Using Java for a Greenfield Project

From idea to implementation. All by yourself. Featuring boring technologies only!

Alexander Reelsen

alex@spinscale.de | @spinscale

Today's goal

> How to run and maintain a modern web java application with minimal resources

About me

- By day: Community @ Elastic
- By night: sleep
- In between: Elasticsearch enthusiast, web framework evaluator, serverless observer, distributed work proponent, basketball fan
- Java since '07

Why greenfield?

- Assessing current skills & technologies
- Acquiring new skills & technologies
- Communication/Feedback
- Business idea
- Test out competition
- Understand the full application lifecycle
- `Just for fun`

Limitations

- Full-time job
- Family & Kids
- Covid-19
- No full stack developer

Management summary

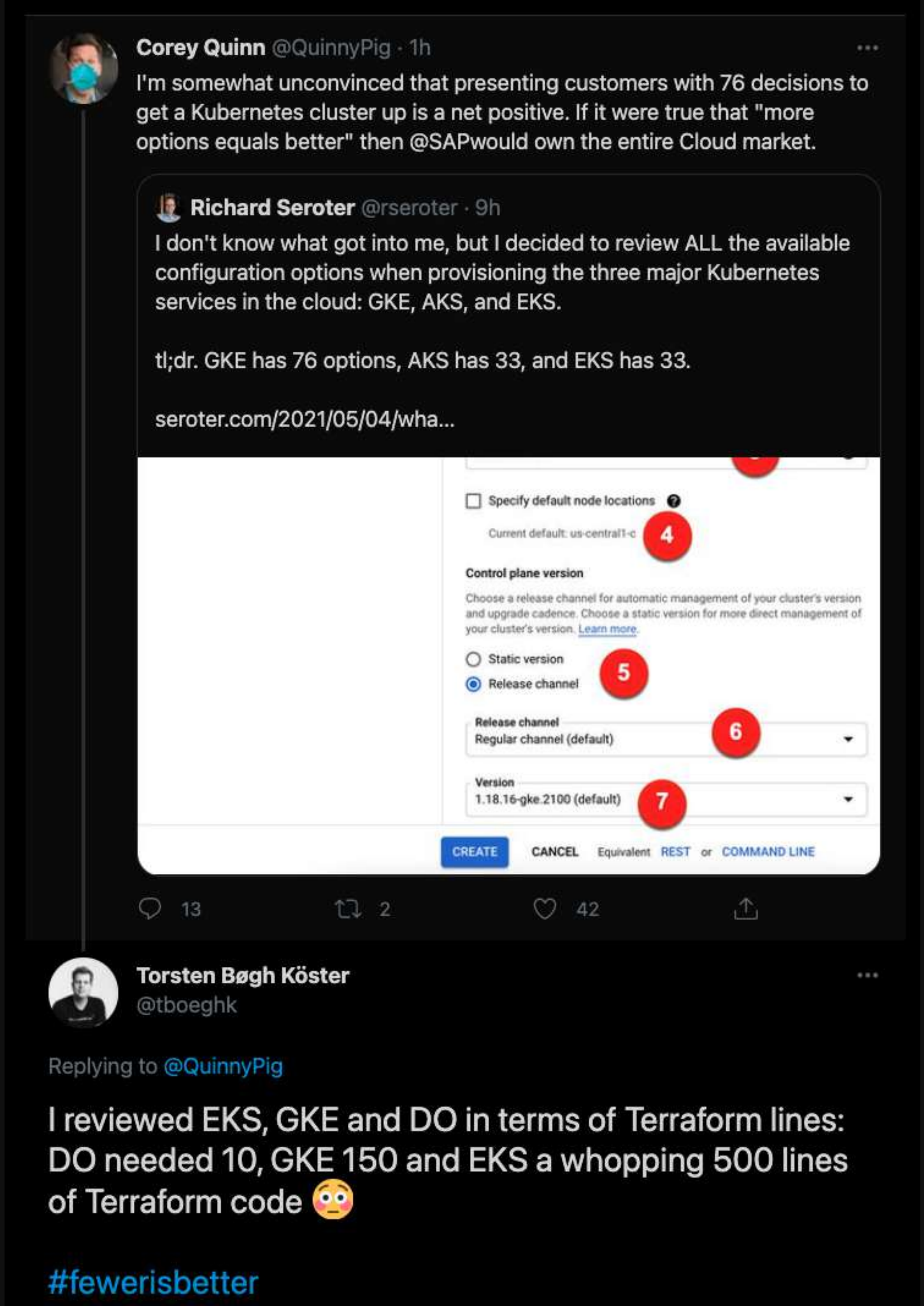
- Effectiveness - Do the **right** things
- Efficiency - Do the things **right**
- Efficiency requires effectiveness

Operational requirements

- Do the needful - *but not more*
- Operational simplicity - *No Kubernetes!*

Operational requirements

- Do the needful - *but not more*
- Operational simplicity - *No Kubernetes!*



Corey Quinn @QuinnyPig · 1h

I'm somewhat unconvinced that presenting customers with 76 decisions to get a Kubernetes cluster up is a net positive. If it were true that "more options equals better" then @SAPwould own the entire Cloud market.

> **Richard Seroter** @rseroter · 9h
>
> I don't know what got into me, but I decided to review ALL the available configuration options when provisioning the three major Kubernetes services in the cloud: GKE, AKS, and EKS.
>
> tl;dr. GKE has 76 options, AKS has 33, and EKS has 33.
>
> seroter.com/2021/05/04/wha...

□ Specify default node locations

Current default: us-central1-c

Control plane version

Choose a release channel for automatic management of your cluster's version and upgrade cadence. Choose a static version for more direct management of your cluster's version. Learn more.

○ Static version

● Release channel

Release channel: Regular channel (default)

Version: 1.18.16-gke.2100 (default)

Torsten Bøgh Köster
@tboeghk

Replying to @QuinnyPig

I reviewed EKS, GKE and DO in terms of Terraform lines: DO needed 10, GKE 150 and EKS a whopping 500 lines of Terraform code 😳

#fewerisbetter

Operational requirements

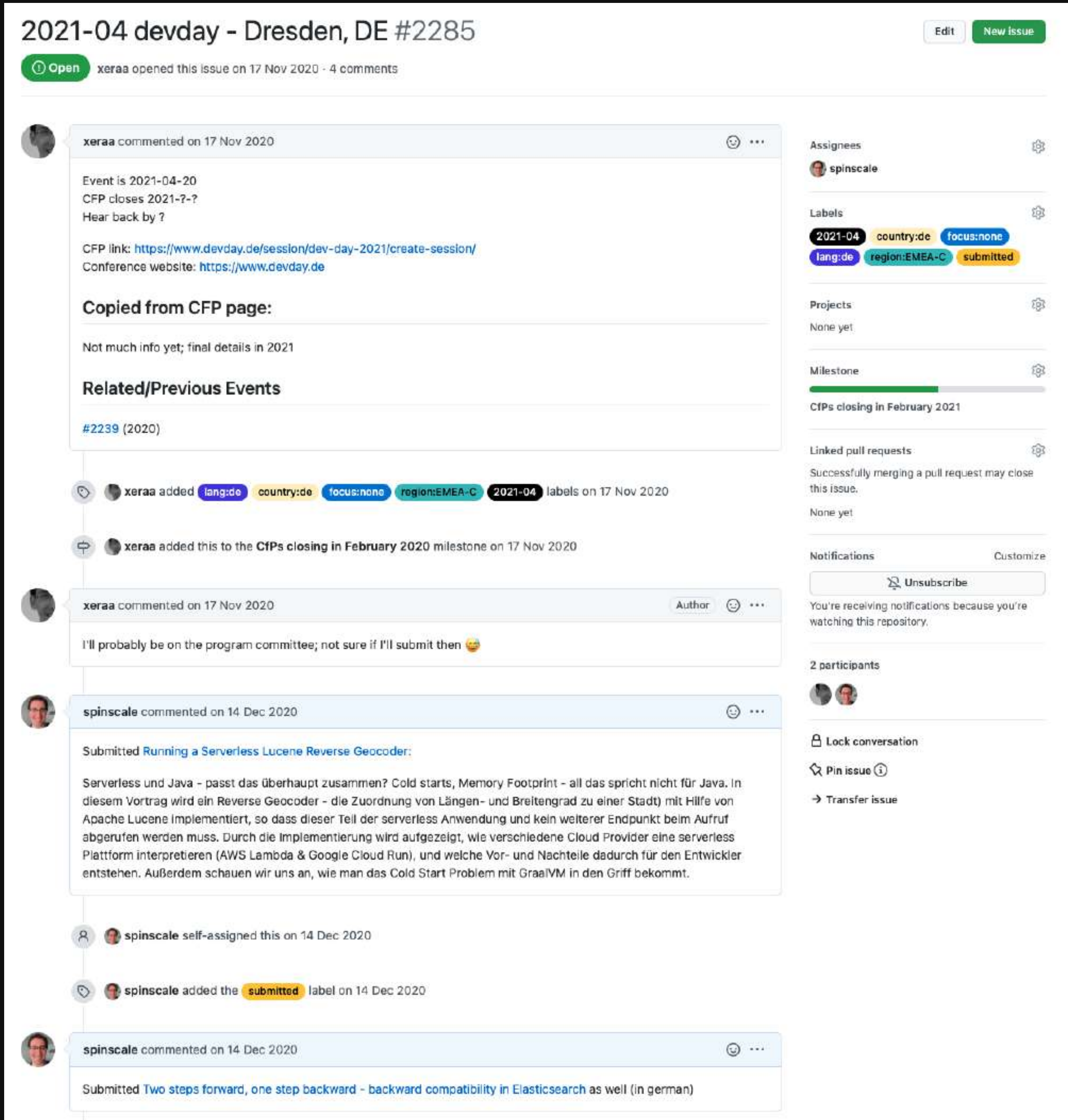
- Do the needful - *but not more*
- Operational simplicity - *No Kubernetes!*
- Easy rollouts - *git pushes*
- No login handling - *OAuth*
- Optimize for (single) developer productivity
- Small deployments, few dependencies!
- Ability to debug

Idea

- Developer Advocate Conference & Talks Management

Conferences @Elastic

- Labels for tagging/reporting
- Milestone for CfP closure
- No fixed date fields
- No final closing report (pictures, sponsors, competitors)
- No easy overview over submitted talks



2021-04 devday - Dresden, DE #2285

Open xeraa opened this issue on 17 Nov 2020 · 4 comments

xeraa commented on 17 Nov 2020

Event is 2021-04-20
CFP closes 2021-?-?
Hear back by ?

