A GENTLE INTRODUCTION TO

DROPWIZARD

Ran Tavory

@rantav

Gormim

Totango

ABOUT ME

- Developer developer developer
- Gormim
  - We help developers find awesome jobs.
  - You should register! www.gormim.com
- Totango
- Previously: outbrain, google, youtube, microsoft

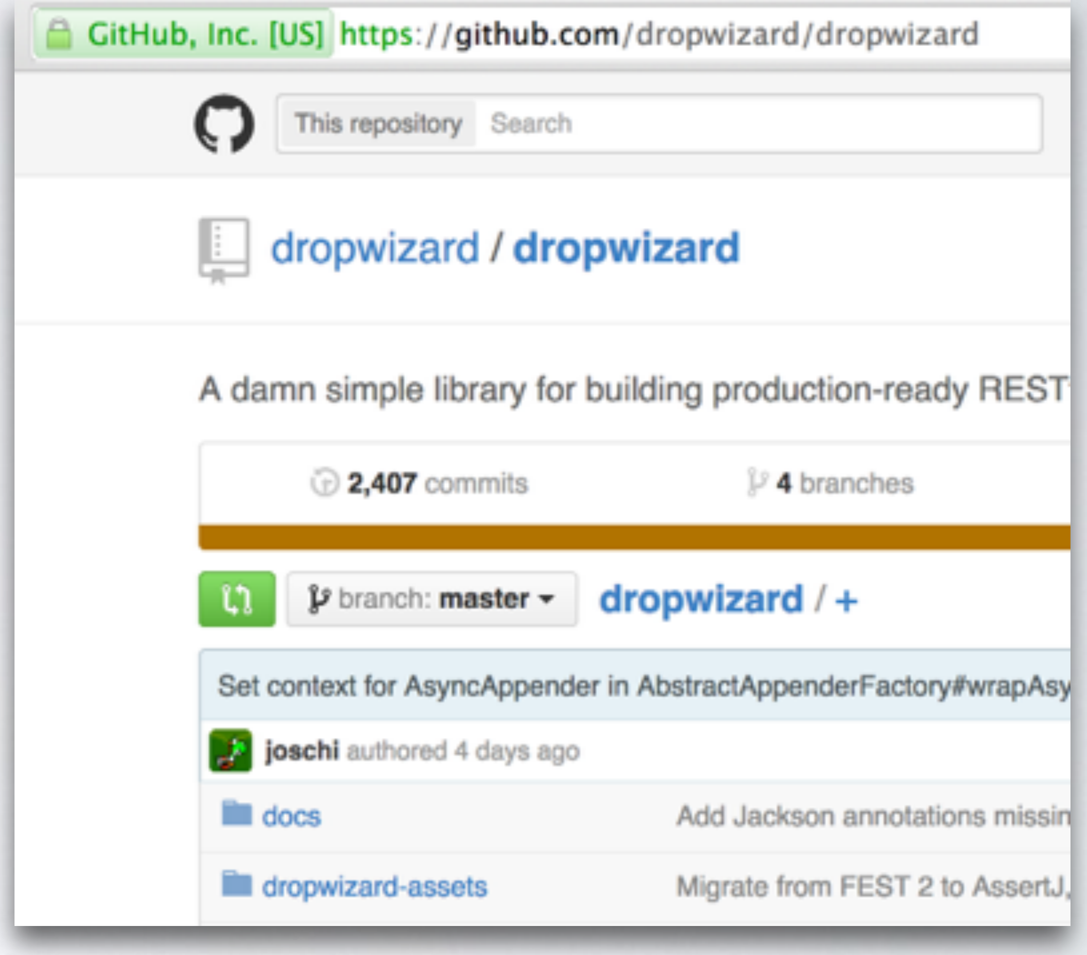
MY EXPERIENCE WITH DROPWIZARD

- So far implemented 3 microsevices with DW

- At Totango

WHAT IS DROPWIZARD?

