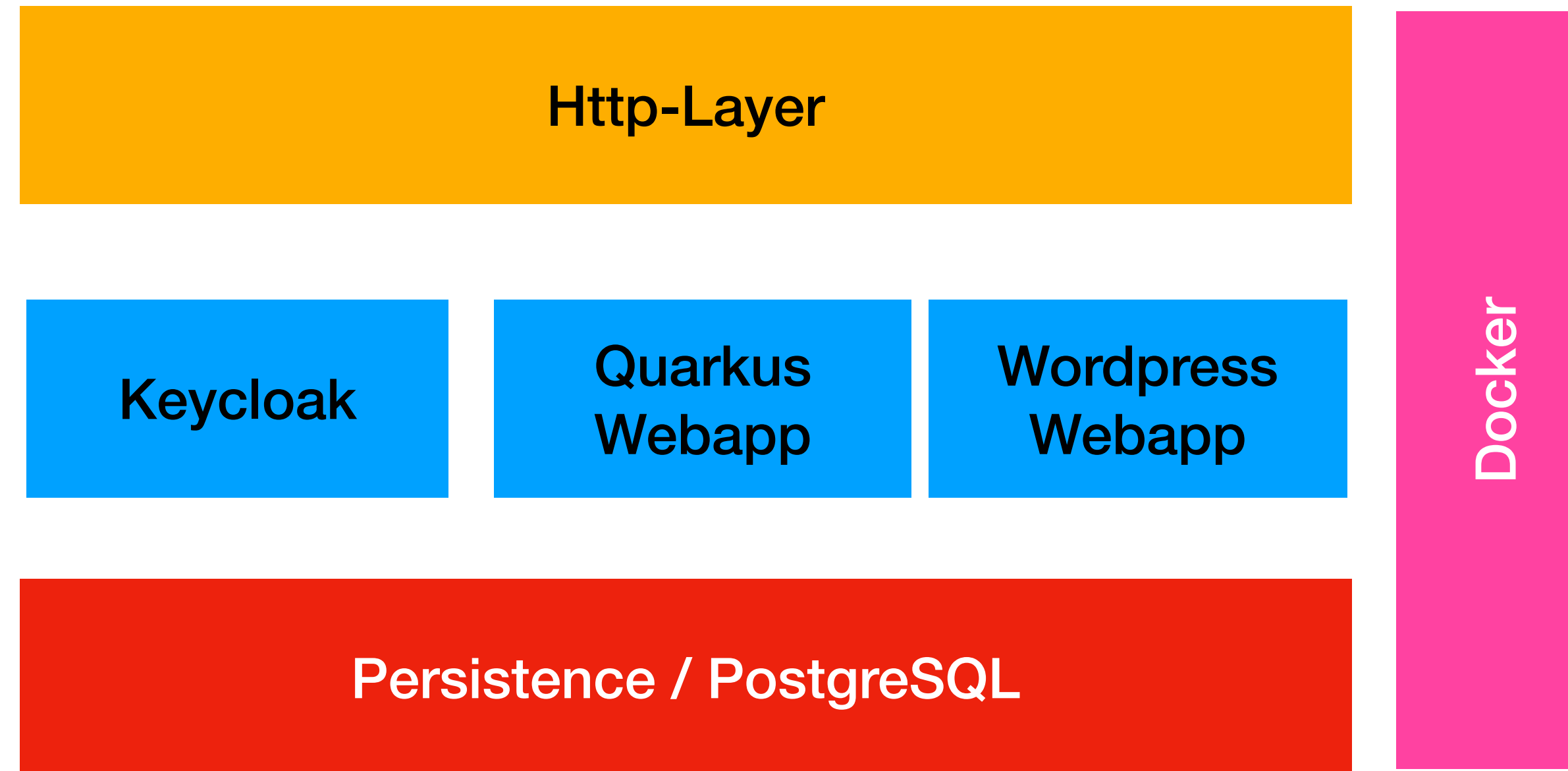
Agenda

- Keycloak in Docker
- Meetup API
- Keycloak Meetup Integration / Keycloak IdP
- Integrating IdP into Keycloak in Docker
- Keycloak Integration in Wordpress
- Quarkus Sample