CFP link: https://www.devday.de/session/dev-day-2021/create-session/
Conference website: https://www.devday.de

Copied from CFP page:

Not much info yet; final details in 2021

Related/Previous Events

#2239 (2020)

xeraa added lang:de country:de focus:none region:EMEA-C 2021-04 labels on 17 Nov 2020

xeraa added this to the **CfPs closing in February 2020** milestone on 17 Nov 2020

xeraa commented on 17 Nov 2020 Author

I'll probably be on the program committee; not sure if I'll submit then 😅

spinscale commented on 14 Dec 2020

Submitted Running a Serverless Lucene Reverse Geocoder:

Serverless und Java - passt das überhaupt zusammen? Cold starts, Memory Footprint - all das spricht nicht für Java. In diesem Vortrag wird ein Reverse Geocoder - die Zuordnung von Längen- und Breitengrad zu einer Stadt) mit Hilfe von Apache Lucene implementiert, so dass dieser Teil der serverless Anwendung und kein weiterer Endpunkt beim Aufruf abgerufen werden muss. Durch die Implementierung wird aufgezeigt, wie verschiedene Cloud Provider eine serverless Plattform interpretieren (AWS Lambda & Google Cloud Run), und welche Vor- und Nachteile dadurch für den Entwickler entstehen. Außerdem schauen wir uns an, wie man das Cold Start Problem mit GraalVM in den Griff bekommt.

spinscale self-assigned this on 14 Dec 2020

spinscale added the submitted label on 14 Dec 2020

spinscale commented on 14 Dec 2020

Submitted Two steps forward, one step backward - backward compatibility in Elasticsearch as well (in german)

Assignees: spinscale

Labels: 2021-04 country:de focus:none lang:de region:EMEA-C submitted

Projects: None yet

Milestone: CfPs closing in February 2021

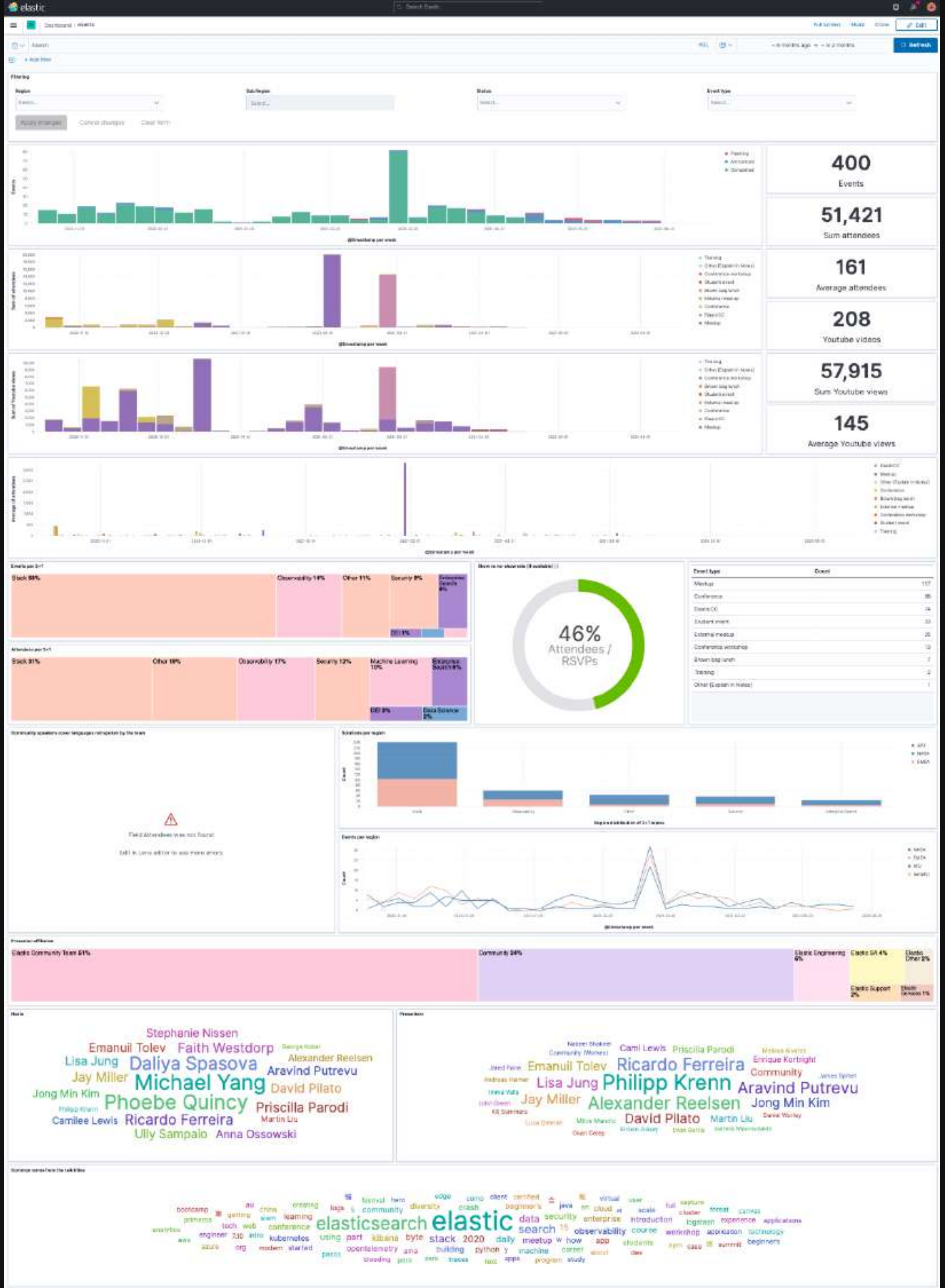
Linked pull requests: Successfully merging a pull request may close this issue. None yet

Notifications: You're receiving notifications because you're watching this repository.

2 participants

Stats (+Airtable)

- Participants
- Youtube links



Event type	Count
Meetup	117
Conference	95
ElasticCC	74
Student event	33
External meetup	25
Conference workshop	12
Brown bag lunch	7
Training	2
Other (Explain in Notes)	1

Project Management

- Not needed, you got it all in your head, or not?
- Pen & Paper for initial modelling (including entities)
- Jetbrains Spaces, mainly as repo and kanban board

Boring technologies

- Javalin
- JDBI
- Pac4j
- Frontend: htmx, apache echarts, custom tailwind css build
- Database: Postgres/H2

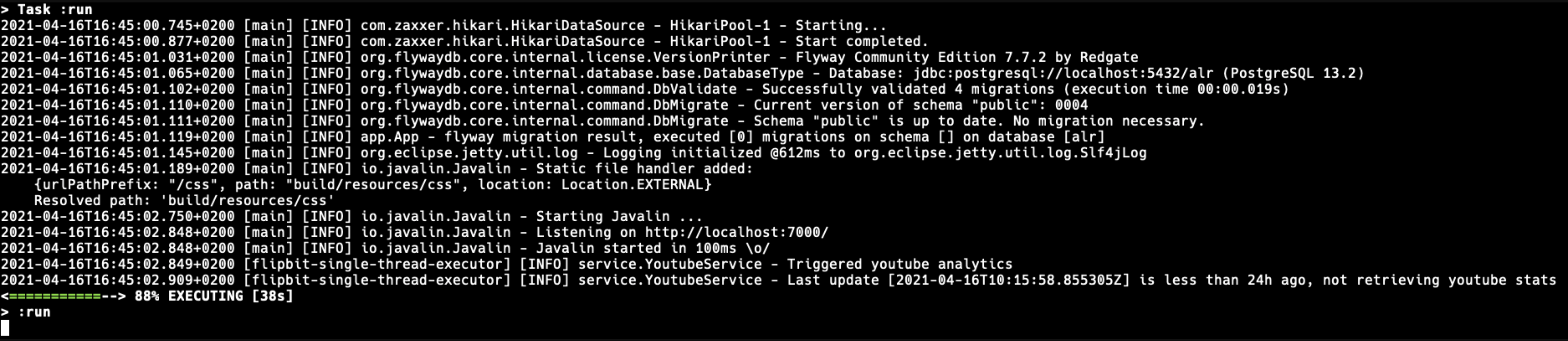
But why no...?

- *Kotlin/Node/Ruby?* Didn't want to learn a new language, getting results fast
- *Spring Boot?* Not increasing my productivity at all...
- *SPA with a JSON API?* Making some things easier, and others harder. Requires to learn Svelte/Vue, TypeScript, node ecosystem
- *Quarkus/Micronaut* Not comfortable enough yet, but a good candidate for the future
- *Serverless* Google Cloud Run is a good candidate, app must be stateless

HTTP Server & Framework

- Javalin
- Written in Kotlin
- On top of Jetty
- Sync/Async APIs
- Various Template Languages

Javalin



```
> Task :run
2021-04-16T16:45:00.745+0200 [main] [INFO] com.zaxxer.hikari.HikariDataSource - HikariPool-1 - Starting...
2021-04-16T16:45:00.877+0200 [main] [INFO] com.zaxxer.hikari.HikariDataSource - HikariPool-1 - Start completed.
2021-04-16T16:45:01.031+0200 [main] [INFO] org.flywaydb.core.internal.license.VersionPrinter - Flyway Community Edition 7.7.2 by Redgate
2021-04-16T16:45:01.065+0200 [main] [INFO] org.flywaydb.core.internal.database.base.DatabaseType - Database: jdbc:postgresql://localhost:5432/alr (PostgreSQL 13.2)
2021-04-16T16:45:01.102+0200 [main] [INFO] org.flywaydb.core.internal.command.DbValidate - Successfully validated 4 migrations (execution time 00:00.019s)
2021-04-16T16:45:01.110+0200 [main] [INFO] org.flywaydb.core.internal.command.DbMigrate - Current version of schema "public": 0004
2021-04-16T16:45:01.111+0200 [main] [INFO] org.flywaydb.core.internal.command.DbMigrate - Schema "public" is up to date. No migration necessary.
2021-04-16T16:45:01.119+0200 [main] [INFO] app.App - flyway migration result, executed [0] migrations on schema [] on database [alr]
2021-04-16T16:45:01.145+0200 [main] [INFO] org.eclipse.jetty.util.log - Logging initialized @612ms to org.eclipse.jetty.util.log.Slf4jLog
2021-04-16T16:45:01.189+0200 [main] [INFO] io.javalin.Javalin - Static file handler added:
    {urlPathPrefix: "/css", path: "build/resources/css", location: Location.EXTERNAL}
    Resolved path: 'build/resources/css'
2021-04-16T16:45:02.750+0200 [main] [INFO] io.javalin.Javalin - Starting Javalin ...
2021-04-16T16:45:02.848+0200 [main] [INFO] io.javalin.Javalin - Listening on http://localhost:7000/
2021-04-16T16:45:02.848+0200 [main] [INFO] io.javalin.Javalin - Javalin started in 100ms \o/
2021-04-16T16:45:02.849+0200 [flipbit-single-thread-executor] [INFO] service.YoutubeService - Triggered youtube analytics
2021-04-16T16:45:02.909+0200 [flipbit-single-thread-executor] [INFO] service.YoutubeService - Last update [2021-04-16T10:15:58.855305Z] is less than 24h ago, not retrieving youtube stats
<===========--> 88% EXECUTING [38s]
> :run
```

Templating Language

- JTE - *aka return of the JSP*
- *Fast*
- Autocomplete in the IDE
- Custom tags
- Template precompilation possible
- Further optimizations to reduce GC/memory and increase throughput