- A collection of **libraries**
- And some **glue** code
- Goal:
  - Make it easy to write (micro)services, in Java



GitHub, Inc. [US] https://github.com/dropwizard/dropwizard

dropwizard / **dropwizard**

A damn simple library for building production-ready REST

2,407 commits **4** branches

dropwizard / +

Set context for AsyncAppender in AbstractAppenderFactory#wrapAsy

joschi authored 4 days ago

docs	Add Jackson annotations missin
dropwizard-assets	Migrate from FEST 2 to AssertJ,

MICROSERVICES, YEAH

- Goal: **easy** to create new services
- Goal: **simple** to use
- Goal: **easy** to deploy
- Goal: **easy** to monitor, manage
    - devops **friendly**

DROPWIZARD IS OPINIONATED*

- **Jetty** for HTTP
- **Jersey** for REST (JAX-RS)
- **Jackson** for JSON
- **Metrics** for metrics
- **SLF4J**, **Logback** for logging

… IS OPINIONATED*

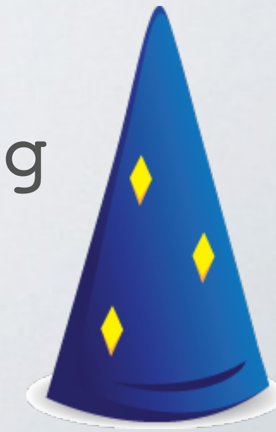
- **Guava** b/c it's useful
- **Hibernate Validator** (JSR-303) for data validation
- **Apache HTTP Client** for low level HTTP
- **Jersey HTTP** client for high level HTTP
- **jDBI** for relational databases
- **Liquibase** for DB schema maintenance

… IS OPINIONATED*

- **Freemarker** and **Mustach** for templating
- **Joda Time**

*SOME **MODULARITY** IN PLACE

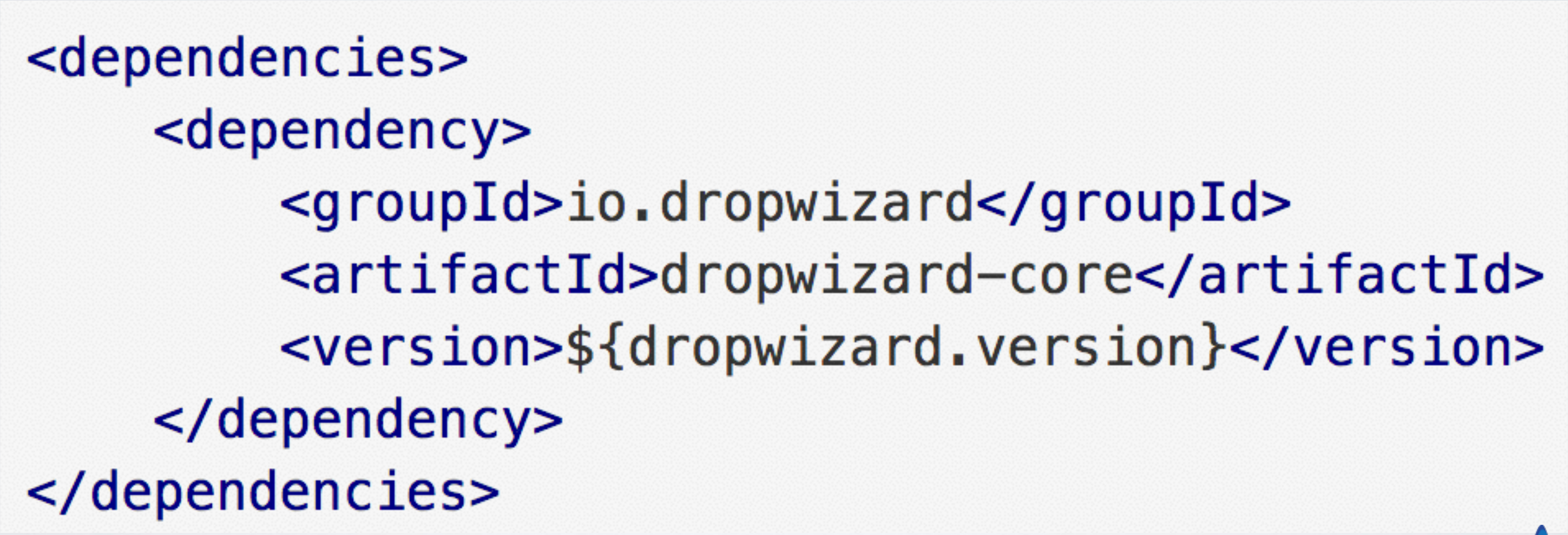
- Maven artifacts:
  - **`dropwizard-core`**
  - **`dropwizard-client`** (http client)
  - **`dropwizard-jdbi`**
  - **`dropwizard-migrations`** (liquibase)
  - **`dropwizard-hibernate`**
  - **`dropwizard-auth`** (HTTP basic and oauth2)
  - **`dropwizard-views-mustache`** and **`dropwizard-views-freemarker`**
  - **`dropwizard-scala`**
  - **`dropwizard-testing`**