Keycloak in Docker

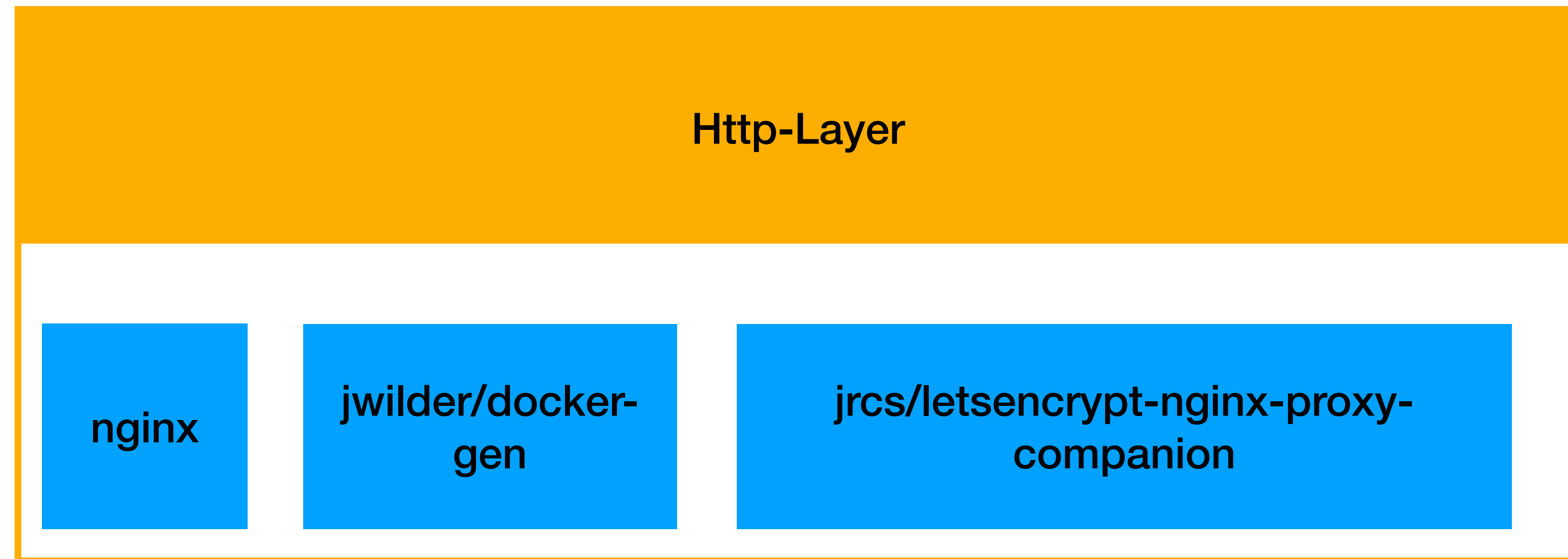


Keycloak in Docker



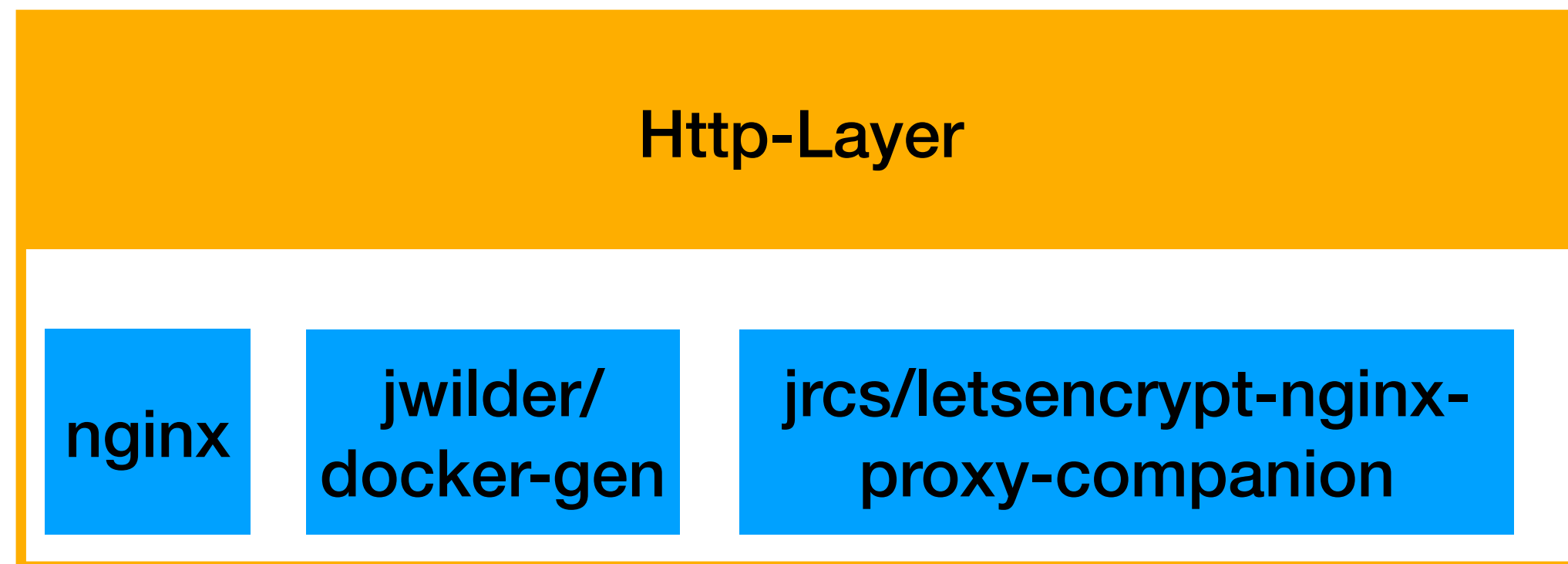
Keycloak in Docker

Basic setup without
k8s/openshift/others



Keycloak in Docker

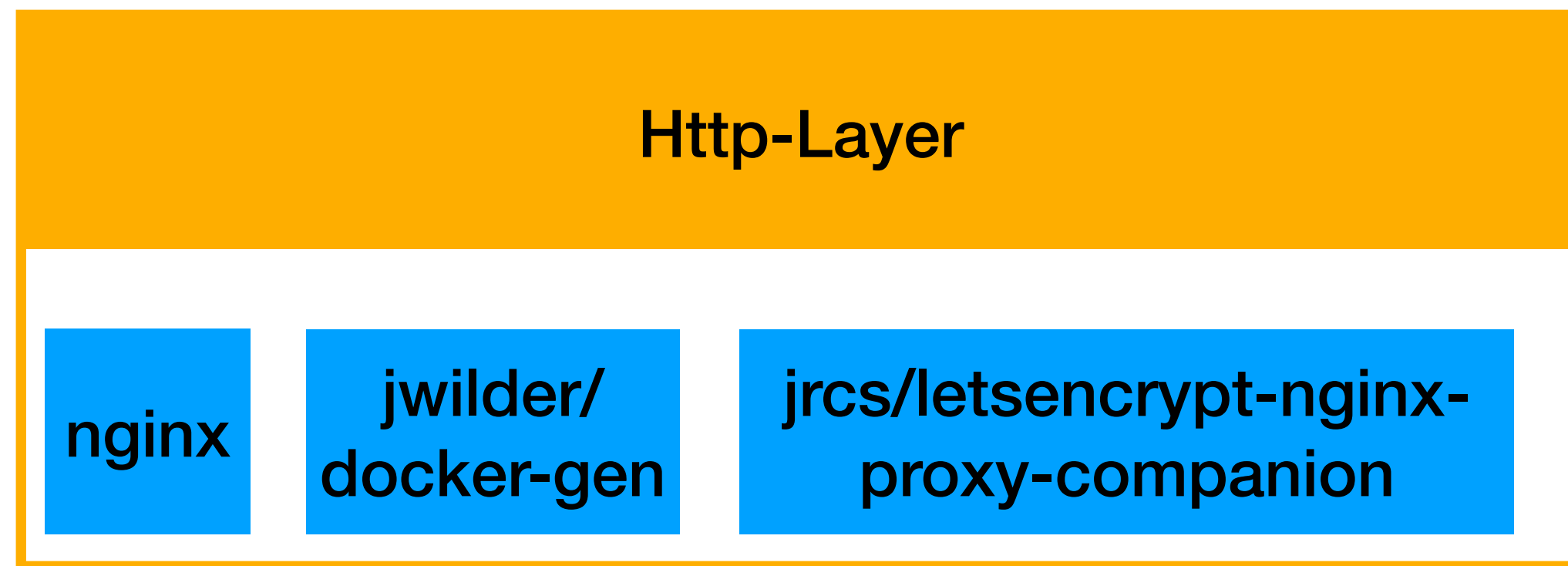
Basic setup without k8s/openshift/others



```
services:
  nginx:
    image: nginx
    container_name: nginx
    restart: always
    ports:
      - "188.40.54.119:80:80"
      - "188.40.54.119:443:443"
    volumes:
      - "/data/nginx/conf.d:/etc/nginx/conf.d"
      - "/data/nginx/vhost.d:/etc/nginx/vhost.d"
      - "/data/nginx/html:/usr/share/nginx/html"
      - "/data/nginx/certs:/etc/nginx/certs:ro"
    networks:
      - nginx-front
```

Keycloak in Docker

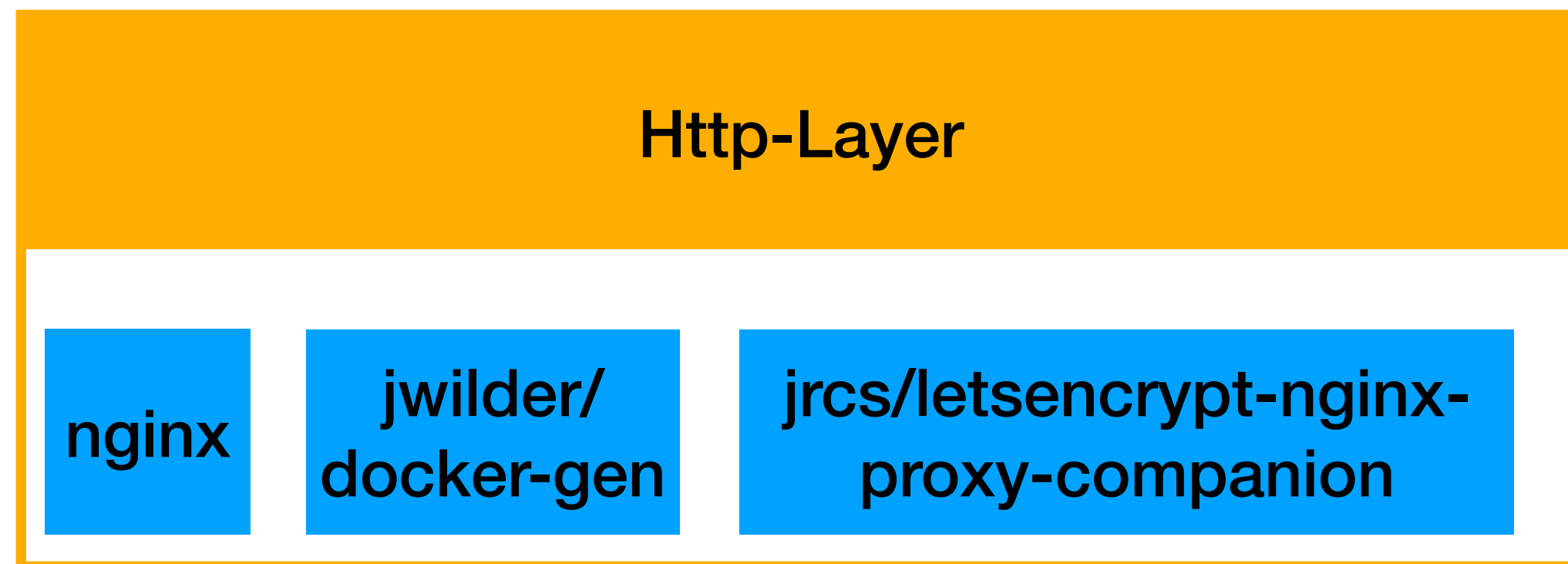
Basic setup without k8s/openshift/others



```
nginx-gen:
  image: jwilder/docker-gen
  container_name: nginx-gen
  mem_limit: 512m
  restart: always
  volumes:
    - "/var/run/docker.sock:/tmp/docker.sock:ro"
    - "/data/nginx/conf.d:/etc/nginx/conf.d"
    - "/data/nginx/vhost.d:/etc/nginx/vhost.d"
    - "/data/nginx/html:/usr/share/nginx/html"
    - "/data/nginx/certs:/etc/nginx/certs:ro"
  volumes_from:
    - nginx
  entrypoint: >
    /usr/local/bin/docker-gen
    -notify-sighup nginx
    -watch -wait 5s:30s
    /etc/docker-gen/templates/nginx.tpl
    /etc/nginx/conf.d/default.conf
```

Keycloak in Docker

Basic setup without k8s/openshift/others

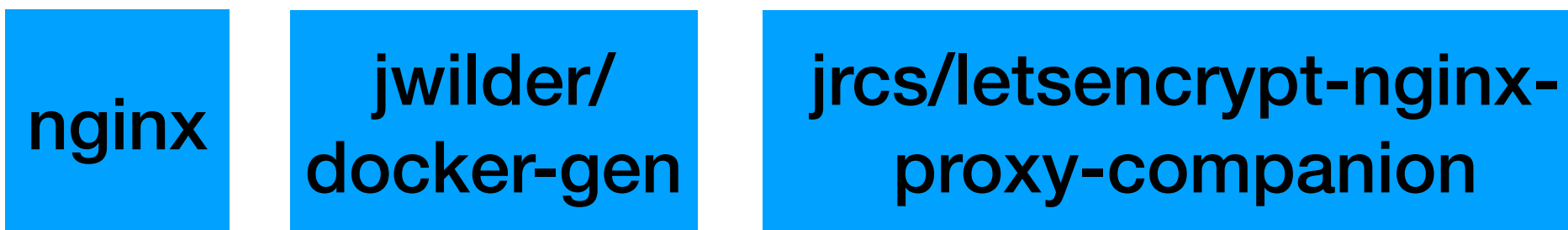


```
letsencrypt-nginx-proxy-companion:  
  image: jrcs/letsencrypt-nginx-proxy-companion  
  container_name: letsencrypt-nginx-proxy-companion  
  restart: always  
  mem_limit: 192m  
  volumes_from:  
    - nginx  
  volumes:  
    - "/var/run/docker.sock:/var/run/docker.sock:ro"  
    - "/data/nginx/certs:/etc/nginx/certs:rw"  
    - "/data/nginx/html:/usr/share/nginx/html"  
  environment:  
    - NGINX_DOCKER_GEN_CONTAINER=nginx-gen
```


Keycloak in Docker

Basic setup without k8s/openshift/others

Http-Layer



```
letsencrypt-nginx-proxy-companion:
  image: jrcs/letsencrypt-nginx-proxy-companion
  container_name: letsencrypt-nginx-proxy-companion
  restart: always
  mem_limit: 192m
  volumes_from:
    - nginx
  volumes:
    - "/var/run/docker.sock:/var/run/docker.sock:ro"
    - "/data/nginx/certs:/etc/nginx/certs:rw"
    - "/data/nginx/html:/usr/share/nginx/html"
  environment:
    - NGINX_DOCKER_GEN_CONTAINER=nginx-gen
```

```
services:
  nginx:
    image: nginx
    container_name: nginx
    restart: always
    ports:
      - "188.40.54.119:80:80"
      - "188.40.54.119:443:443"
    volumes:
      - "/data/nginx/conf.d:/etc/nginx/conf.d"
      - "/data/nginx/vhost.d:/etc/nginx/vhost.d"
      - "/data/nginx/html:/usr/share/nginx/html"
      - "/data/nginx/certs:/etc/nginx/certs:ro"
    networks:
      - nginx-front
```

```
nginx-gen:
  image: jwilder/docker-gen
  container_name: nginx-gen
  mem_limit: 512m
  restart: always
  volumes:
    - "/var/run/docker.sock:/tmp/docker.sock:ro"
    - "/data/nginx/conf.d:/etc/nginx/conf.d"
    - "/data/nginx/vhost.d:/etc/nginx/vhost.d"
    - "/data/nginx/html:/usr/share/nginx/html"
    - "/data/nginx/certs:/etc/nginx/certs:ro"
  volumes_from:
    - nginx
  entrypoint: /usr/local/bin/docker-gen -notify-sighup nginx \
    -watch -wait 5s:30s /etc/docker-gen/templates/nginx.tmpl \
    /etc/nginx/conf.d/default.conf
```