JTE

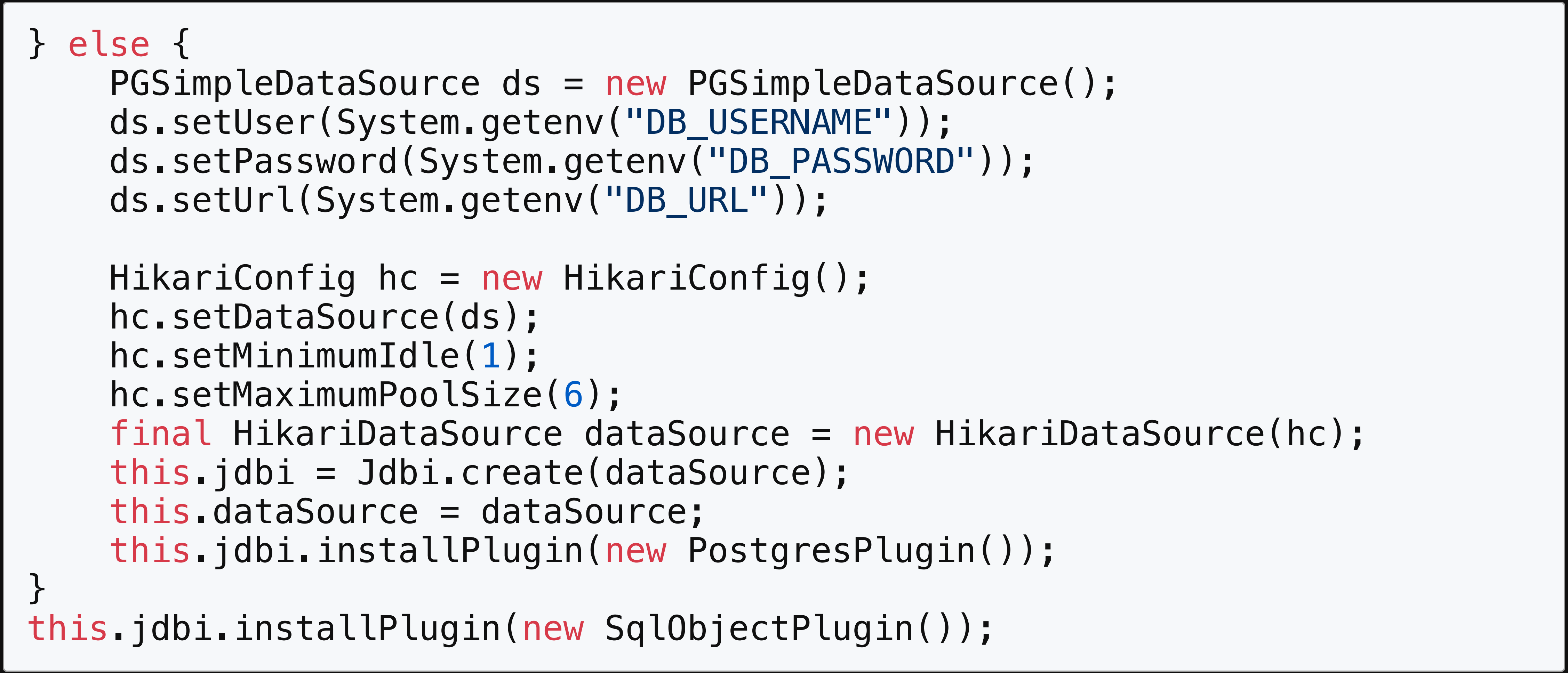
```
@import static utils.TemplateUtils.capitalize

@param String organization
@param Talk talk
@param List<Event> events

<div class="col-span-1">
  <label for="type" class="block text-sm font-medium text-gray-700 font-mono">Type</label>
  <select id="type" name="type" class="mt-1 text-lg pl-2 pr-8 shadow-sm focus:bg-yellow-100 focus:border-none">
  @for(var type : ContentResource.ResourceType.values())
   <option value="${type.name()}">${capitalize(type.name())}</option>
  @endfor
  </select>
</div>
```

Persistence Layer

1. Elasticsearch using templating to prevent Object Mapping and not take the whole of Elasticsearch as dependency into
2. Postgres/H2 using ebean
3. Postgres/H2 using JDBI

Database setup

```java
if (System.getenv("DB_URL").startsWith("jdbc:h2:")) {
    JdbcDataSource jdbcDataSource = new JdbcDataSource();
    jdbcDataSource.setUser(System.getenv("DB_USERNAME"));
    jdbcDataSource.setPassword(System.getenv("DB_PASSWORD"));
    jdbcDataSource.setUrl(System.getenv("DB_URL"));
    this.dataSource = jdbcDataSource;

    this.jdbi = Jdbi.create(dataSource);
    this.jdbi.installPlugin(new H2DatabasePlugin());
} else {
```



```java
} else {
    PGSimpleDataSource ds = new PGSimpleDataSource();
    ds.setUser(System.getenv("DB_USERNAME"));
    ds.setPassword(System.getenv("DB_PASSWORD"));
    ds.setUrl(System.getenv("DB_URL"));

    HikariConfig hc = new HikariConfig();
    hc.setDataSource(ds);
    hc.setMinimumIdle(1);
    hc.setMaximumPoolSize(6);
    final HikariDataSource dataSource = new HikariDataSource(hc);
    this.jdbi = Jdbi.create(dataSource);
    this.dataSource = dataSource;
    this.jdbi.installPlugin(new PostgresPlugin());
}
this.jdbi.installPlugin(new SqlObjectPlugin());
```

JDBI

> Jdbi provides convenient, idiomatic access to relational data in Java.

JDBI

> Jdbi provides convenient, idiomatic access to relational data in Java.

> Unlike an ORM, we do not aim to provide a complete object relational mapping framework - instead of that hidden complexity, we provide building blocks that allow you to construct the mapping between relations and objects as appropriate for your application.

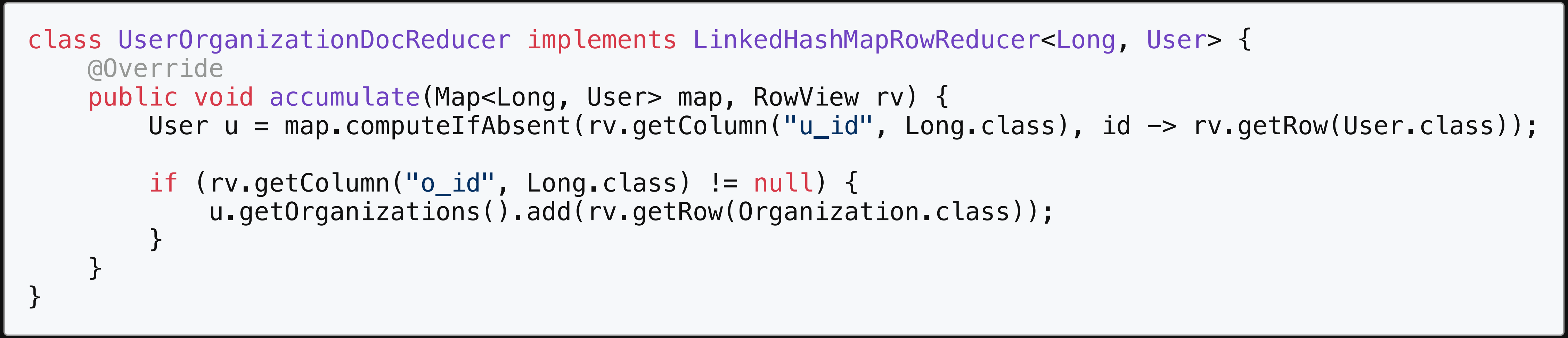
User entity

```java
public final class User {

  private Long id;
  ...
  private List<Organization> organizations = new ArrayList<>(1);

}
```

JDBI

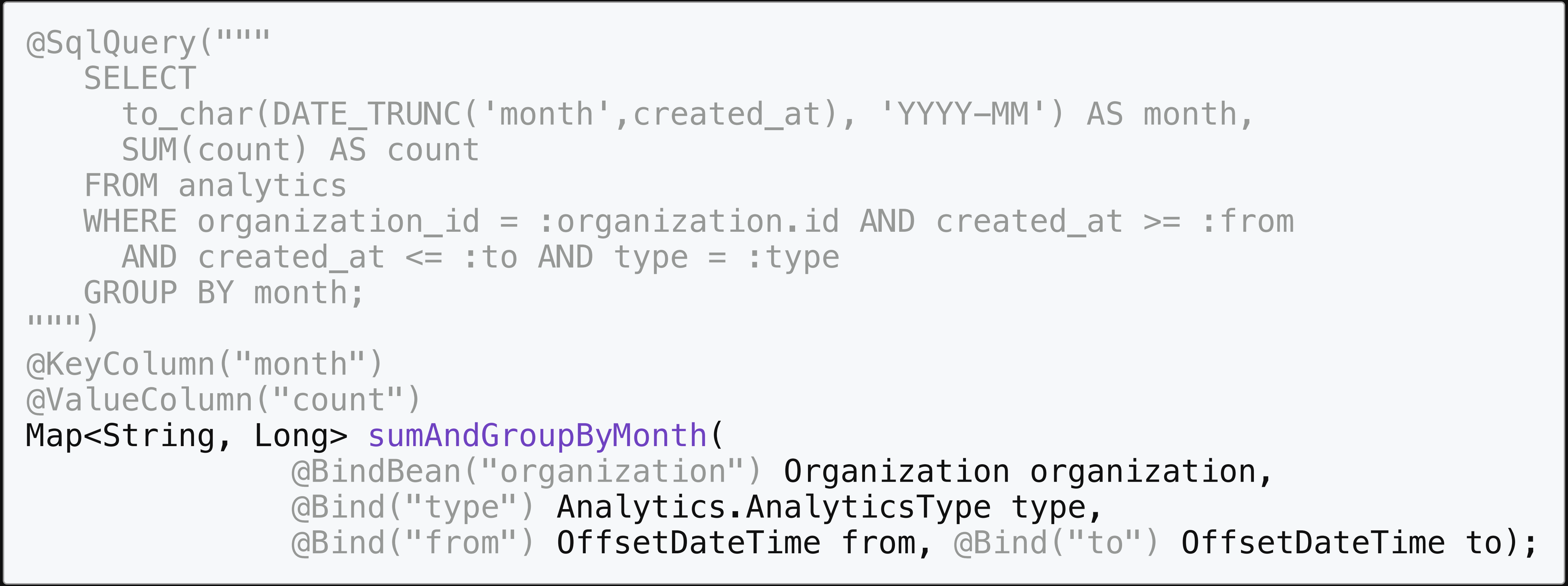
```
@RegisterBeanMapper(value = User.class, prefix = "u")
@RegisterBeanMapper(value = Organization.class, prefix = "o")
@UseRowReducer(UserOrganizationDocReducer.class)
@SqlQuery("""
        SELECT
            u.id u_id, u.username u_username, u.display_name u_display_name,
            u.picture_url u_picture_url, u.email u_email, u.created_at u_created_at,
            u.modified_at u_modified_at,
            o.id o_id, o.name o_name, o.created_at o_created_at, o.modified_at o_modified_at
        FROM users u
        JOIN user_organizations uo ON uo.user_id = u.id
        JOIN organization o ON uo.organization_id = o.id
        WHERE u.username = ?
        """)
User findByUsername(String username);
```