GETTING STARTED

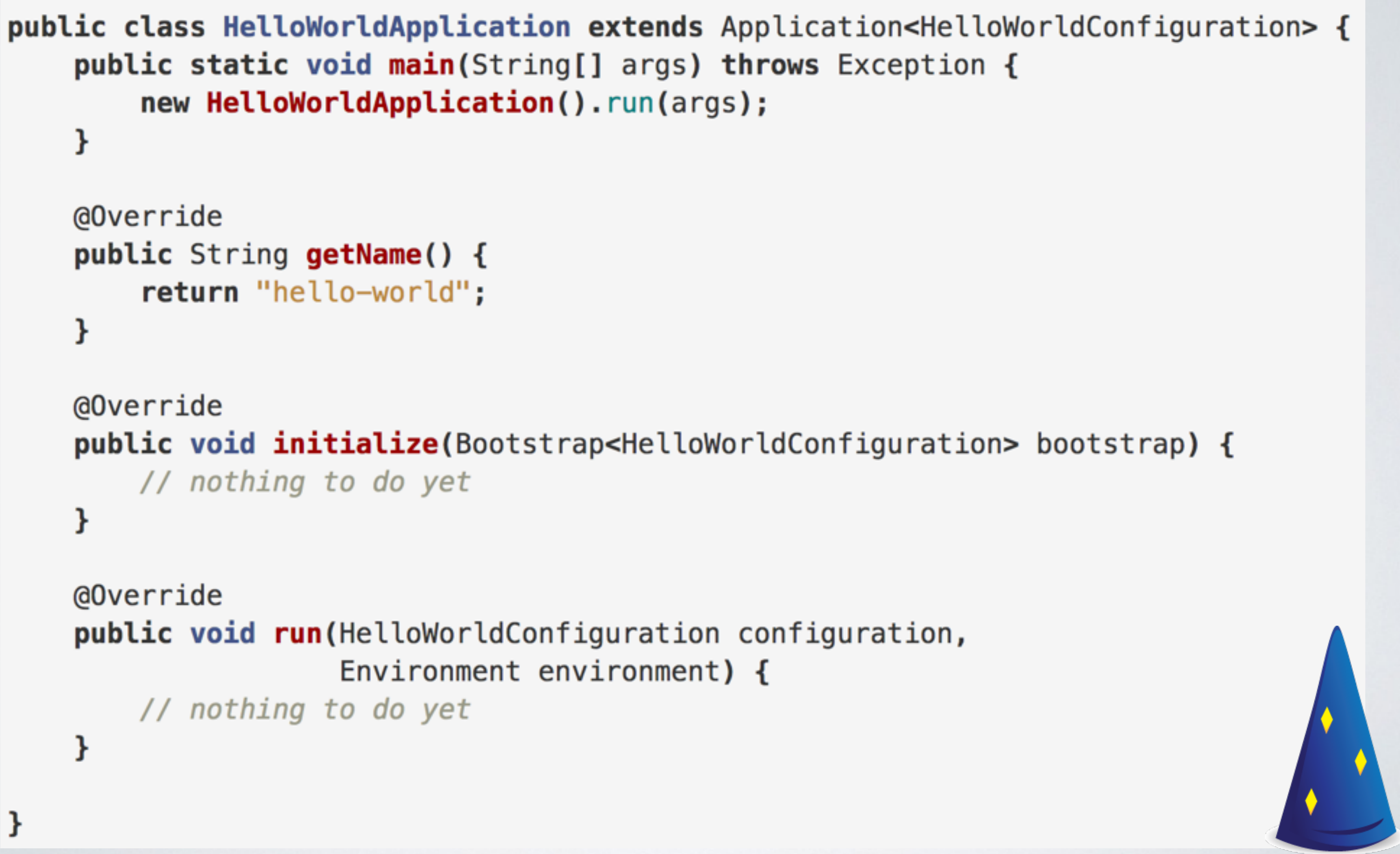
- Maven
- Application class
- Configuration class
- Representation class (optional)
- Resource class (optional)
- Healthcheck