Keycloak in Docker

Basic setup without k8s/openshift/others

Persistence

postgres

```
version: '3'
```

```
volumes:
```

```
  postgres_data:  
    driver: local
```

```
services:
```

```
  postgres:
```

```
    image: library/postgres:14.5  
    restart: unless-stopped
```

```
    volumes:
```

- /data/jsug/postgres/data:/var/lib/postgresql/data
- ./postgres-init:/docker-entrypoint-initdb.d/

```
  env_file:
```

- ./postgres.env

```
  environment:
```

- POSTGRES_HOST_AUTH_METHOD="scram-sha-256"
- POSTGRES_INITDB_ARGS="--auth-host=scram-sha-256"

```
  networks:
```

- db-backend

Keycloak in Docker

Basic setup without k8s/openshift/others

Persistence

postgres

```
-- postgres-init/setupdb.sql
```

```
SET default_transaction_read_only = off;
```

```
SET client_encoding = 'UTF8';
```

```
SET standard_conforming_strings = on;
```

```
CREATE ROLE jsug_keycloak;
```

```
ALTER ROLE jsug_keycloak WITH NOSUPERUSER INHERIT NOCREATEROLE  
NOCREATEDB LOGIN NOREPLICATION NOBYPASSRLS PASSWORD 'xxxxxxx';
```

```
CREATE DATABASE jsug_keycloak OWNER jsug_keycloak;
```

```
version: '3'
```

```
volumes:
```

```
  postgres_data:
```

```
    driver: local
```

```
services:
```

```
  postgres:
```

```
    image: library/postgres:14.5
```

```
    restart: unless-stopped
```

```
    volumes:
```

```
      - /data/jsug/postgres/data:/var/lib/postgresql/data
```

```
      - ./postgres-init:/docker-entrypoint-initdb.d/
```

```
    env_file:
```

```
      - ./postgres.env
```

```
    environment:
```

```
      - POSTGRES_HOST_AUTH_METHOD="scram-sha-256"
```

```
      - POSTGRES_INITDB_ARGS="--auth-host=scram-sha-256"
```

```
    networks:
```

```
      - db-backend
```

Keycloak in Docker

Basic setup without k8s/openshift/others

Keycloak

quay.io/keycloak/keycloak:19.0.1

```
version: '3'
```

```
volumes:
```

```
  postgres_data:  
    driver: local
```

```
services:
```

```
  keycloak:
```

```
    image: quay.io/keycloak/keycloak:19.0.1
```

```
    restart: unless-stopped
```

```
    env_file:
```

- ./keycloak.env
- ./keycloak-postgres.env

```
    environment:
```

```
      VIRTUAL_HOST: 'auth.java.wien'
```

```
      VIRTUAL_NETWORK: 'nginx-front-119'
```

```
      VIRTUAL_PORT: 8080
```

```
      LETSENCRYPT_HOST: 'auth.java.wien'
```

```
      LETSENCRYPT_EMAIL: "dominik@dominikdorn.com"
```

```
    entrypoint: /opt/keycloak/bin/kc.sh start
```

```
    expose:
```

- "8080"

```
    depends_on:
```

- postgres

```
    networks:
```

- nginx-front
- db-backend

Keycloak in Docker

Basic setup without k8s/openshift/others

Keycloak

quay.io/keycloak/keycloak:19.0.1

```
# keycloak.env
KC_HOSTNAME=auth.java.wien
KC_HOSTNAME_STRICT=false
KC_HTTP_ENABLED="true"
KC_HTTP_PORT=8080
KC_METRICS_ENABLED=true
KC_PROXY=edge
KEYCLOAK_ADMIN=xxxxadmin
KEYCLOAK_ADMIN_PASSWORD=xxxxx
PROXY_ADDRESS_FORWARDING=true
```

```
# keycloak-postgres.env
KC_DB=postgres
KC_DB_USERNAME=jsug_keycloak
KC_DB_PASSWORD=xxxxxxx
KC_DB_SCHEMA=public
KC_DB_URL_DATABASE=jsug_keycloak
KC_DB_URL_HOST=postgres
KC_DB_URL_PORT=5432
```

Meetup API

Meetup API

meetup

Start a new group English Log in Sign up

Introduction to GraphQL ^

- Benefits of GraphQL
- Business Solutions
- API Terms of Service

Guide to GraphQL API v

Authentication & Security v

GraphQL schema v

Migrating from REST

GraphQL Playground

OAuth Clients

Benefits of GraphQL

GraphQL is a modern way to request data from APIs. An alternative to REST implementations, it was developed to minimize the amount of data transferred and increase software engineer productivity. It was originally an internal tool within Facebook, but has been open source since 2015. GraphQL has since gained an amazing amount of traction, and is in active use by companies spanning all sizes and industries.

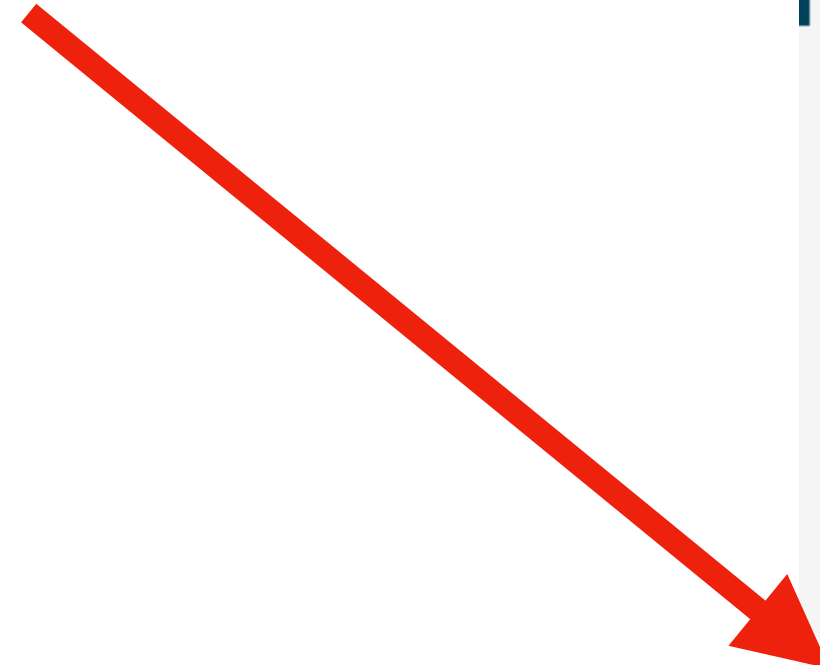
A REST API returns all fields and data it was set up to deliver, even if you are not using them. GraphQL allows you to craft your own queries and returns the exact fields you specify. By specifying the exact fields you need, network traffic and unnecessary processing is reduced. More exact queries also increase clarity for your engineering staff. Unlike REST APIs where there are many endpoints or routes, there is only one endpoint for GraphQL. You can also combine multiple GraphQL queries in one request.

Business Solutions

Meetup delivers the resources you need to build a thriving community. The GraphQL API adds additional functionality by enabling customers to tailor community management to specific needs.

Here are some examples of how customers and partners are using the

Meetup API



← → ↻ https://www.meetup.com/api/general/ Google 120%

meetup [Start a new group](#) English Log in Sign up

Introduction to GraphQL ^

- Benefits of GraphQL
- Business Solutions
- API Terms of Service

Guide to GraphQL API v

Authentication & Security v

GraphQL schema v

Migrating from REST

GraphQL Playground

OAuth Clients

Benefits of GraphQL

GraphQL is a modern way to request data from APIs. An alternative to REST implementations, it was developed to minimize the amount of data transferred and increase software engineer productivity. It was originally an internal tool within Facebook, but has been open source since 2015. GraphQL has since gained an amazing amount of traction, and is in active use by companies spanning all sizes and industries.

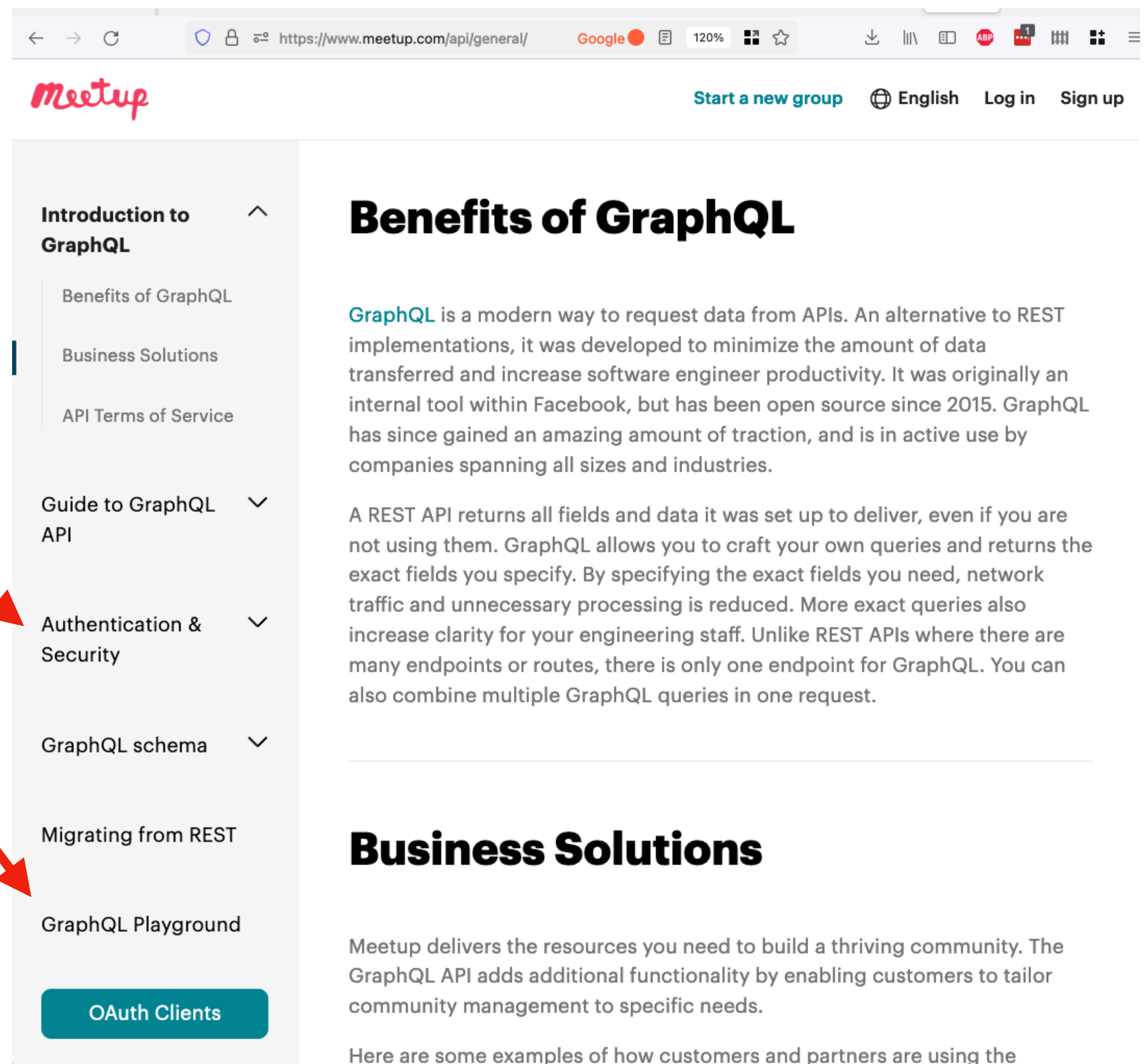
A REST API returns all fields and data it was set up to deliver, even if you are not using them. GraphQL allows you to craft your own queries and returns the exact fields you specify. By specifying the exact fields you need, network traffic and unnecessary processing is reduced. More exact queries also increase clarity for your engineering staff. Unlike REST APIs where there are many endpoints or routes, there is only one endpoint for GraphQL. You can also combine multiple GraphQL queries in one request.