JDBI - RowReducer



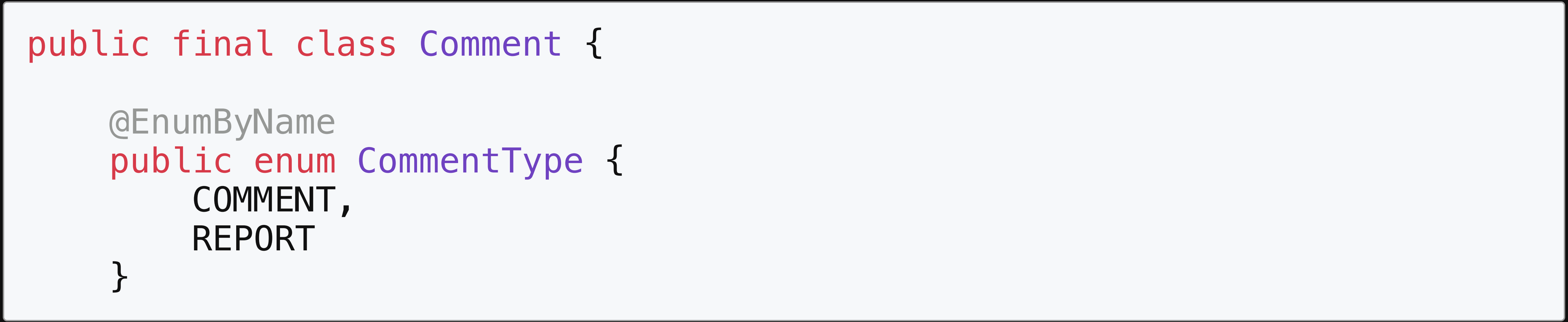
```
class UserOrganizationDocReducer implements LinkedHashMapRowReducer<Long, User> {
    @Override
    public void accumulate(Map<Long, User> map, RowView rv) {
        User u = map.computeIfAbsent(rv.getColumn("u_id", Long.class), id -> rv.getRow(User.class));

        if (rv.getColumn("o_id", Long.class) != null) {
            u.getOrganizations().add(rv.getRow(Organization.class));
        }
    }
}
```

JDBI



```
@SqlQuery("""
   SELECT
     to_char(DATE_TRUNC('month',created_at), 'YYYY-MM') AS month,
     SUM(count) AS count
   FROM analytics
   WHERE organization_id = :organization.id AND created_at >= :from
     AND created_at <= :to AND type = :type
   GROUP BY month;
""")
@KeyColumn("month")
@ValueColumn("count")
Map<String, Long> sumAndGroupByMonth(
              @BindBean("organization") Organization organization,
              @Bind("type") Analytics.AnalyticsType type,
              @Bind("from") OffsetDateTime from, @Bind("to") OffsetDateTime to);
```

JDBI - enums



```
public final class Comment {

    @EnumByName
    public enum CommentType {
        COMMENT,
        REPORT
    }
```

JDBI - security

- Multi-tenant system, implicit security issue
- How to prevent unauthorized access?
- Every DAO read operation requires an organization argument
- Organization cannot be injected from outside

Flyway Database Migrations

```
// run the flyway migration before the app starts
Flyway flyway = Flyway.configure().dataSource(dataSource).load();
final MigrateResult result = flyway.migrate();
logger.info("flyway migration result, executed [{}] migrations on schema [{}] on database [{}]",
        result.migrationsExecuted, result.schemaName, result.database);
```

Local system setup via direnv

direnv - unclutter your .profile

`.envrc`

```
dotenv
```

Local system setup via direnv

.env

```
GITHUB_KEY="<<KEY>>"
GITHUB_SECRET="<<SECRET>>"
GITHUB_CALLBACK_URL="http://localhost:7000/auth/login/github"
MODE=development
PORT="7000"
YOUTUBE_TOKEN="<<YT_TOKEN>>"

#DB_USERNAME="sa"
#DB_PASSWORD=""
#DB_URL="jdbc:h2:mem:flipbit_in_memory;MODE=PostgreSQL;DB_CLOSE_DELAY=-1;DATABASE_TO_UPPER=false"
DB_USERNAME="alr"
DB_PASSWORD=""
DB_URL="jdbc:postgresql://localhost:5432/alr"
```

Development vs. production mode

```java
public enum Mode {

    PRODUCTION, DEVELOPMENT;

    public static Mode current() {
        final String mode = System.getenv("MODE").toUpperCase(Locale.ROOT);
        return Mode.valueOf(mode);
    }

    public static boolean isProduction() { return current() == PRODUCTION; }

    public static boolean isDevelopment() { return current() == DEVELOPMENT; }
}
```

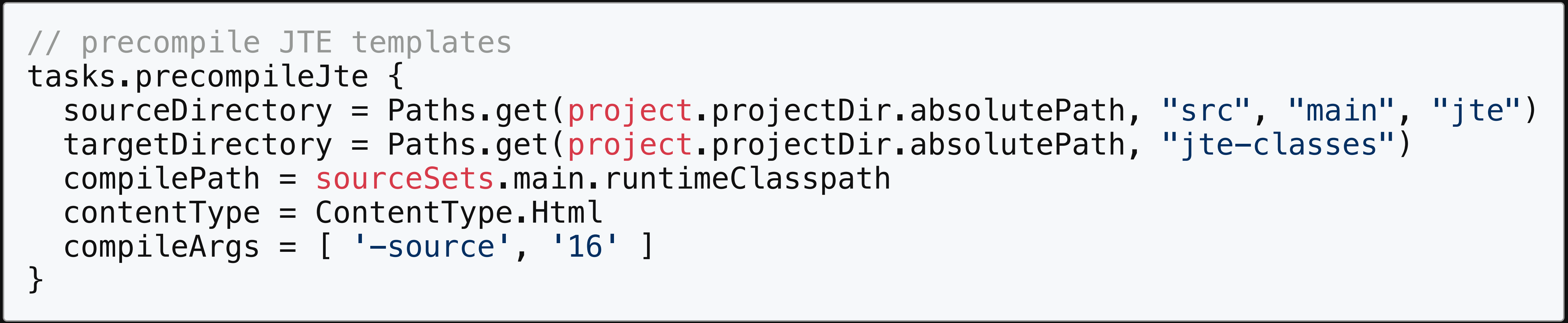
Development vs. production mode

```
// this is the location in the dockerfile
// probably makes more sense to put this into an env var
if (Mode.isProduction()) {
    config.addStaticFiles("/css", "/flipbit-app/resources/css", Location.EXTERNAL);
} else {
    config.addStaticFiles("/css", "build/resources/css", Location.EXTERNAL);
}
```

Development vs. production mode

```java
final TemplateEngine templateEngine;
// no template compilation in prod, because of the read only docker image, templates are created as part of the build
if (Mode.isProduction()) {
    templateEngine = TemplateEngine.createPrecompiled(Paths.get("/jte-classes"), ContentType.Html);
} else {
    final DirectoryCodeResolver resolver = new DirectoryCodeResolver(Paths.get("src", "main", "jte"));
    templateEngine = TemplateEngine.create(resolver, ContentType.Html);
    templateEngine.precompileAll();
}
JavalinJte.configure(templateEngine);
```

Precompiled templates



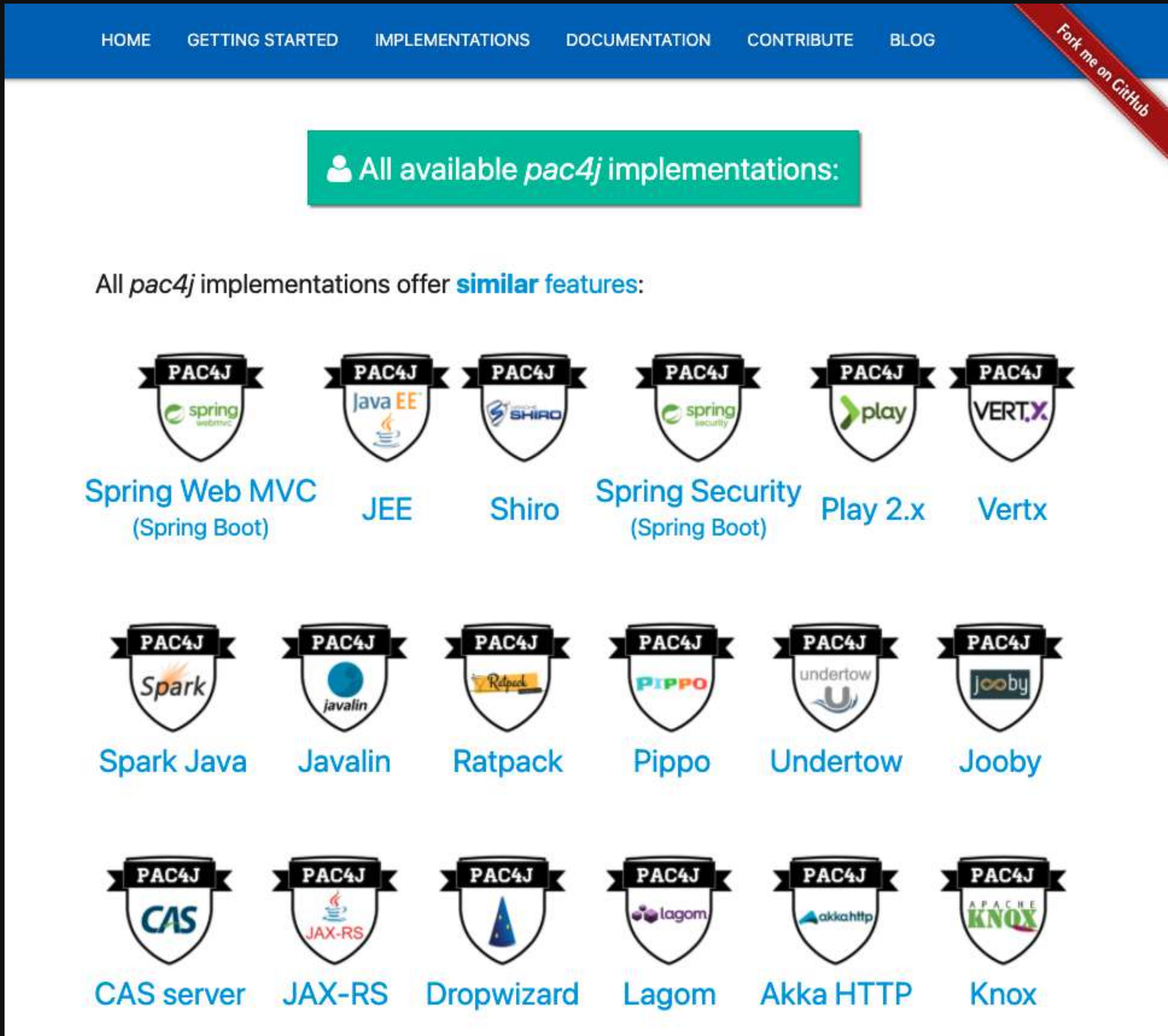
```
// precompile JTE templates
tasks.precompileJte {
  sourceDirectory = Paths.get(project.projectDir.absolutePath, "src", "main", "jte")
  targetDirectory = Paths.get(project.projectDir.absolutePath, "jte-classes")
  compilePath = sourceSets.main.runtimeClasspath
  contentType = ContentType.Html
  compileArgs = [ '-source', '16' ]
}
```