MAVEN



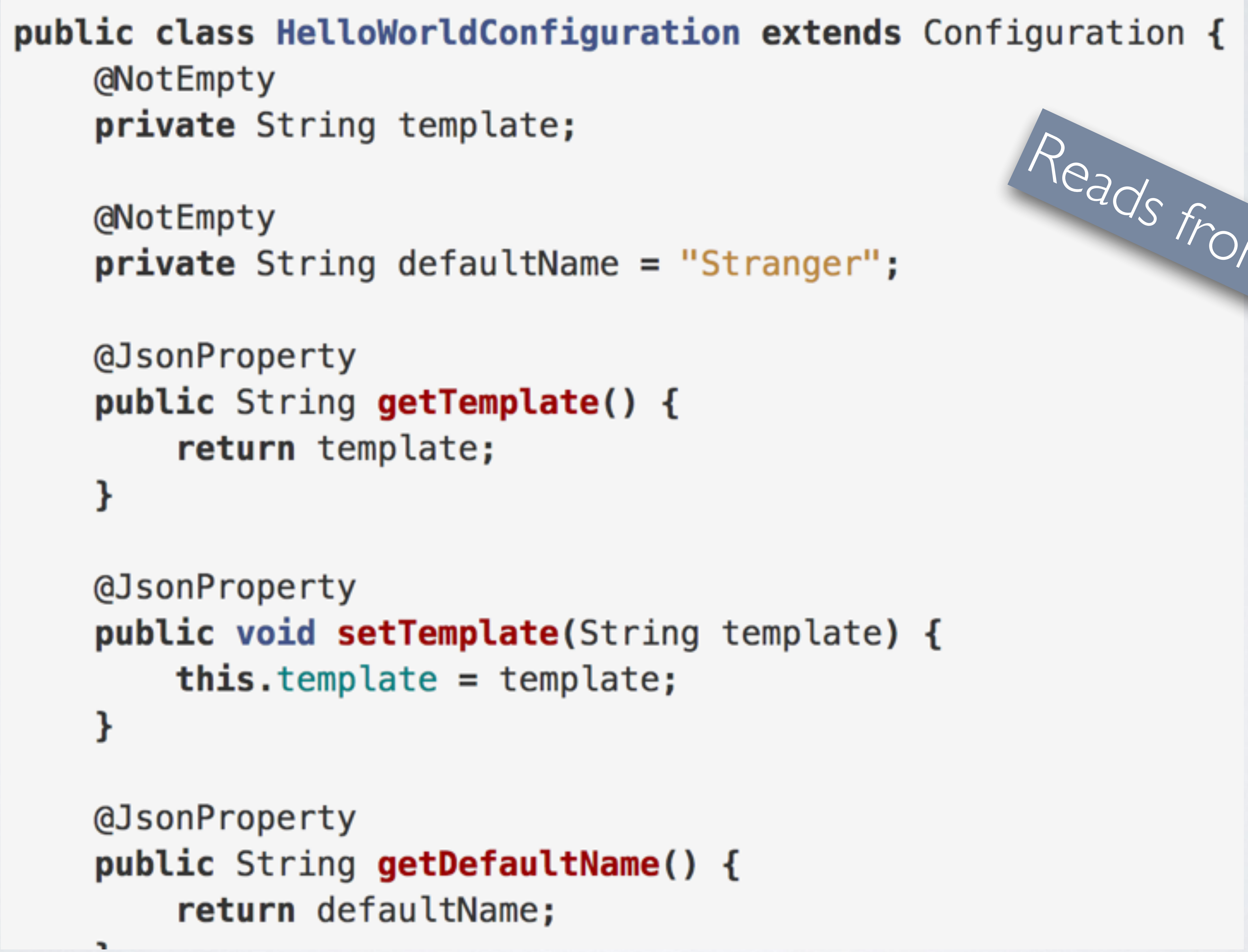
```
<dependencies>
    <dependency>
        <groupId>io.dropwizard</groupId>
        <artifactId>dropwizard-core</artifactId>
        <version>${dropwizard.version}</version>
    </dependency>
</dependencies>
```

APPLICATION CLASS



```java
public class HelloWorldApplication extends Application<HelloWorldConfiguration> {
    public static void main(String[] args) throws Exception {
        new HelloWorldApplication().run(args);
    }

    @Override
    public String getName() {
        return "hello-world";
    }

    @Override
    public void initialize(Bootstrap<HelloWorldConfiguration> bootstrap) {
        // nothing to do yet
    }

    @Override
    public void run(HelloWorldConfiguration configuration,
                    Environment environment) {
        // nothing to do yet
    }

}
```