Business Solutions

Meetup delivers the resources you need to build a thriving community. The GraphQL API adds additional functionality by enabling customers to tailor community management to specific needs.

Here are some examples of how customers and partners are using the

Meetup API



The screenshot shows the Meetup GraphQL API documentation page. The browser address bar displays `https://www.meetup.com/api/general/`. The Meetup logo is in the top left, and navigation links for "Start a new group", "English", "Log in", and "Sign up" are in the top right. A sidebar on the left contains a list of links: "Introduction to GraphQL", "Benefits of GraphQL", "Business Solutions", "API Terms of Service", "Guide to GraphQL API", "Authentication & Security", "GraphQL schema", "Migrating from REST", and "GraphQL Playground". At the bottom of the sidebar is a teal button labeled "OAuth Clients". Two red arrows originate from the left side of the image; one points to the "Authentication & Security" link in the sidebar, and the other points to the "GraphQL Playground" link. The main content area features a section titled "Benefits of GraphQL" with a paragraph explaining GraphQL as a modern data request method, followed by another paragraph comparing REST and GraphQL. Below this is a section titled "Business Solutions" with a paragraph about building a thriving community and a final line of text starting with "Here are some examples of how customers and partners are using the".

meetup

Start a new group English Log in Sign up

Introduction to GraphQL

- Benefits of GraphQL
- Business Solutions
- API Terms of Service

Guide to GraphQL API

- Authentication & Security
- GraphQL schema

Migrating from REST

GraphQL Playground

OAuth Clients

Benefits of GraphQL

GraphQL is a modern way to request data from APIs. An alternative to REST implementations, it was developed to minimize the amount of data transferred and increase software engineer productivity. It was originally an internal tool within Facebook, but has been open source since 2015. GraphQL has since gained an amazing amount of traction, and is in active use by companies spanning all sizes and industries.

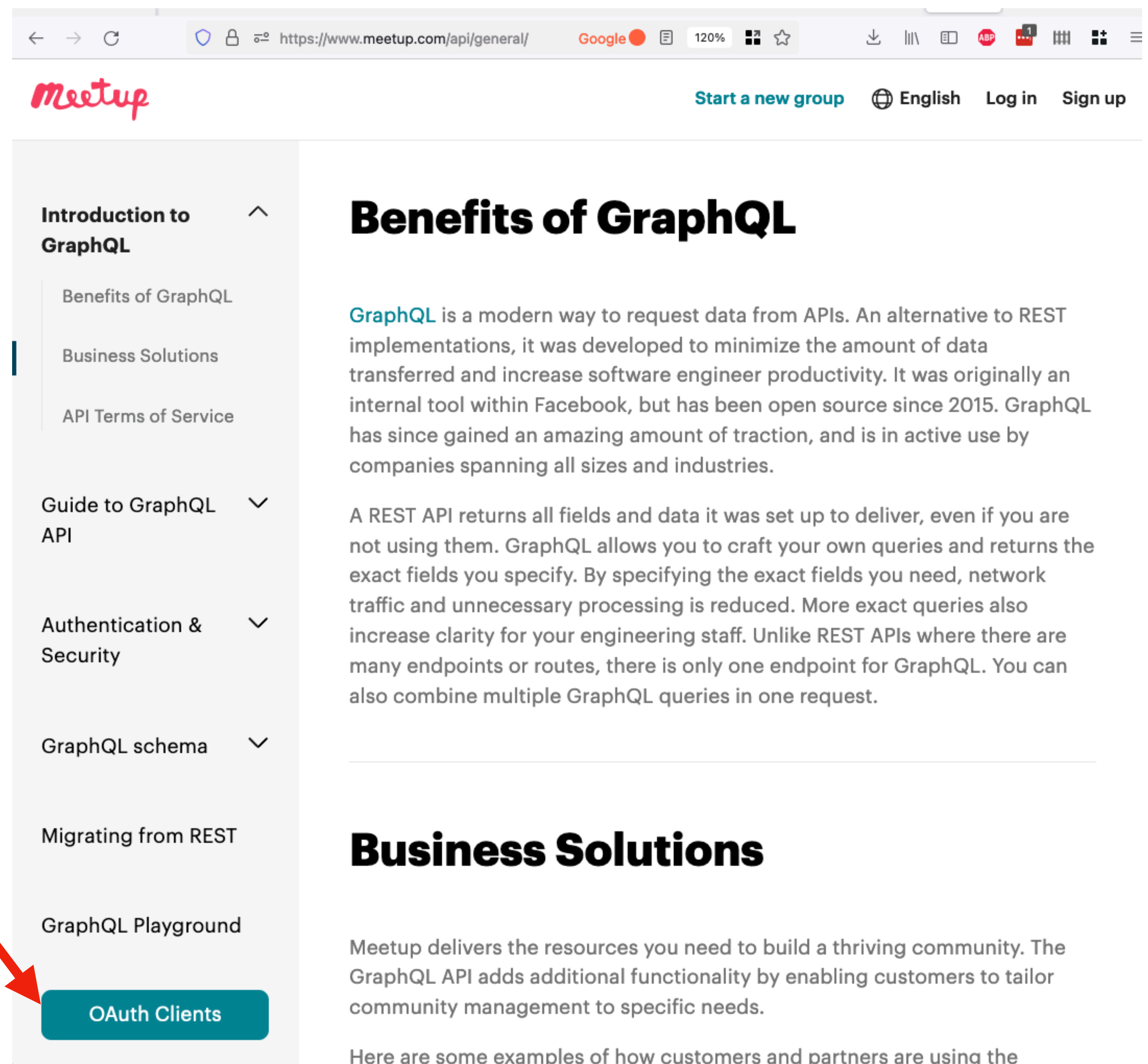
A REST API returns all fields and data it was set up to deliver, even if you are not using them. GraphQL allows you to craft your own queries and returns the exact fields you specify. By specifying the exact fields you need, network traffic and unnecessary processing is reduced. More exact queries also increase clarity for your engineering staff. Unlike REST APIs where there are many endpoints or routes, there is only one endpoint for GraphQL. You can also combine multiple GraphQL queries in one request.

Business Solutions

Meetup delivers the resources you need to build a thriving community. The GraphQL API adds additional functionality by enabling customers to tailor community management to specific needs.

Here are some examples of how customers and partners are using the

Meetup API



The screenshot shows the Meetup GraphQL API documentation page. The browser address bar displays `https://www.meetup.com/api/general/`. The Meetup logo is in the top left, and navigation links for 'Start a new group', 'English', 'Log in', and 'Sign up' are in the top right. The left sidebar contains a list of navigation items: 'Introduction to GraphQL' (with a sub-menu), 'Guide to GraphQL API', 'Authentication & Security', 'GraphQL schema', 'Migrating from REST', 'GraphQL Playground', and 'OAuth Clients' (highlighted with a teal button). A red arrow points from the 'Meetup API' title to the 'OAuth Clients' button. The main content area features the heading 'Benefits of GraphQL' followed by a paragraph explaining GraphQL as a modern data request method. Below this is the heading 'Business Solutions' followed by a paragraph about building a thriving community with the GraphQL API.

meetup

Start a new group English Log in Sign up

Introduction to GraphQL

- Benefits of GraphQL
- Business Solutions
- API Terms of Service

Guide to GraphQL API

Authentication & Security

GraphQL schema

Migrating from REST

GraphQL Playground

OAuth Clients

Benefits of GraphQL

GraphQL is a modern way to request data from APIs. An alternative to REST implementations, it was developed to minimize the amount of data transferred and increase software engineer productivity. It was originally an internal tool within Facebook, but has been open source since 2015. GraphQL has since gained an amazing amount of traction, and is in active use by companies spanning all sizes and industries.

A REST API returns all fields and data it was set up to deliver, even if you are not using them. GraphQL allows you to craft your own queries and returns the exact fields you specify. By specifying the exact fields you need, network traffic and unnecessary processing is reduced. More exact queries also increase clarity for your engineering staff. Unlike REST APIs where there are many endpoints or routes, there is only one endpoint for GraphQL. You can also combine multiple GraphQL queries in one request.

Business Solutions

Meetup delivers the resources you need to build a thriving community. The GraphQL API adds additional functionality by enabling customers to tailor community management to specific needs.