Authentication

- Storing own passwords is not an option in a hobby project
- OAuth is the perfect fit to give away responsibilities
- Login via github, then storing in session
- Advantage: Unique usernames
- Solution: pac4j

Pac4j

- javalin-pac4j
- Support for OAuth, SAML, CAS, OpenID Connect, SPNEGO
- OAuth: GitHub, LinkedIn, Dropbox, Facebook, Google, Twitter, Wordpress (Generic for custom endpoints)



All available *pac4j* implementations:

All *pac4j* implementations offer **similar** features:

Spring Web MVC (Spring Boot)

JEE

Shiro

Spring Security (Spring Boot)

Play 2.x

Vertx

Spark Java

Javalin

Ratpack

Pippo

Undertow

Jooby

CAS server

JAX-RS

Dropwizard

Lagom

Akka HTTP

Knox

App Setup

- Big `App` class, that could be modularized
- DB setup
- DAOs
- Routes & Controllers
- Background job initialization

App Setup

```
OrganizationDao organizationDao = jdbi.onDemand(OrganizationDao.class);
UserDao userDao = jdbi.onDemand(UserDao.class);
EventDao eventDao = jdbi.onDemand(EventDao.class);
AnalyticsDao analyticsDao = jdbi.onDemand(AnalyticsDao.class);
ContentResourceDao contentResourceDao = jdbi.onDemand(ContentResourceDao.class);
TalkDao talkDao = jdbi.onDemand(TalkDao.class);
SubmissionDao submissionDao = jdbi.onDemand(SubmissionDao.class);
CommentDao commentDao = jdbi.onDemand(CommentDao.class);
EventTagsDao eventTagsDao = jdbi.onDemand(EventTagsDao.class);
InvitationDao invitationDao = jdbi.onDemand(InvitationDao.class);
```

App Setup

```
ReportingController reportingController = new ReportingController(eventDao, analyticsDao, organizationDao);
SearchController searchController = new SearchController(eventDao, talkDao);
TalkController talkController = new TalkController(talkDao, eventDao, contentResourceDao, submissionDao);
SettingsController settingsController = new SettingsController(userDao, invitationDao);
Security security = new Security(organizationDao, userDao, invitationDao);
InviteeController inviteeController = new InviteeController(invitationDao);
```

App Setup

```
app.routes(() -> {
    // login button redirection
    get("/auth/login/github", security.callbackHandler());
    // ensure that logout kills the session, that could be replaced with checkUserSessionHandler()?
    before("/auth/logout", security.securityHandler());
    get("/auth/logout", security.logoutHandler());
    // the main handler that ensures, everything behind /app is secured and we have a user session
    before("/app/*", security.checkUserSessionHandler());

    ...
}
```

App setup

```
get("/app/:organization/dashboard", reportingController.dashboard());
get("/app/:organization/reporting", reportingController.reportingEcharts());
get("/app/:organization/settings", settingsController.settings());
post("/app/:organization/search", searchController.search());
```

App setup

```
get("/app/:organization/event", eventController.renderEvents());
get("/app/:organization/event/new", eventController.showCreateEventForm());
post("/app/:organization/event", eventController.createEvent());
get("/app/:organization/event/:id", eventController.renderEvent());
post("/app/:organization/event/:id", eventController.updateExistingEvent());
get("/app/:organization/event/:id/edit", eventController.editEventForm());
post("/app/:organization/event/:id/delete", eventController.deleteEvent());
post("/app/:organization/event/:id/comment", eventController.addComment());
post("/app/:organization/event/:id/submission", eventController.addSubmission());
post("/app/:organization/event/:id/submission/:talkId/status/:status", eventController.changeTalkStatus());
post("/app/:organization/event/:id/submission/:talkId/attendees", eventController.insertAttendeeAnalytics());
```

Sample controller

```
public Handler deleteEvent() {
    return ctx -> {
        Organization organization = ctx.sessionAttribute("organization");
        final Long id = ctx.pathParam("id", Long.class).get();
        eventDao.delete(id, organization.getId());
        ctx.redirect("/app/" + organization.getName() + "/event/");
    };
}
```

- Organization in session to ensure it cannot be modified
- Event deletion is based on organization to prevent deleting other's events

Sample controller

```java
public Handler deleteEvent() {
    return ctx -> {
        Organization organization = ctx.sessionAttribute("organization");
        final Long id = ctx.pathParam("id", Long.class).get();
        eventDao.delete(id, organization.getId());
        ctx.redirect("/app/" + organization.getName() + "/event/");
    };
}
```

```java
@SqlUpdate("UPDATE events SET active = false, modified_at = NOW() WHERE id = :event_id AND organization_id = :organization_id")
void delete(@Bind("event_id") long eventId, @Bind("organization_id") long organizationId);
```

Organizational check

```
User loggedInUser = ctx.sessionAttribute("user");
if (loggedInUser != null) {
    if (ctx.pathParamMap().containsKey("organization")) {
        final String pathOrganization = ctx.pathParam("organization");
        boolean isUserPartOfOrganization = loggedInUser.getOrganizations().stream()
            .anyMatch(o -> o.getName().equals(pathOrganization));
        if (isUserPartOfOrganization == false) {
            throw new NotFoundResponse("...");
        }
        final Organization organization = organizationDao.findByName(pathOrganization);
        ctx.sessionAttribute("organization", organization);
    }
}
```

- Can you spot the issue?

Testing

```
testImplementation "org.assertj:assertj-core:3.19.0"
testImplementation 'org.mockito:mockito-core:3.8.0'
testImplementation 'org.junit-pioneer:junit-pioneer:1.3.8'

testImplementation "org.testcontainers:junit-jupiter:1.15.2"
testImplementation "org.testcontainers:postgresql:1.15.2"

testImplementation 'nl.jqno.equalsverifier:equalsverifier:3.5.5'

testImplementation 'org.junit.jupiter:junit-jupiter-api:5.7.1'
testRuntimeOnly 'org.junit.jupiter:junit-jupiter-engine:5.7.1'
```

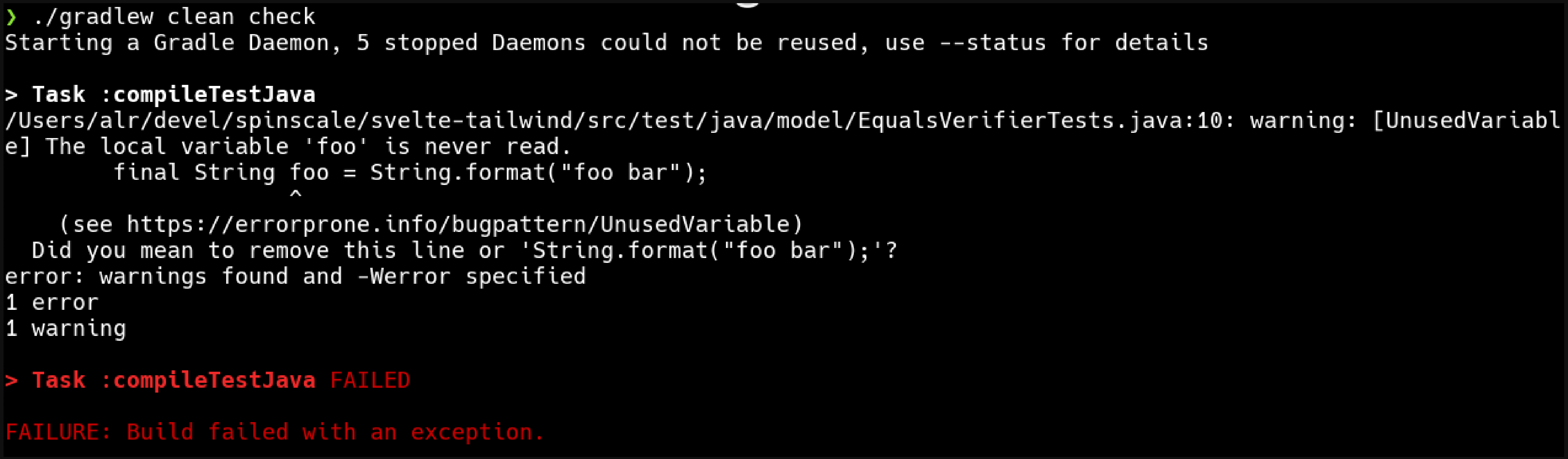
EqualsVerifier

```java
@Test
public void testEqualsVerifierUser() {
    EqualsVerifier.simple().forClass(User.class).verify();
}
```

Supporting Tools

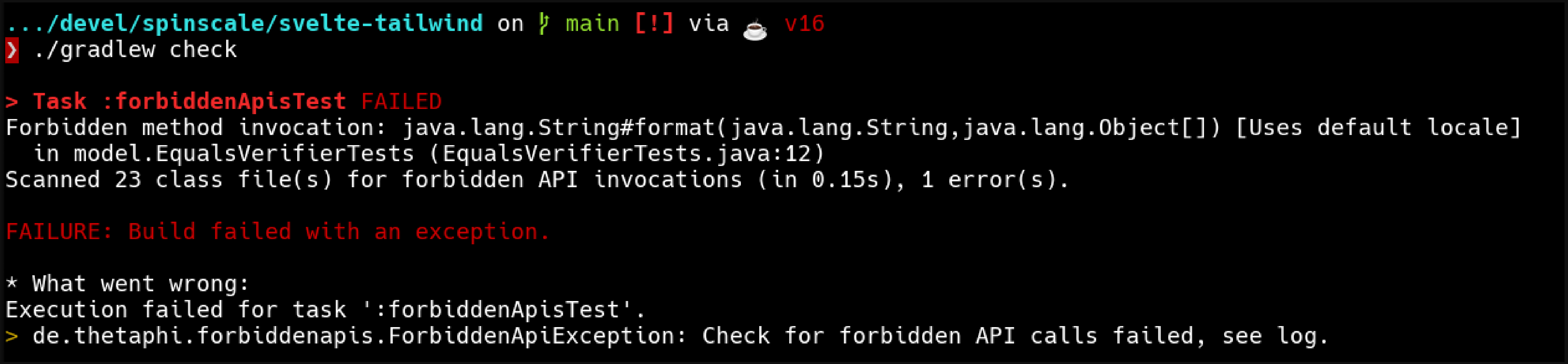
- Forbidden APIs
- Errorprone

Errorprone



```
❯ ./gradlew clean check
Starting a Gradle Daemon, 5 stopped Daemons could not be reused, use --status for details

> Task :compileTestJava
/Users/alr/devel/spinscale/svelte-tailwind/src/test/java/model/EqualsVerifierTests.java:10: warning: [UnusedVariabl
e] The local variable 'foo' is never read.
        final String foo = String.format("foo bar");
                     ^
    (see https://errorprone.info/bugpattern/UnusedVariable)
  Did you mean to remove this line or 'String.format("foo bar");'?
error: warnings found and -Werror specified
1 error
1 warning

> Task :compileTestJava FAILED

FAILURE: Build failed with an exception.
```