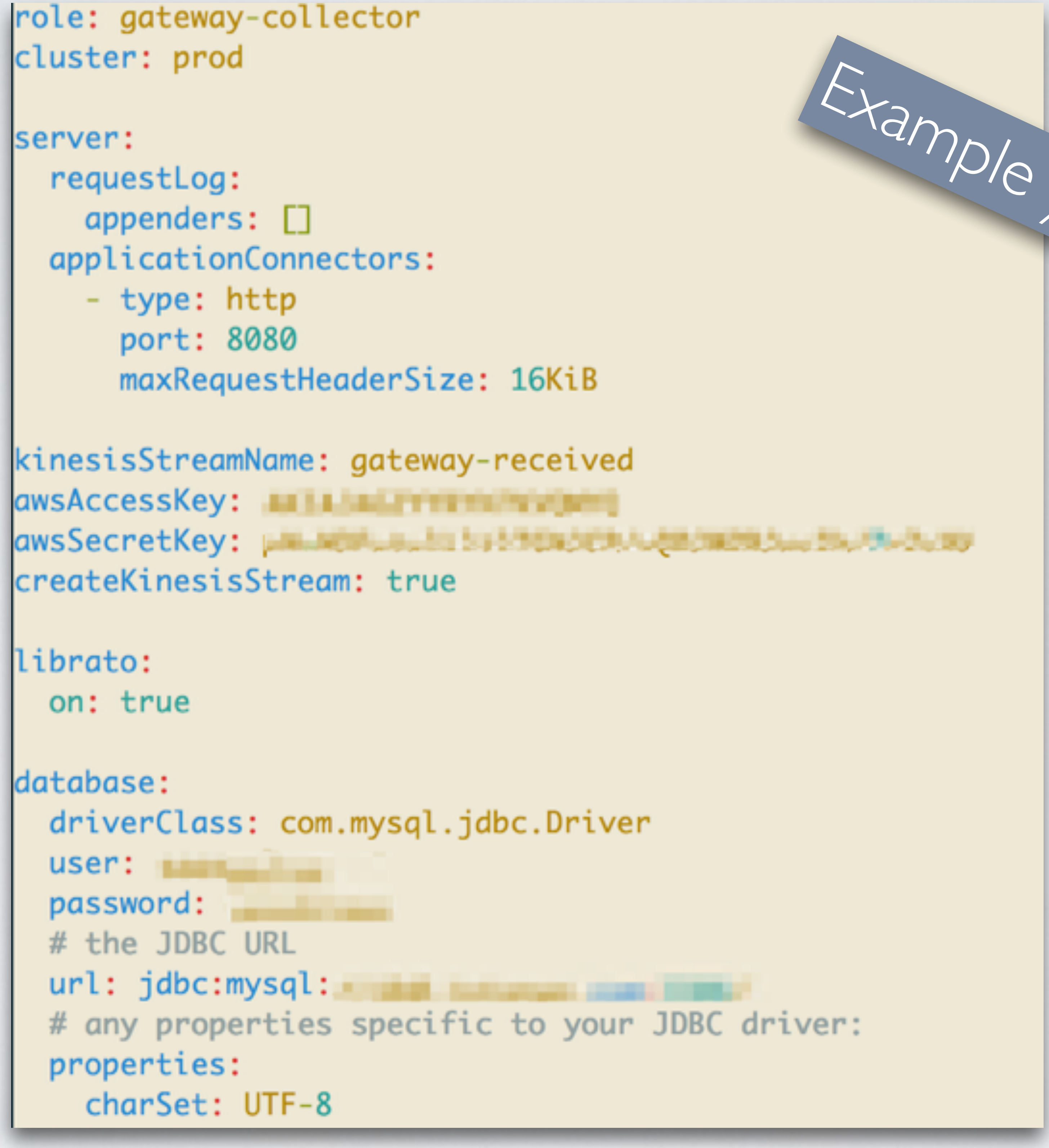
CONFIGURATION CLASS



```java
public class HelloWorldConfiguration extends Configuration {
    @NotEmpty
    private String template;

    @NotEmpty
    private String defaultName = "Stranger";

    @JsonProperty
    public String getTemplate() {
        return template;
    }

    @JsonProperty
    public void setTemplate(String template) {
        this.template = template;
    }

    @JsonProperty
    public String getDefaultName() {
        return defaultName;
    }
```