Here are some examples of how customers and partners are using the

Meetup API

The screenshot shows the Meetup API documentation page. The browser address bar displays `https://www.meetup.com/api/general/`. The Meetup logo is in the top left, and navigation links for "Start a new group", "English", "Log in", and "Sign up" are in the top right. A left sidebar contains a menu with items: "Introduction to GraphQL" (expanded), "Benefits of GraphQL", "Business Solutions", "API Terms of Service", "Guide to GraphQL API", "Authentication & Security", "GraphQL schema", "Migrating from REST", "GraphQL Playground", and "OAuth Clients". The main content area starts with the text "Here are some examples of how customers and partners are using the Meetup API:" followed by a bulleted list of use cases. Below this is a section titled "Get API access and more with Meetup Pro", which is highlighted by a red arrow. The text "In addition to all of the feature you already have with a standard plan, pro gives you" is followed by a list of features, each preceded by a checkmark. The feature "Platform to manage a network of groups" is highlighted by another red arrow. A red "Sign up" button is located at the bottom of the main content area.

meetup

[Start a new group](#) [English](#) [Log in](#) [Sign up](#)

Introduction to GraphQL

- Benefits of GraphQL
- Business Solutions
- API Terms of Service

Guide to GraphQL API

Authentication & Security

GraphQL schema

Migrating from REST

GraphQL Playground

[OAuth Clients](#)

Here are some examples of how customers and partners are using the Meetup API:

- Deliver real-time performance statistics through a custom dashboard (eg highlighting most active groups, or events with low RSVP rates)
- Drive RSVPs and group joins by showcasing upcoming events and local groups on public (non-Meetup) pages
- Schedule events and manage groups using existing CRM tools

Get API access and more with Meetup Pro

In addition to all of the feature you already have with a standard plan, pro gives you

- ✓ API access
- ✓ Platform to manage a network of groups
- ✓ Access to attendee emails
- ✓ Advanced network analytics
- ✓ Custom attendance forms
- ✓ SEO-friendly branded network page

[Sign up](#)

Meetup API

← → ↻ https://www.meetup.com/api/general/ Google 120% ☆

meetup [Start a new group](#) [English](#) [Log in](#) [Sign up](#)

Introduction to GraphQL ^

[Benefits of GraphQL](#)

[Business Solutions](#)

[API Terms of Service](#)

Guide to GraphQL API v

Authentication & Security v

GraphQL schema v

Migrating from REST

GraphQL Playground

[OAuth Clients](#)

Here are some examples of how customers and partners are using the Meetup API:

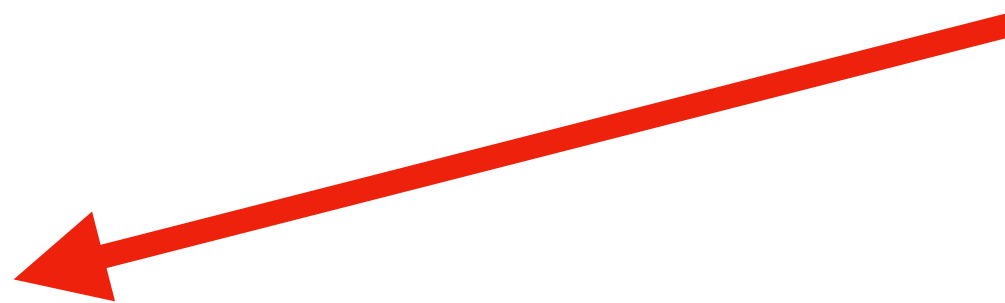
- Deliver real-time performance statistics through a custom dashboard (eg highlighting most active groups, or events with low RSVP rates)
- Drive RSVPs and group joins by showcasing upcoming events and local groups on public (non-Meetup) pages
- Schedule events and manage groups using existing CRM tools

Get API access and more with Meetup Pro

In addition to all of the feature you already have with a standard plan, pro gives you

- ✓ API access
- ✓ Platform to manage a network of groups
- ✓ Access to attendee emails
- ✓ Advanced network analytics
- ✓ Custom attendance forms
- ✓ SEO-friendly branded network page

[Sign up](#)



Meetup API

Profil bearbeiten

Personal Info

Account
Management

E-Mail Updates

Push
Nachrichten

Privatsphäre

Social Media

Interessen

**Organizer
Subscription**

Payment
Methods

Geleistete
Zahlungen

Erhaltene
Zahlungen

Apps

Organizer Subscription

You are currently on a 6-month **Pro** subscription plan. Your subscription will renew automatically every 6 months until canceled.

Your subscription rate is 30,00 \$ per group per month, billed every 6 months, with a minimum charge of 180,00 \$ per billing (the equivalent of 1 group). Tax may apply. You currently have 1 group.

If you increase the number of groups during a billing period, the following month we will charge a one-time prorated amount covering the remainder of the current billing period.

As an example:

Suppose you started your 6 month subscription with 5 groups. You add 2 new groups during your 4th month into your billing period.

*At the end of the 4th month, there are 2 remaining months in the billing period. **The per-month cost for each group is 30,00 \$**, so we will make a one-time charge of 120,00 \$ (30,00 \$ per group per month x 2 groups x 2 months) to your credit card on file. Tax may apply.*

Your next payment is scheduled for Oktober 6, 2022.

If you have any questions about your payment status, billing history, or changes to your subscription, send us an email at pro-billing@meetup.com.

[Contact billing support](#) | [Update credit card](#) | [Payment history](#)

Meetup API

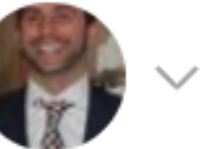


[Start a new group](#)

 Pro Dashboard

 Messages

 Notifications



Introduction to GraphQL

Guide to GraphQL API

Authentication & Security

GraphQL schema

Migrating from REST

GraphQL Playground

OAuth Clients

Your OAuth Clients

Java Vienna OpenID Connect

 [Settings](#)

Your Key 7jbdac6ugdb5in0pb44k1u2ojo

Secret  

Website http://java.wien/

Redirect URL https://auth.java.wien/realms/Java-Vienna/broker

Java-Vienna OpenID Connect

Rejected

 [Settings](#)

Create new client

What is OAuth?

OAuth is an authorization protocol that allows API applications to securely act on a user's behalf.

For more information on OAuth, please visit the [OAuth Community Site](#).

Why OAuth?

OAuth removes the need for members to hand over their credentials to you directly.

Please see our [developer documentation](#) for more information.

Meetup API



Introduction to GraphQL

Guide to GraphQL API

Authentication & Security

GraphQL schema

Migrating from REST

GraphQL Playground

OAuth Clients

Using OAuth 2

The Meetup API supports authenticating requests using [OAuth 2](#) over HTTPS. We provide implementations a number of protocol flows outlined below. We provide the following endpoints for acquiring member authorization and access tokens.

Endpoint	URL
Authorization	https://secure.meetup.com/oauth2/authorize
Access Tokens	https://secure.meetup.com/oauth2/access

Redirect URIs

Before you can use OAuth 2 for member authorization, you need to either [register](#) a new OAuth client or add a **redirect_uri** to an existing client by clicking the edit link next to you client's [listing](#). The **redirect_uri** you register for a given client will be used to validate future oauth2 requests.

This uri is used as a basis for validating the address authorization requests will redirect a member to after granting or denying access to your app.

If provided, the redirect URL's host and port must exactly match the callback URL. The redirect URL's path must reference a subdirectory of the callback URL.

CALLBACK: `http://example.com/path`

GOOD: `http://example.com/path`

GOOD: `http://example.com/path/subdir/other`

BAD: `http://example.com/bar`


BAD: `http://example.com/`

BAD: `http://example.com:8080/path`

Meetup API


GraphQL

/userinfo



Start a new group

Pro DashboardMessagesNotifications



Introduction to GraphQL

Guide to GraphQL API

Authentication & Security

GraphQL schema

Migrating from REST

GraphQL Playground

OAuth Clients

GraphQL Playground

Meetup Graphql API

1 query { self {

2 id

3 email

4 name

5 memberUrl

6 memberPhotoUrl

7 } }

QUERY VARIABLES

1 {"eventId": "276754274"}

PrettifyMergeCopyHistory

< Docs

{

"data": {

"self": {

"id": "36867052",

"email": "dominik@dominikdorn.com",

"name": "Dominik Dorn",

"memberUrl": "https://www.meetup.com

/members/36867052",

"memberPhotoUrl":

"https://secure.meetupstatic.com/photos/member

/6/1/d/7/highres_244645047.jpeg"

}

}

}

**Meetup Keycloak IdP
-> Demo ;-)**

Meetup Keycloak IdP

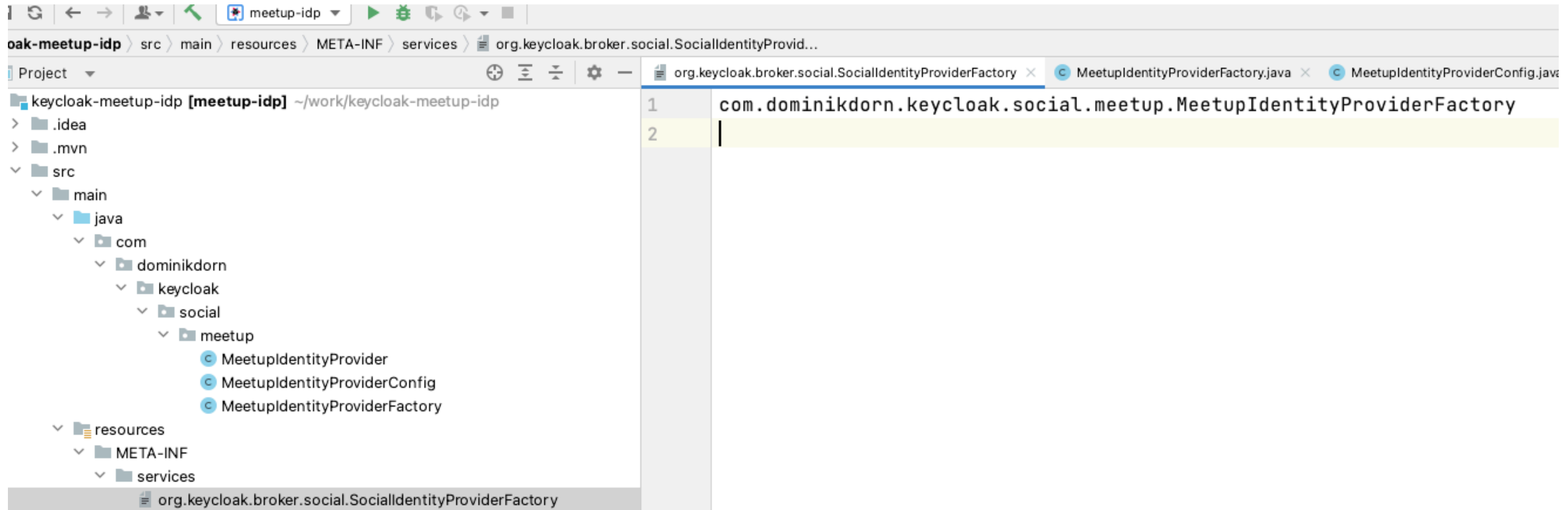
```
keycloak-meetup-idp [meetup-idp] ~/work/keycloak-meetup-idp
├── .idea
├── .mvn
├── src
│   ├── main
│   │   ├── java
│   │   │   ├── com
│   │   │   │   ├── dominikdorn
│   │   │   │   │   ├── keycloak
│   │   │   │   │   │   ├── social
│   │   │   │   │   │   │   ├── meetup
│   │   │   │   │   │   │   │   ├── MeetupIdentityProvider
│   │   │   │   │   │   │   │   ├── MeetupIdentityProviderConfig
│   │   │   │   │   │   │   │   └── MeetupIdentityProviderFactory
│   │   │   │   │   │   │   └── resources
│   │   │   │   │   │   │   │   ├── META-INF
│   │   │   │   │   │   │   │   │   ├── services
│   │   │   │   │   │   │   │   │   └── org.keycloak.broker.social.SocialIdentityProviderFactory
│   │   │   │   │   │   └── target
│   │   │   │   │   │   ├── demo.txt
│   │   │   │   │   │   ├── mvnw
│   │   │   │   │   │   ├── mvnw.cmd
│   │   │   │   │   │   └── pom.xml
```