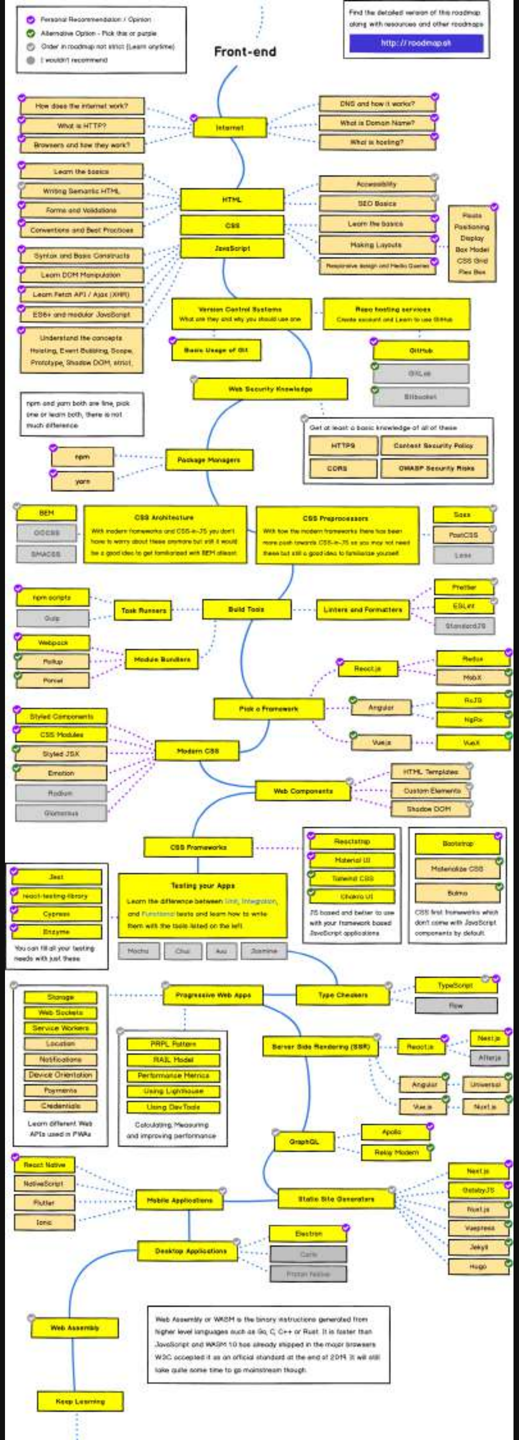
Forbidden APIS



```
.../devel/spinscale/svelte-tailwind on  main [!] via ☕ v16
❯ ./gradlew check

> Task :forbiddenApisTest FAILED
Forbidden method invocation: java.lang.String#format(java.lang.String,java.lang.Object[]) [Uses default locale]
  in model.EqualsVerifierTests (EqualsVerifierTests.java:12)
Scanned 23 class file(s) for forbidden API invocations (in 0.15s), 1 error(s).

FAILURE: Build failed with an exception.

* What went wrong:
Execution failed for task ':forbiddenApisTest'.
> de.thetaphi.forbiddenapis.ForbiddenApiException: Check for forbidden API calls failed, see log.
```

Frontend

See https://roadmap.sh/frontend



Find the detailed version of this roadmap along with resources and other roadmaps

npm and yarn both are fine, pick one or learn both, there is not much difference.

Get at least a basic knowledge of all of these

With modern frameworks and CSS-in-JS you don't have to worry about these anymore but still it would be a good idea to get familiarized with BEM atleast.

With how the modern frameworks there has been more push towards CSS-in-JS so you may not need these but still a good idea to familiarize yourself.

Learn the difference between Unit, Integration, and Functional tests and learn how to write them with the tools listed on the left.

You can fill all your testing needs with just these

JS based and better to use with your framework based JavaScript applications

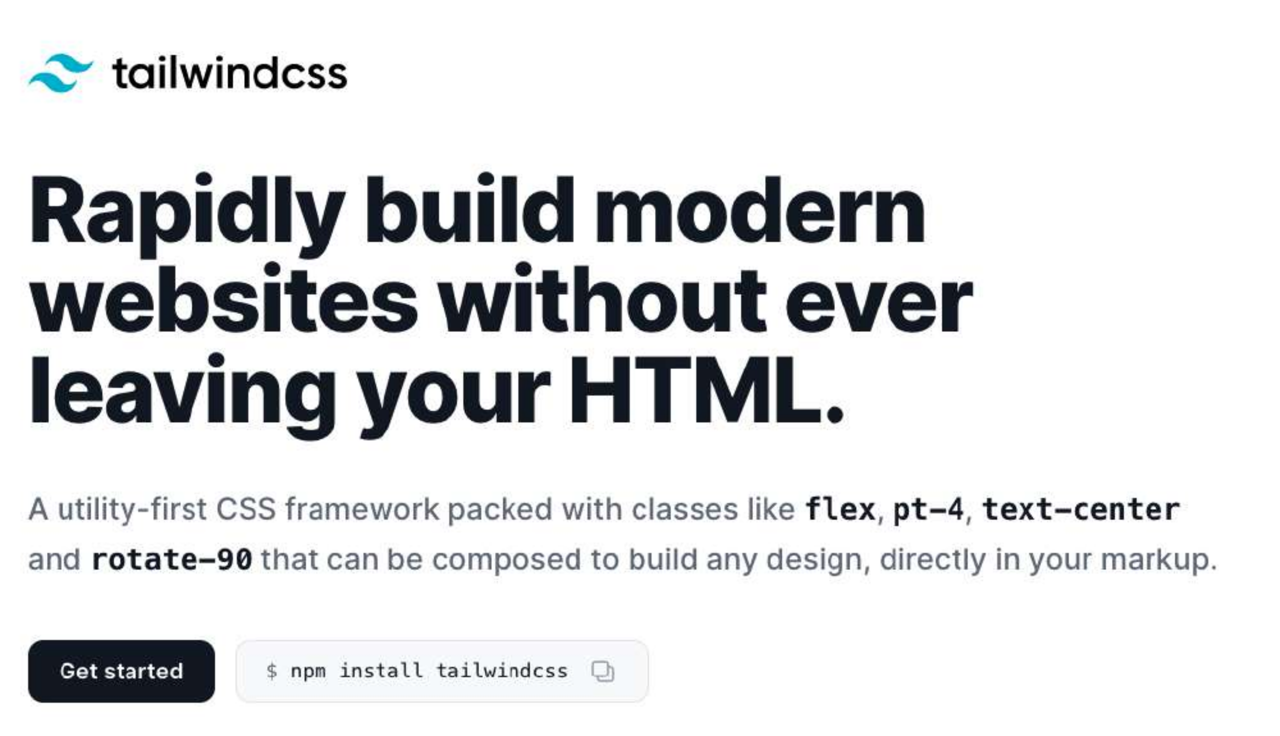
CSS first frameworks which don't come with JavaScript components by default.

Learn different Web APIs used in PWAs

Calculating, Measuring and improving performance

Web Assembly or WASM is the binary instructions generated from higher level languages such as Go, C, C++ or Rust. It is faster than JavaScript and WASM 1.0 has already shipped in the major browsers W3C accepted it as an official standard at the end of 2019. It will still take quite some time to go mainstream though.

Frontend



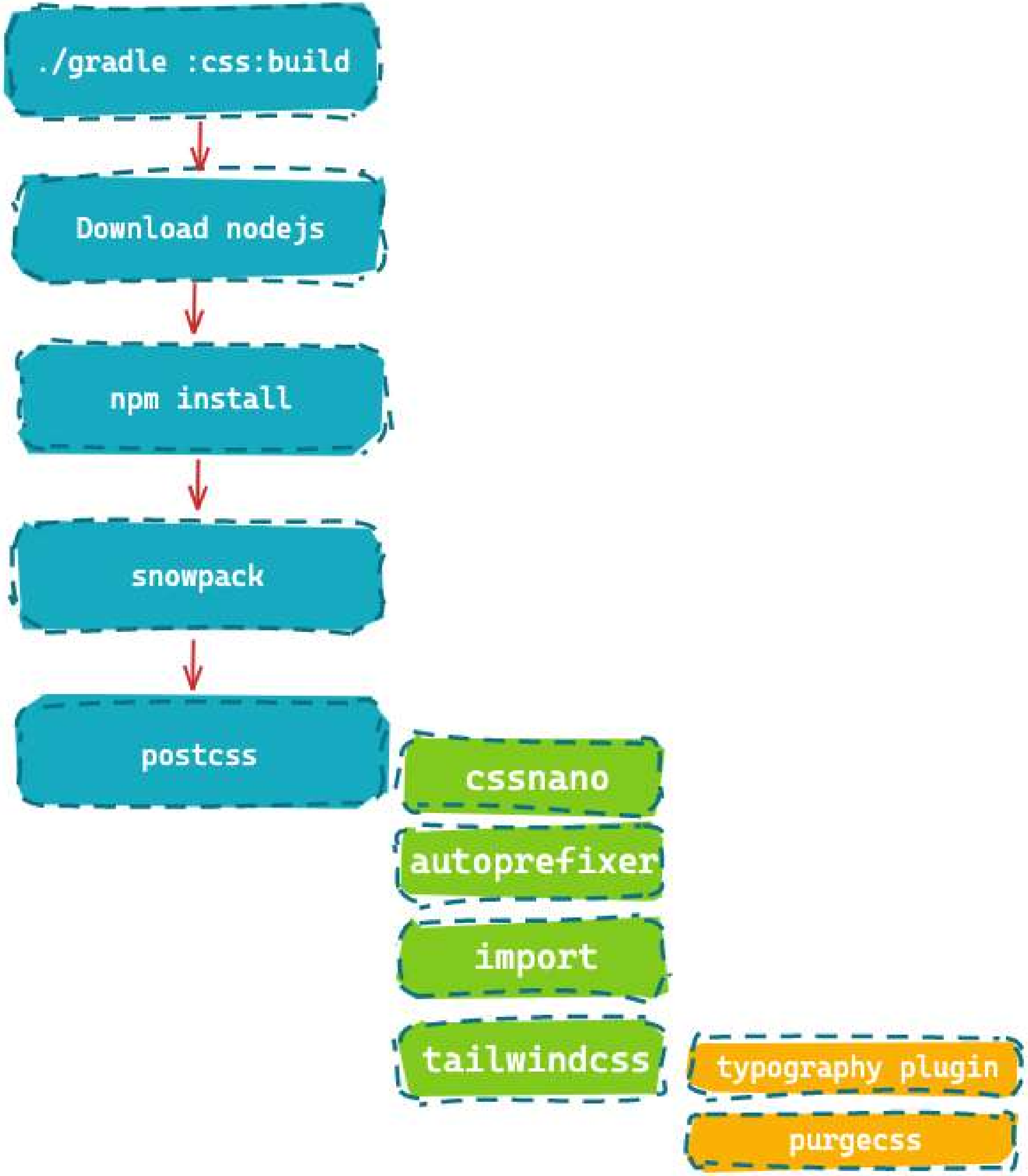
Rapidly build modern websites without ever leaving your HTML.