Reads from yml file

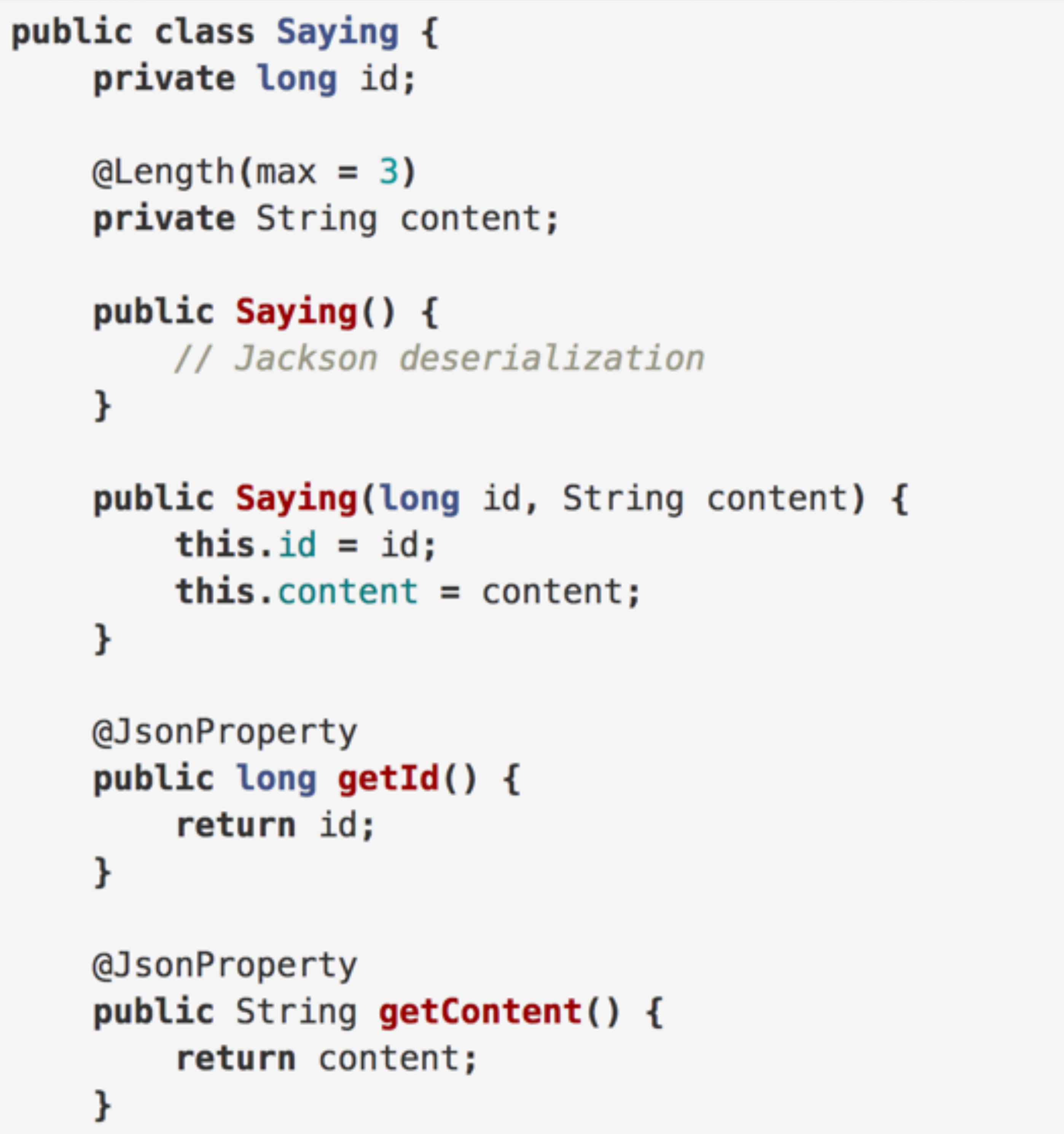
CONFIGURATION FILE



```
role: gateway-collector
cluster: prod

server:
  requestLog:
    appenders: []
  applicationConnectors:
    - type: http
      port: 8080
      maxRequestHeaderSize: 16KiB

kinesisStreamName: gateway-received
awsAccessKey: [redacted]
awsSecretKey: [redacted]
createKinesisStream: true

librato:
  on: true

database:
  driverClass: com.mysql.jdbc.Driver
  user: [redacted]
  password: [redacted]
  # the JDBC URL
  url: jdbc:mysql:[redacted]
  # any properties specific to your JDBC driver:
  properties:
    charSet: UTF-8
```