Meetup Keycloak IdP

```
etupIdentityProvider.java x pom.xml (meetup-idp) x SocialIdentityProvider.java x org.keycloak.broker.social.SocialIdentityProviderFa

<groupId>com.dominikdorn.keycloak</groupId>
<artifactId>meetup-idp</artifactId>
<version>1.0.0-SNAPSHOT</version>
<packaging>jar</packaging>
<properties>
  <compiler-plugin.version>3.8.1</compiler-plugin.version>
  <maven.compiler.release>11</maven.compiler.release>
  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
  <surefire-plugin.version>3.0.0-M7</surefire-plugin.version>
  <apache.httpcomponents.version>4.5.13</apache.httpcomponents.version>
</properties>
<dependencies>
  <dependency>
    <groupId>org.keycloak</groupId>
    <artifactId>keycloak-services</artifactId>
    <version>19.0.1</version>
    <scope>provided</scope>
  </dependency>
  <dependency>
    <groupId>org.keycloak</groupId>
    <artifactId>keycloak-server-spi-private</artifactId>
    <version>19.0.1</version>
    <scope>provided</scope>
  </dependency>
  <dependency>
    <groupId>org.keycloak</groupId>
    <artifactId>keycloak-server-spi</artifactId>
    <version>19.0.1</version>
    <scope>provided</scope>
  </dependency>
  <dependency>
    <groupId>org.apache.httpcomponents</groupId>
    <artifactId>httpclient</artifactId>
    <version>${apache.httpcomponents.version}</version>
    <scope>provided</scope>
  </dependency>
</dependencies>
```

Meetup Keycloak IdP



Identity providers

Identity providers are social networks or identity brokers that allow users to authenticate to Keycloak. [Learn more](#)

→ **Add provider** Manage display order

Name		Provider details
meetup-oauth2	User-defined	Meetup-oauth2
	Keycloak OpenID Connect	
	OpenID Connect v1.0	
	SAML v2.0	
	Social	
	BitBucket	
	Facebook	
	GitHub	
	GitLab	
	Google	
	Instagram	
	LinkedIn	
	Meetup.com (OAuth2)	
	Microsoft	
	Openshift v3	
	Openshift v4	
	PayPal	

```
meetupIdentityProviderFactory
org.keycloak.broker.social.SocialIdentityProviderFactory x MeetupIdentityProviderFactory.java x MeetupIdentityProviderConfig.java x Abstrac
1 package com.dominikdorn.keycloak.social.meetup;
2
3 import ...
4
5 /**
6  * Meetup.com (OAuth2) Identity Provider factory class.
7  *
8  * @author Dominik Dorn
9  */
10
11 1 usage
12 public class MeetupIdentityProviderFactory extends
13     AbstractIdentityProviderFactory<MeetupIdentityProvider> implements
14     SocialIdentityProviderFactory<MeetupIdentityProvider> {
15
16 1 usage
17     public static final String PROVIDER_ID = "meetup-oauth2";
18     1 usage
19     public static final String NAME = "Meetup.com (OAuth2)";
20
21     @Override
22     public String getName() { return NAME; }
23
24
25     @Override
26     public String getId() { return PROVIDER_ID; }
27
28
29     @Override
30     public MeetupIdentityProvider create(KeycloakSession session, IdentityProvider
31         | return new MeetupIdentityProvider(session, new MeetupIdentityProviderConfig(
32     )
33
34 1 usage
35     @Override
36     public OAuth2IdentityProviderConfig createConfig() { return new MeetupIdentity
37
38
39
40
41
42 }
```


Meetup Keycloak IdP

```
IdentityProviderConfig > MeetupIdentityProviderConfig
g.keycloak.broker.social.SocialIdentityProviderFactory x MeetupIdentityProviderFactory.java x MeetupIdentityProviderConfig.java x AbstractOA

package com.dominikdorn.keycloak.social.meetup;

import ...

5 usages
public class MeetupIdentityProviderConfig extends OAuth2IdentityProviderConfig {

    2 usages
    public MeetupIdentityProviderConfig(IdentityProviderModel model) {
        super(model);
    }

    2 usages
    public MeetupIdentityProviderConfig() {
    }
}
```

Meetup Keycloak IdP

Using OAuth 2

The Meetup API supports authenticating requests using [OAuth 2](#) over HTTPS. We provide implementations a number of protocol flows outlined below. We provide the following endpoints for acquiring member authorization and access tokens.

Endpoint	URL
Authorization	https://secure.meetup.com/oauth2/authorize
Access Tokens	https://secure.meetup.com/oauth2/access

Redirect URIs

Before you can use OAuth 2 for member authorization, you need to either [register](#) a new OAuth client or add a **redirect_uri** to an existing client by clicking the edit link next to you client's [listing](#). The **redirect_uri** you register for a given client will be used to validate future oauth2 requests.

```
package com.dominikdorn.keycloak.social.meetup;

import ...

5 usages
public class MeetupIdentityProvider extends
    AbstractOAuth2IdentityProvider<MeetupIdentityProviderConfig> implements
    SocialIdentityProvider<MeetupIdentityProviderConfig>, ExchangeTokenToIdentityProviderToken {

2 usages
public static final String BASE_URL = "https://secure.meetup.com";

1 usage
public static final String AUTHORIZATION_URL = BASE_URL + "/oauth2/authorize";
1 usage
public static final String ACCESS_TOKEN_URL = BASE_URL + "/oauth2/access";

protected static final Logger logger = Logger.getLogger(MeetupIdentityProvider.class);

2 usages
public MeetupIdentityProvider(KeycloakSession session, MeetupIdentityProviderConfig config) {
    super(session, config);
    config.setAuthorizationUrl(AUTHORIZATION_URL);
    config.setTokenUrl(ACCESS_TOKEN_URL);
    config.setTrustEmail(true);
}

@Override
protected String getDefaultScopes() { return "profile"; }
```


Meetup Keycloak IdP

Server Flow

This flow is suitable for applications that are capable of securely storing consumer secrets and target interactive member authentication.

Requesting Authorization

Redirect your user to the following URL in a browser:

`https://secure.meetup.com/oauth2/authorize?client_id={YOUR_CLIENT_KEY}&response_type=code&redirect_uri={YOUR_CLIENT_REDIRECT_URI}`

No scope is allowed in the URL!

```
@Override
protected UriBuilder createAuthorizationUrl(AuthenticationRequest request) {
    final UriBuilder uriBuilder = UriBuilder.fromUri(getConfig().getAuthorizationUrl())
        .queryParams(OAUTH2_PARAMETER_SCOPE, getConfig().getDefaultScope())
        .queryParams(OAUTH2_PARAMETER_STATE, request.getState().getEncoded())
        .queryParams(OAUTH2_PARAMETER_RESPONSE_TYPE, ...objects: "code")
        .queryParams(OAUTH2_PARAMETER_CLIENT_ID, getConfig().getClientId())
        .queryParams(OAUTH2_PARAMETER_REDIRECT_URI, request.getRedirectUri());

    return uriBuilder;
}
```

Meetup Keycloak IdP

```
Meetup GraphQL API
1 query { self {
2   id
3   email
4   name
5   memberUrl
6   memberPhotoUrl
7 } }
```

```
2 usages
class GraphQLQuery {
  2 usages
  private String query;

  1 usage
  public GraphQLQuery(String query) { this.query = query; }

  public String getQuery() { return query; }
}

@Override
protected BrokeredIdentityContext doGetFederatedIdentity(String accessToken) {
  try {
    GraphQLQuery query = new GraphQLQuery(
      "query { self { id email name memberUrl memberPhotoUrl } }");
    JsonNode profile = SimpleHttp.doPost( url: "https://api.meetup.com/gql",
      session
    ).json(query).header( name: "Authorization", value: "Bearer " + accessToken).asJson();

    BrokeredIdentityContext context = extractIdentityFromProfile( event: null, profile);

    if(getConfig().isStoreToken()) {
      context.setToken(accessToken);
    }

    return context;
  } catch (Exception e) {
    throw new IdentityBrokerException("Could not obtain user profile from meetup.", e);
  }
}
```