A utility-first CSS framework packed with classes like **`flex`**, **`pt-4`**, **`text-center`** and **`rotate-90`** that can be composed to build any design, directly in your markup.

```
$ npm install tailwindcss
```

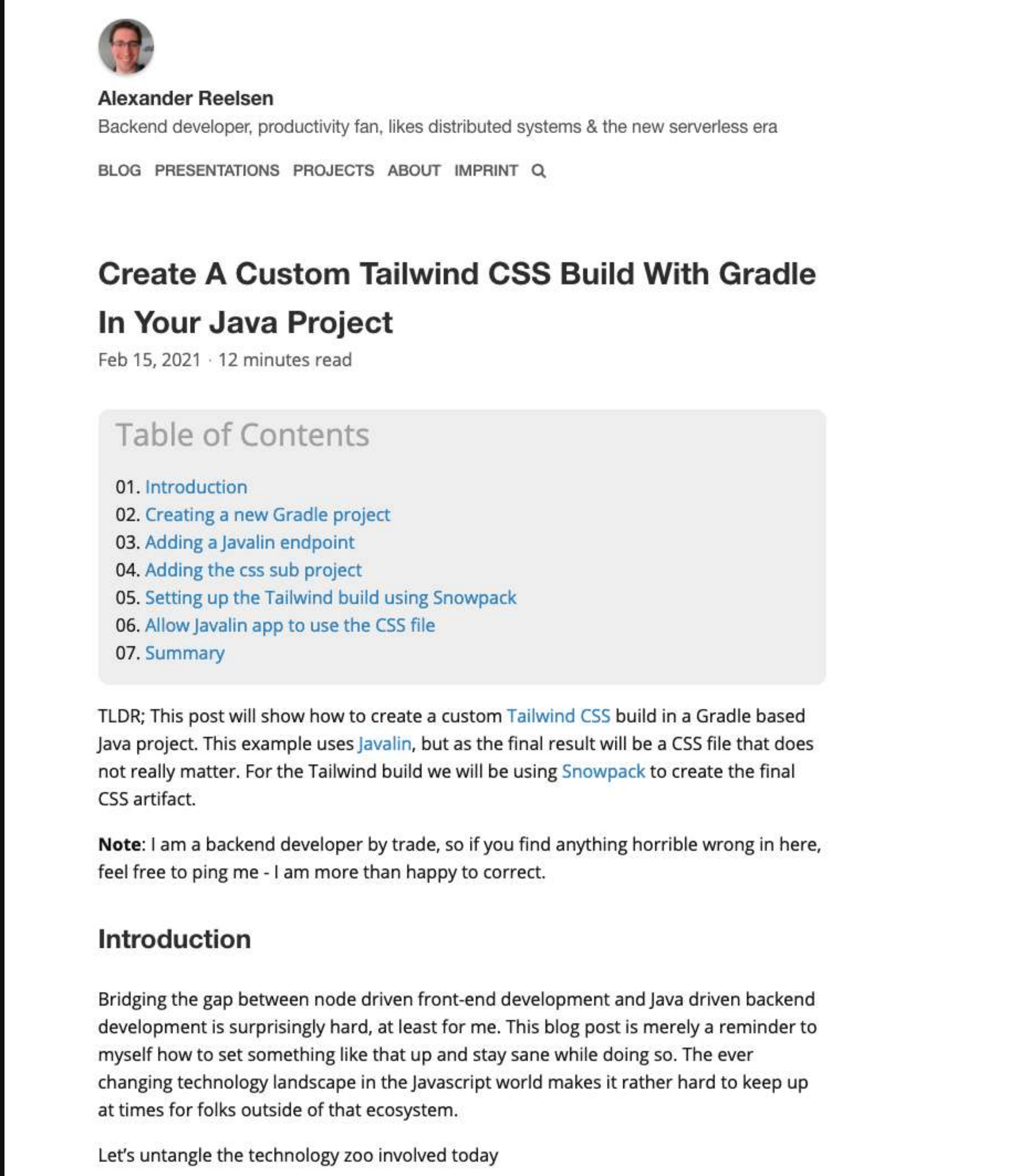
Frontend

Custom Tailwind CSS build



Frontend

Blog post + Sample repo



Alexander Reelsen

Backend developer, productivity fan, likes distributed systems & the new serverless era

Create A Custom Tailwind CSS Build With Gradle In Your Java Project

Feb 15, 2021 · 12 minutes read

Table of Contents

TLDR; This post will show how to create a custom Tailwind CSS build in a Gradle based Java project. This example uses Javalin, but as the final result will be a CSS file that does not really matter. For the Tailwind build we will be using Snowpack to create the final CSS artifact.

Note: I am a backend developer by trade, so if you find anything horrible wrong in here, feel free to ping me - I am more than happy to correct.

Introduction

Bridging the gap between node driven front-end development and Java driven backend development is surprisingly hard, at least for me. This blog post is merely a reminder to myself how to set something like that up and stay sane while doing so. The ever changing technology landscape in the Javascript world makes it rather hard to keep up at times for folks outside of that ecosystem.

Let's untangle the technology zoo involved today

```
plugins {
  id "com.github.node-gradle.node" version "3.0.1"
}

node {
  // current LTS release
  version = "14.16.1"
  download = true
}

// build css and put it into resources directory
task build(type: NpxTask) {
  dependsOn npmInstall
  command = './node_modules/.bin/snowpack'
  args = ['build']
  onlyIf {
    File destFile   = file("$projectDir/build/src/flipbit.css")
    File sourceFile = file("$projectDir/src/flipbit.css")

    if (!destFile.exists()) {
      return true
    }

    return sourceFile.lastModified() > destFile.lastModified()
  }
}

task forceBuild(type: NpxTask) {
  dependsOn npmInstall
  command = './node_modules/.bin/snowpack'
  args = ['build']
}
```

htmx - Ajax without JavaScript

> htmx allows you to access AJAX, CSS Transitions, WebSockets and Server Sent Events directly in HTML, using attributes, so you can build modern user interfaces with the simplicity and power of hypertext

> htmx is small (~10k min.gz'd), dependency-free, extendable & IE11 compatible

htmx in HTML

```
<div class="hidden pt-3" id="resource-form">
  <form hx-post="/app/${organization}/talk/${talk.getId()}/resource" hx-push-url="true" hx-target="#main">
    ...
    <input ... />

    <button type="submit" class="flipbit-button">
      Save
    </button>
  </form>
</div>
```

Server side handling

```
public Handler addResource() {
    return ctx -> {
        Organization organization = ctx.sessionAttribute("organization");
        long id = ctx.pathParam("id", Long.class).get();
        Talk talk = talkDao.findByIdAndOrganization(id, organization);

        String url = URI.create(ctx.formParam("url")).toString();
        ContentResource.ResourceType type =
            ContentResource.ResourceType.valueOf(ctx.formParam("type").toUpperCase(Locale.ROOT));
        final ContentResource contentResource = new ContentResource();
        contentResource.setType(type);
        contentResource.setUrl(url);
        contentResource.setOrganization(organization);
        contentResourceDao.insert(talk.getId(), contentResource);

        ctx.redirect("/app/" + organization.getName() + "/talk/" + talk.getId());
    };
}
```

Rollout

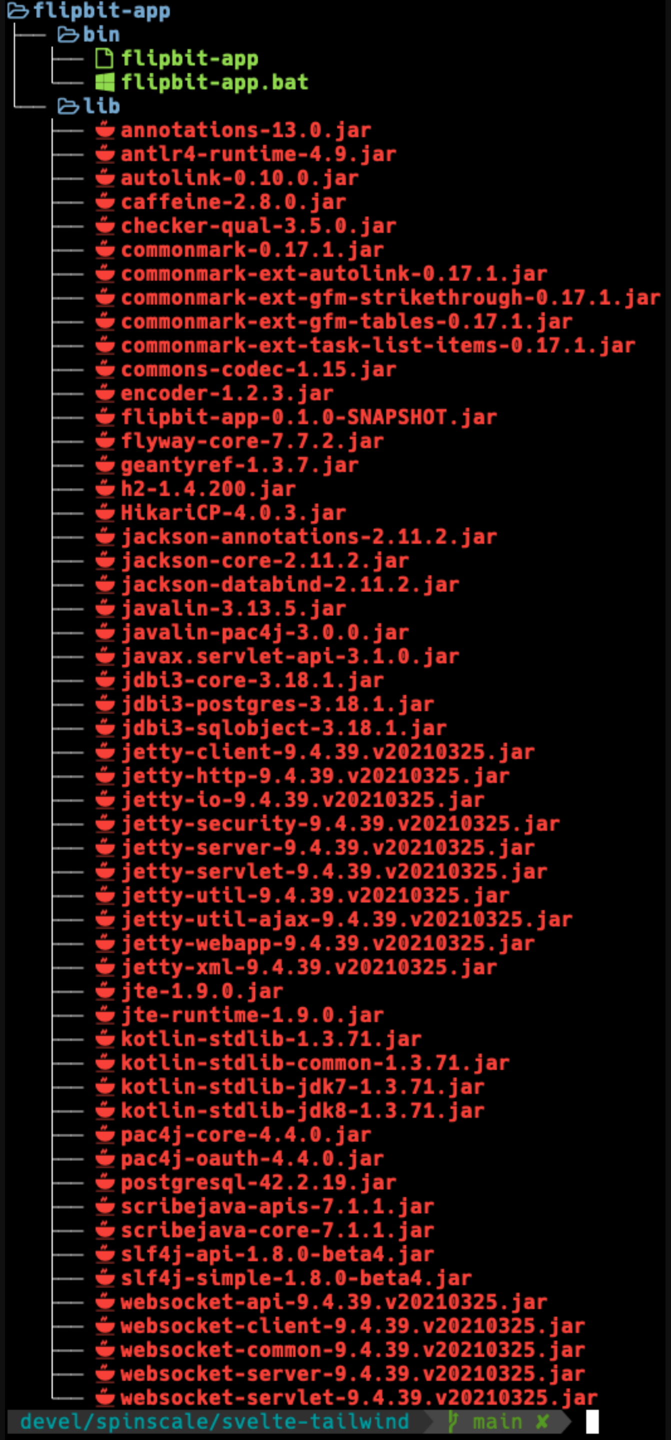
- `git push spinscale main`
- Digital Ocean App Platform
- Postgresql
- Costs: 27$ per month (different kind of gym costs 😀)
- One click scale up