Meetup Keycloak IdP

```
Meetup GraphQL API ▶ Prettify Merge Copy History < Docs
```

```
1 query { self {  
2   id  
3   email  
4   name  
5   memberUrl  
6   memberPhotoUrl  
7 } }
```

```
{  
  "data": {  
    "self": {  
      "id": "36867052",  
      "email": "dominik@dominikdorn.com",  
      "name": "Dominik Dorn",  
      "memberUrl": "https://www.meetup.com/members/36867052",  
      "memberPhotoUrl":  
        "https://secure.meetupstatic.com/photos/member/6/1/d/7/highres_244645047.jpeg"  
    }  
  }  
}
```

```
1 usage  
@Override  
protected BrokeredIdentityContext extractIdentityFromProfile(EventBuilder event, JsonNode profile) {  
  try {  
    JsonNode data = profile.get("data");  
    JsonNode self = data.get("self");  
  
    String userId = self.get("id").asText();  
    String email = self.get("email").asText();  
    String name = self.get("name").asText();  
    String memberUrl = self.get("memberUrl").asText();  
    String memberPhotoUrl = self.get("memberPhotoUrl").asText();  
  
    BrokeredIdentityContext user = new BrokeredIdentityContext(userId);  
    user.setId(userId);  
    user.setUsername(email);  
    user.setIdpConfig(getConfig());  
    user.setIdp(this);  
    user.setName(name);  
    user.setEmail(email);  
    user.setModelUsername(userId);  
    user.setBrokerUserId(userId);  
    HashMap<String, Object> contextData = new HashMap<>();  
    contextData.put("memberUrl", memberUrl);  
    contextData.put("memberPhotoUrl", memberPhotoUrl);  
    user.setContextData(contextData);  
    user.setUserAttribute(attributeName: "memberUrl", memberUrl);  
    user.setUserAttribute(attributeName: "memberPhotoUrl", memberPhotoUrl);  
  
    AbstractJsonUserAttributeMapper.storeUserProfileForMapper(user, profile,  
      getConfig().getAlias());  
    return user;  
  } catch (Exception e) {  
    throw new IdentityBrokerException("Could not obtain user profile from meetup.", e);  
  }  
}
```

Deploying to Docker


```
domdorn@MacBook-Pro-5:~/work/keycloak-meetup-idp|
⇒ ./mvnw package
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.dominikdorn.keycloak:meetup-idp >-----
[INFO] Building meetup-idp 1.0.0-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ meetup-idp ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 1 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ meetup-idp ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 3 source files to /Users/domdorn/work/keycloak-meetup-idp/target/classes
[INFO] /Users/domdorn/work/keycloak-meetup-idp/src/main/java/com/dominikdorn/keycloak/social/meetup/MeetupIdentityProvider.java: /Users/domdorn/work/keycloak-meetup-idp/src/main/java/com/dominikdorn/keycloak/social/meetup/MeetupIdentityProvider.java uses or overrides a deprecated API.
[INFO] /Users/domdorn/work/keycloak-meetup-idp/src/main/java/com/dominikdorn/keycloak/social/meetup/MeetupIdentityProvider.java: Recompile with -Xlint:deprecation for details.
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ meetup-idp ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /Users/domdorn/work/keycloak-meetup-idp/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:testCompile (default-testCompile) @ meetup-idp ---
[INFO] No sources to compile
[INFO]
[INFO] --- maven-surefire-plugin:3.0.0-M7:test (default-test) @ meetup-idp ---
[INFO] No tests to run.
[INFO]
[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ meetup-idp ---
[INFO] Building jar: /Users/domdorn/work/keycloak-meetup-idp/target/meetup-idp-1.0.0-SNAPSHOT.jar
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 2.226 s
[INFO] Finished at: 2022-10-02T18:06:22+02:00
[INFO]
domdorn@MacBook-Pro-5:~/work/keycloak-meetup-idp|
⇒ scp /Users/domdorn/work/keycloak-meetup-idp/target/meetup-idp-1.0.0-SNAPSHOT.jar domdorn@l1:~/meetup-idp-1.0.0-SNAPSHOT.jar
100% 8685 167.8KB/s 00:00
domdorn@MacBook-Pro-5:~/work/keycloak-meetup-idp|
⇒
```

Meetup Keycloak IdP

```
ssh l1
domdorn@Ubuntu-1404-trusty-64-minimal ~
% docker cp ~domdorn/meetup-idp-1.0.0-SNAPSHOT.jar jsug_keycloak_1:/opt/keycloak/providers
domdorn@Ubuntu-1404-trusty-64-minimal ~
% docker exec jsug_keycloak_1 /opt/keycloak/bin/kc.sh build
Updating the configuration and installing your custom providers, if any. Please wait.
2022-10-02 17:43:15,154 WARN [org.keycloak.services] (build-15) KC-SERVICES0047: meetup-oauth2 (com.dominikdorn.keycloak.social.meetup.MeetupIdentity
ProviderFactory) is implementing the internal SPI social. This SPI is internal and may change without notice
2022-10-02 17:43:17,812 INFO [io.quarkus.deployment.QuarkusAugmentor] (main) Quarkus augmentation completed in 7112ms
Server configuration updated and persisted. Run the following command to review the configuration:

    kc.sh show-config










domdorn@Ubuntu-1404-trusty-64-minimal ~
% docker exec -it jsug_keycloak_1 /bin/bash
bash-4.4$ kill 1
bash-4.4$ %
domdorn@Ubuntu-1404-trusty-64-minimal ~
% █
```

Keycloak Configuration

Keycloak Config

[Identity providers](#) > Add provider

Add Meetup-oauth2 provider

Redirect URI 	<input type="text" value="https://auth.java.wien/realms/Java-Vienna/broker/meetup-oauth2/endpoint"/>	
Client ID * 	<input type="text" value="my-client-id-given-by-meetup"/>	
Client Secret * 	<input type="password" value="....."/>	 
Display order 	<input type="text"/>	

Keycloak Config



≡

KEYCLOAK

Java-Vienna

▼

Manage

Clients

Client scopes

Realm roles

Users

Groups

Sessions

Events

Configure

Realm settings

Authentication

Identity providers

User federation

Identity providers

Provider details

Meetup-oauth2

Settings

Mappers

Permissions

General settings

Redirect URI ?

https://auth.java.wien/realms/Java-Vienna/broker/meetup-oauth2/endpoint

Client ID * ?

my-client-id-given-by-meetup

Client Secret * ?

.....

Display order ?

Advanced settings

Scopes ?

Store tokens ?

Off

Accepts prompt=none forward from client ?

Off

Disable user info ?

Off

Trust Email ?

On

Account linking only ?

Off

Hide on login page ?

Off

First login flow ?

first broker login

Post login flow ?

None

Sync mode ?

Import

Save

Revert

Keycloak Config

Java-Vienna

Manage

Clients

Client scopes

Realm roles

Users

Groups

Sessions

Events

Configure

Realm settings

Authentication

Identity providers

Clients

Clients are applications and services that can request authentication of a user. [Learn more](#)

Clients listInitial access token

Search for client

Create client

Import client

Client ID	Type	Description	Home URL
account	OpenID Connect	—	https://auth.java.wien/realms/Java-Vienna/account/
account-console	OpenID Connect	—	https://auth.java.wien/realms/Java-Vienna/account/
admin-cli	OpenID Connect	—	—
broker	OpenID Connect	—	—
java-vienna-wordpress	OpenID Connect	—	https://www.java.wien/wp-admin/
realm-management	OpenID Connect	—	—
security-admin-console	OpenID Connect	—	https://auth.java.wien/admin/Java-Vienna/console/

1-7

Keycloak Config

java-vienna-wordpress

OpenID Connect

Clients are applications and services that can request authentication of a user.

Settings

Keys

Credentials

Roles

Client scopes

Authorization

Service accounts roles

Sessions

Advanced

General Settings

Client ID * ?

java-vienna-wordpress

Name ?

Java Vienna Wordpress

Description ?

Always display in console ?

Off

Access settings

Root URL ?

https://www.java.wien

Home URL ?

https://www.java.wien/wp-admin/

Valid redirect URIs ?

https://www.java.wien/wp-admin/admin-ajax.php?action=openid-connect-authorize

+ Add valid redirect URIs

Valid post logout redirect URIs ?

https://www.java.wien/*

+ Add valid post logout redirect URIs

Web origins ?

+ Add web origins

Admin URL ?

Jump to section

General Settings

Access settings

Capability config

Login settings

Logout settings

Keycloak Config

Capability config

Client authentication ?

On

Authorization ?

On

Authentication flow

☒ Standard flow ?

☒ Direct access grants ?

☐ Implicit flow ?

☒ Service accounts roles ?

☐ OAuth 2.0 Device Authorization Grant ?

☐ OIDC CIBA Grant ?

Login settings

Login theme ?

keycloak

Consent required ?

Off

Display client on screen ?

Off

Client consent screen text ?

Logout settings

Front channel logout ?

On

Front-channel logout URL ?

https://www.java.wien/wp-login.php?action=logout

Backchannel logout URL ?

Backchannel logout session required ?

On

Backchannel logout revoke offline sessions ?

Off

Integration in Wordpress

OpenID Connect Generic Client

Beschreibung Installation FAQ Änderungsprotokoll Bewertungen

This plugin allows to authenticate users against OpenID Connect OAuth2 API with Authorization Code Flow.

Once installed, it can be configured to automatically authenticate users (SSO), or provide a „Login with OpenID Connect“ button on the login form. After consent has been obtained, an existing user is automatically logged into WordPress, while new users are created in WordPress database.

Much of the documentation can be found on the Settings > OpenID Connect Generic dashboard page.

Please submit issues to the Github repo: <https://github.com/daggerhart/openid-connect-generic>

Version: 3.9.1

Autor: [daggerhart](#)

Zuletzt aktualisiert: vor 1 Monat

Erforderliche WordPress-Version: 4.9 oder höher

Kompatibel bis zu: 6.0.2

Erforderliche PHP-Version: 7.2 oder höher

Aktive Installationen: 3.000+

[WordPress.org-Plugin-Seite »](#)

[Plugin-Homepage »](#)

DURCHSCHNITTliche BEWERTUNG

★★★★★
(basierend auf 13 Abstimmungen)

BEWERTUNGEN

Lies alle Bewertungen auf WordPress.org oder schreib deine Eigene!

5 Sterne	13
4 Sterne	0
3 Sterne	0
2 Sterne	0
1 Stern	0

MITWIRKENDE

 [Jonathan Daggerhart](#)

 [Tim Nolte](#)

[Für dieses Plugin spenden »](#)

Groups

[groups_member group="Premium"]
Only members can see this.
[/groups_member]



Groups

Membership and Access Control

Beschreibung Installation FAQ Änderungsprotokoll Screenshots Bewertungen

Groups is designed as an efficient, powerful and flexible solution for group-oriented memberships and content access control.

It provides group-based user membership management, group-based capabilities and access control for content, built on solid principles.