Rollout alternatives

- Own VM: Price 💰| Maintenance 👎 | Setup (TLS) 👎 | Custom rollout 👎
- Heroku: 💰💰💰
- Qovery 💰💰 👍
- Digital Ocean App Platform 💰💰 👍
- Others: fly.io, AWS Beanstalk
- Java makes options more expensive

Start scripts

- gradle `application` plugin
- Creates shell scripts for services automatically via `installDist` task
- `bin/flipbit-app`
- Package size: less than 15MB



```
flipbit-app
├── bin
│   ├── flipbit-app
│   └── flipbit-app.bat
└── lib
    ├── annotations-13.0.jar
    ├── antlr4-runtime-4.9.jar
    ├── autolink-0.10.0.jar
    ├── caffeine-2.8.0.jar
    ├── checker-qual-3.5.0.jar
    ├── commonmark-0.17.1.jar
    ├── commonmark-ext-autolink-0.17.1.jar
    ├── commonmark-ext-gfm-strikethrough-0.17.1.jar
    ├── commonmark-ext-gfm-tables-0.17.1.jar
    ├── commonmark-ext-task-list-items-0.17.1.jar
    ├── commons-codec-1.15.jar
    ├── encoder-1.2.3.jar
    ├── flipbit-app-0.1.0-SNAPSHOT.jar
    ├── flyway-core-7.7.2.jar
    ├── geantyref-1.3.7.jar
    ├── h2-1.4.200.jar
    ├── HikariCP-4.0.3.jar
    ├── jackson-annotations-2.11.2.jar
    ├── jackson-core-2.11.2.jar
    ├── jackson-databind-2.11.2.jar
    ├── javalin-3.13.5.jar
    ├── javalin-pac4j-3.0.0.jar
    ├── javax.servlet-api-3.1.0.jar
    ├── jdbi3-core-3.18.1.jar
    ├── jdbi3-postgres-3.18.1.jar
    ├── jdbi3-sqlobject-3.18.1.jar
    ├── jetty-client-9.4.39.v20210325.jar
    ├── jetty-http-9.4.39.v20210325.jar
    ├── jetty-io-9.4.39.v20210325.jar
    ├── jetty-security-9.4.39.v20210325.jar
    ├── jetty-server-9.4.39.v20210325.jar
    ├── jetty-servlet-9.4.39.v20210325.jar
    ├── jetty-util-9.4.39.v20210325.jar
    ├── jetty-util-ajax-9.4.39.v20210325.jar
    ├── jetty-webapp-9.4.39.v20210325.jar
    ├── jetty-xml-9.4.39.v20210325.jar
    ├── jte-1.9.0.jar
    ├── jte-runtime-1.9.0.jar
    ├── kotlin-stdlib-1.3.71.jar
    ├── kotlin-stdlib-common-1.3.71.jar
    ├── kotlin-stdlib-jdk7-1.3.71.jar
    ├── kotlin-stdlib-jdk8-1.3.71.jar
    ├── pac4j-core-4.4.0.jar
    ├── pac4j-oauth-4.4.0.jar
    ├── postgresql-42.2.19.jar
    ├── scribejava-apis-7.1.1.jar
    ├── scribejava-core-7.1.1.jar
    ├── slf4j-api-1.8.0-beta4.jar
    ├── slf4j-simple-1.8.0-beta4.jar
    ├── websocket-api-9.4.39.v20210325.jar
    ├── websocket-client-9.4.39.v20210325.jar
    ├── websocket-common-9.4.39.v20210325.jar
    ├── websocket-server-9.4.39.v20210325.jar
    └── websocket-servlet-9.4.39.v20210325.jar
devel/spinscale/svelte-tailwind  main ✗
```

Use ZGC

```
def jvmOptions = ["-Xmx512m", "-XX:+UseZGC"]

startScripts {
  defaultJvmOpts = jvmOptions
}

run {
  dependsOn('copyCssFile')
  jvmArgs += jvmOptions
}
```

Docker image creation

- App Platform is read-only: Precompiled templates
- Shell script creation!

Dockerfile

```
FROM gradle:7.0-jdk16 as build
COPY --chown=gradle:gradle . /home/gradle/src
WORKDIR /home/gradle/src
RUN gradle clean check precompileJte installDist --no-daemon

###############
FROM openjdk:16-slim-buster
RUN addgroup --system flipbit && adduser --system flipbit --ingroup flipbit
USER flipbit:flipbit

WORKDIR /flipbit-app
COPY --from=build /home/gradle/src/build/ /flipbit-app
COPY --from=build /home/gradle/src/jte-classes/ /jte-classes

EXPOSE 7000
ENTRYPOINT [ "/flipbit-app/install/flipbit-app/bin/flipbit-app" ]
```

`Dockerfile` - agent support

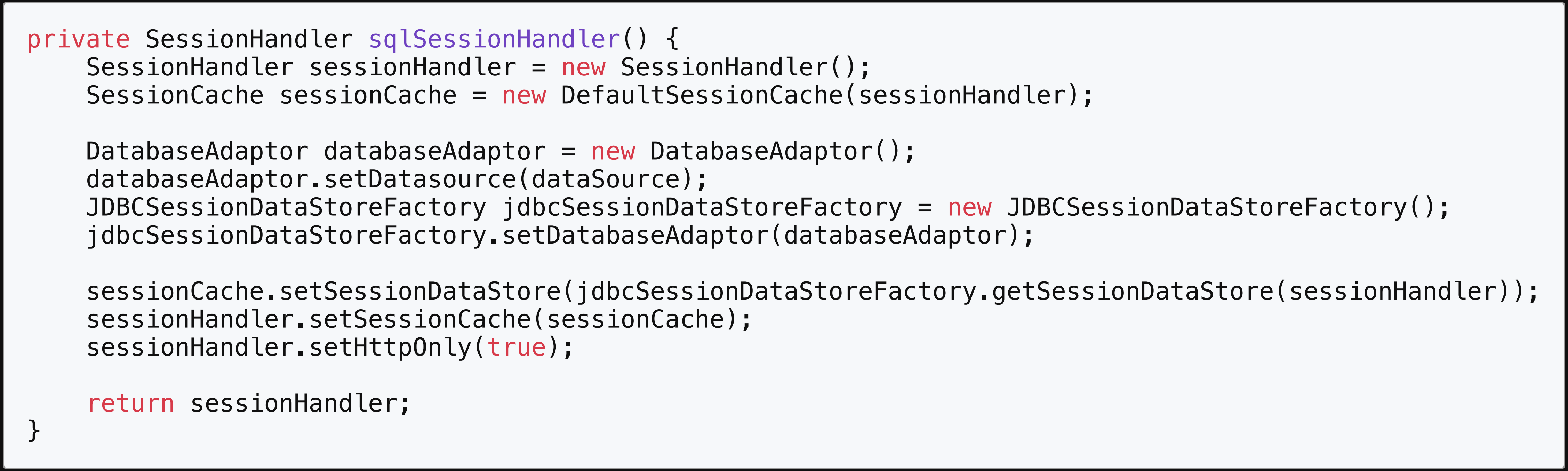
```
// download the lightstep agent as part of our bundle
// so we can have autoinstrumentation
def agentVersion = "0.16.0"
task downloadAgent(type: Download) {
  mkdir ".gradle/agent/"
  src "https://github.com/lightstep/otel-launcher-java/releases/download/${agentVersion}/lightstep-opentelemetry-ja
vaagent-${agentVersion}.jar"  dest new File(".gradle/agent", "lightstep-opentelemetry-agent-${agentVersion}.jar")
  onlyIfModified true
  overwrite false
}

task copyAgent(type: Copy) {
  dependsOn 'downloadAgent'
  from(".gradle/agent/lightstep-opentelemetry-agent-${agentVersion}.jar") {
    rename "(.*)-${agentVersion}.jar", '$1.jar'
  }
  into "build/install/flipbit-app/agent/"
}
copyAgent.mustRunAfter installDist
```

- Download & copy agent via gradle
- Add env vars: `LS_SERVICE_NAME` & `FLIPBIT_APP_OPTS`

Persistent sessions across restarts

```
this.app = Javalin.create(config -> {
    ...
    config.sessionHandler(this::sqlSessionHandler);
});
```

Persistent sessions across restarts



```java
private SessionHandler sqlSessionHandler() {
    SessionHandler sessionHandler = new SessionHandler();
    SessionCache sessionCache = new DefaultSessionCache(sessionHandler);

    DatabaseAdaptor databaseAdaptor = new DatabaseAdaptor();
    databaseAdaptor.setDatasource(dataSource);
    JDBCSessionDataStoreFactory jdbcSessionDataStoreFactory = new JDBCSessionDataStoreFactory();
    jdbcSessionDataStoreFactory.setDatabaseAdaptor(databaseAdaptor);

    sessionCache.setSessionDataStore(jdbcSessionDataStoreFactory.getSessionDataStore(sessionHandler));
    sessionHandler.setSessionCache(sessionCache);
    sessionHandler.setHttpOnly(true);

    return sessionHandler;
}
```

The missing parts

> So much... all the time

- Development & Testing
- Build
- Metrics
- Ops
- Security
- UI

Missing: Development & Testing

- Build high level classes around low level DAOs
- E2E-Testing
- Don't run flyway for all unit tests, don't recreate h2 db for every test class
- Unique incrementing ids per organization
- Easy local development: offline support (no oauth)
- H2/Postgres gap

Build

- GraalVM all the things - reduces monitoring
- Reproducible builds

Missing: Metrics

- Business metrics/health endpoints
- Analytics (not needed with web logs?)
- Monitoring/APM

Missing: Ops

- Terraform setup for DO
- Rate Limiting (Javalin built-in?)
- Terraform based monitoring (for example Checkly)
- APM: Every agent supports Spring Boot and then nothing...

Missing: Security

- Add all the fancy security headers, CORS
  - `X-XXS-Protection`, `X-Frame-Options`, Content-Security-Policy, `X-Content-Type-Options`. See also OWASP Secure Header Project
- Add seccomp policy to never fork processes
- Become smarter about landlock as a java security manager replacement

Missing: UI

- Rework UI to make it acceptable

Summary

- Don't under estimate the work
- There is no failure!
- **Keep it fun!**
- Combining different libraries means a lot of learning
- Also, who wants to test? :-)
- `PaaS` > `K8s`

Thanks for listening

Q & A

Alexander Reelsen
alr@spinscale.de | @spinscale

Discussion

- `What technologies would you use?`
- `Where did I go wrong?`

Thanks for listening

Q & A

Alexander Reelsen

alr@spinscale.de | @spinscale