Groups is light-weight and offers an easy user interface, while it acts as a framework and integrates standard WordPress capabilities and application-specific capabilities along with an extensive API.

Enhanced functionality is available via [Official Extensions](#) for Groups.

Dokumentation

The official documentation is located at the [Groups Documentation](#) pages.

Funktionen

User groups

- Supports an unlimited number of groups
- Provides a Registered group which is automatically maintained
- Users can be assigned to any group
- Users are added automatically to the Registered group

Groups hierarchy

- Supports group hierarchies with capability inheritance

Group capabilities

Version: 2.16.2

Autor: [itthinx](#)

Zuletzt aktualisiert: vor 5 Monaten

Erforderliche WordPress-Version: 5.5 oder höher

Kompatibel bis zu: 6.0.2

Erforderliche PHP-Version: 5.6.0 oder höher

Aktive Installationen: 20.000+

[WordPress.org-Plugin-Seite »](#)

[Plugin-Homepage »](#)

DURCHSCHNITTliche BEWERTUNG

★★★★★
(basierend auf 336 Abstimmungen)

BEWERTUNGEN

Lies alle Bewertungen auf WordPress.org oder schreib deine Eigene!

5 Sterne	304
4 Sterne	19
3 Sterne	4
2 Sterne	2
1 Stern	7

MITWIRKENDE

 [itthinx](#)  [Kento](#)

[Für dieses Plugin spenden »](#)

OpenID Connect

```
Meetup GraphQL API ▶ Prettify Merge Copy History  
1 query { self {  
2   id  
3   email  
4   name  
5   memberUrl  
6   memberPhotoUrl  
7 } }  
  
{  
  "data": {  
    "self": {  
      "id": "36867052",  
      "email": "domini@dominikdorn.com",  
      "name": "Dominik Dorn",  
      "memberUrl": "https://www.meetup.com/members/36867052",  
      "memberPhotoUrl": "https://secure.meetupstatic.com/photos/members/36867052/6/1/d/7/highres_244645047.jpeg"  
    }  
  }  
}
```

OpenID Connect - Generic Client

Client Settings

Enter your OpenID Connect identity provider settings.

Login Type

Auto Login - SSO

Select how the client (login form) should provide login options.

Client ID

java-vienna-wordpress

The ID this client will be recognized as when connecting the to Identity provider server.

Example: my-wordpress-client-id

Client Secret Key

xxxxx

Arbitrary secret key the server expects from this client. Can be anything, but should be very unique.

OpenID Scope

openid profile email roles offline_access microprofile-jwt

Space separated list of scopes this client should access.

Example: email profile openid offline_access

Login Endpoint URL

https://auth.java.wien/realms/Java-Vienna/protocol/openid-connect/auth

Identify provider authorization endpoint.

Example: https://example.com/oauth2/authorize

Userinfo Endpoint URL

https://auth.java.wien/realms/Java-Vienna/protocol/openid-connect/userinfo

Identify provider User information endpoint.

Example: https://example.com/oauth2/UserInfo

Token Validation Endpoint URL

https://auth.java.wien/realms/Java-Vienna/protocol/openid-connect/token

Identify provider token endpoint.

Example: https://example.com/oauth2/token

End Session Endpoint URL

https://auth.java.wien/realms/Java-Vienna/protocol/openid-connect/logout

Identify provider logout endpoint.

Example: https://example.com/oauth2/logout

ACR values

Use a specific defined authentication contract from the IDP - optional.

Identity Key

id

Where in the user claim array to find the user's identification data. Possible standard values: preferred_username

Example: preferred_username

OpenID Connect

java-vienna-wordpress OpenID Connect

Clients are applications and services that can request authentication of a user.

Settings Keys Credentials Roles Client scopes Authorization Service

General Settings

Client ID * ? java-vienna-wordpress

Name ? Java Vienna Wordpress

Description ?

Always display in console ? ☐ Off

Nickname Key

Where in the user claim array to find the user's nickname. Possible standard values: preferred_username, name, or sub.
Example: preferred_username

Email Formatting

String from which the user's email address is built. Specify "{email}" as long as the user claim contains an email claim.
Example: {email}

Display Name Formatting

String from which the user's display name is built.
Example: {given_name} {family_name}

Identify with User Name ☐

If checked, the user's identity will be determined by the user name instead of the email address.

State time limit

State valid time in seconds. Defaults to 180

Enable Refresh Token ☒

If checked, support refresh tokens used to obtain access tokens from supported IDPs.

WordPress User Settings

Modify the interaction between OpenID Connect and WordPress users.

Link Existing Users ☒

If a WordPress account already exists with the same identity as a newly-authenticated user over OpenID Connect, login as that user instead of generating an error.

Create user if does not exist ☒

If the user identity is not linked to an existing WordPress user, it is created. If this setting is not enabled, and if the user authenticates with an account which is not linked to an existing WordPress user, then the authentication will fail.

Redirect Back to Origin Page ☐

After a successful OpenID Connect authentication, this will redirect the user back to the page on which they clicked the OpenID Connect login button. This will cause the login process to proceed in a traditional WordPress fashion. For example, users logging in through the default wp-login.php page would end up on the WordPress Dashboard and users logging in through the WooCommerce "My Account" page would end up on their account page.

Redirect to the login screen when session is expired ☒

When enabled, this will automatically redirect the user back to the WordPress login page if their access token has expired.


Authorization Settings


Control the authorization mechanics of the site.


Enforce Privacy ☐


Require users be logged in to see the site.


Groups


 Dashboard


 Beiträge


 Medien


 Seiten


 Kommentare


 **Groups**


 Gruppen


 Kapazitäten


 **Optionen**


 Add-Ons


 Design

 Plugins

 Benutzer

 Werkzeuge

 **Einstellungen**

 Menü einklappen

Groups Optionen

Speichern

Zugangsbeschränkungen

Beitragstypen

Zugangsbeschränkungen für diese Beitragstypen zeigen.

☒ Medien

attachment

☒ Seite

page

☒ Beitrag

post

☐ Individuelles CSS

custom_css

☐ Änderungs-Set

customize_changeset

☒ Wiederverwendbarer Block

wp_block

☒ Navigationsmenü-Eintrag

nav_menu_item

☐ oEmbed-Antwort

oembed_cache

☐ Revision

revision

☐ Benutzer-Anfrage

user_request

☐ Globale Stile

wp_global_styles

☒ Navigationsmenü

wp_navigation

☐ Template

wp_template

☐ Template-Teil

wp_template_part

Groups

The screenshot shows the Joomla! Groups management interface. On the left is a dark sidebar with navigation links: Dashboard, Beiträge, Medien, Seiten, Kommentare, Groups (highlighted), Gruppen, Kapazitäten, Optionen, Add-Ons, Design, and Plugins. The main content area is titled 'Gruppen' and includes a '+ Neue Gruppe' button. Below the title are filter fields for 'Gruppen-ID' and 'Gruppenname'. A section for 'Kapazitäten ...' contains an 'Aktion wählen' dropdown and an 'Anwenden' button. A table lists the groups with columns for selection, ID, Gruppe, and Beschreibung. One group is listed: ID 1, Gruppe Registered (2).

	ID	Gruppe ▲	Beschreibung
<input type="checkbox"/>	1	Registered (2)	

Groups

The screenshot shows the WordPress Groups page editor. The main content area displays the title 'Mitgliederbereich' and two paragraphs of placeholder text. A small black square with a white plus sign is visible on the right side of the main content area. The right sidebar contains several settings sections: 'Seite' (Page), 'Beitragsbild' (Featured Image), 'Diskussion' (Discussion), 'Seiten-Attribute' (Page Attributes), and 'Gruppen' (Groups). The 'Gruppen' section is expanded, showing a 'Lesen' (Read) dropdown menu with 'Registered' selected. A red arrow points to the 'Registered' option in the dropdown menu.

WordPress Groups page editor interface.

Main Content:

Mitgliederbereich

Willkommen im Mitgliederbereich von Java-Vienna!

Hier kommen in Zukunft exklusive Inhalte für Mitglieder unseres Meetups!

Right Sidebar (Settings):

- Seite** (Page): </mitgliederbereich/>
- Beitragsbild** (Featured Image): Beitragsbild festlegen
- Diskussion** (Discussion): ☐ Kommentare erlauben
- Seiten-Attribute** (Page Attributes):
 - Übergeordnete Seite:
 - Reihenfolge:
- Gruppen** (Groups):
 - Lesen: **Registered** (selected)
 - Beschränkt die Seite-Sichtbarkeit auf Mitglieder der ausgewählten Gruppen.

Groups

Seiten [Erstellen](#)

Ansicht anpassen ▼

Hilfe ▼

Alle (4) | [Veröffentlichte](#) (4) | [Papierkorb](#) (8)

[Seiten durchsuchen](#)

Mehrfachaktionen ▼

[Übernehmen](#)

Alle Daten ▼

Gruppen ...

[Auswahl einschränken](#)

4 Einträge

<input type="checkbox"/> Titel	Autor		Datum	Gruppen
<input type="checkbox"/> Datenschutzerklärung — Seite Datenschutzerklärung	admin	—	Veröffentlicht 08.09.2022 um 19:52 Uhr	
<input type="checkbox"/> Impressum	admin	—	Veröffentlicht 11.09.2022 um 2:25 Uhr	
<input type="checkbox"/> Mitgliederbereich	admin	—	Veröffentlicht 02.10.2022 um 11:24 Uhr	Registered
<input type="checkbox"/> Willkommen! — Startseite	admin	—	Veröffentlicht 08.09.2022 um 19:52 Uhr	
<input type="checkbox"/> Titel	Autor		Datum	Gruppen

Mehrfachaktionen ▼

[Übernehmen](#)

4 Einträge

Quarkus + Keycloak Integration

Quarkus - Keycloak Integration

Links

- <https://quarkus.io/guides/security-openid-connect>
-
- <https://quarkus.io/guides/security-openid-connect-web-authentication>
- Template Engine Qute: <https://quarkus.io/guides/qute>

Quarkus Templating “Qute”

Quarkus Templating “Qute”

Guide: <https://quarkus.io/guides/qute>
<https://quarkus.pro/guides/qute-reference.html>

```
<dependency>  
  <groupId>io.quarkus</groupId>  
  <artifactId>quarkus-resteasy-reactive-qute</artifactId>  
</dependency>
```