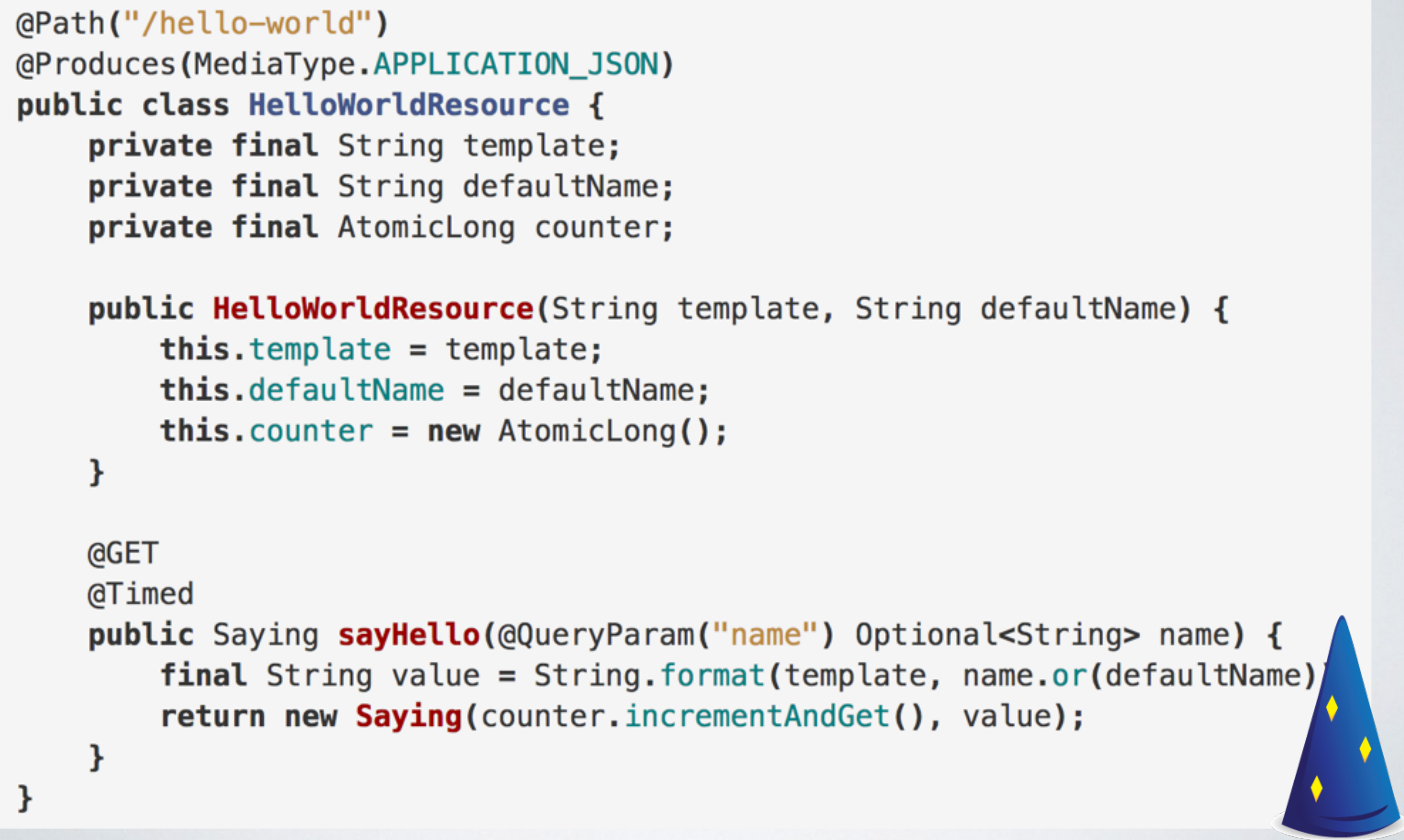
Example yml file

REPRESENTATION CLASS



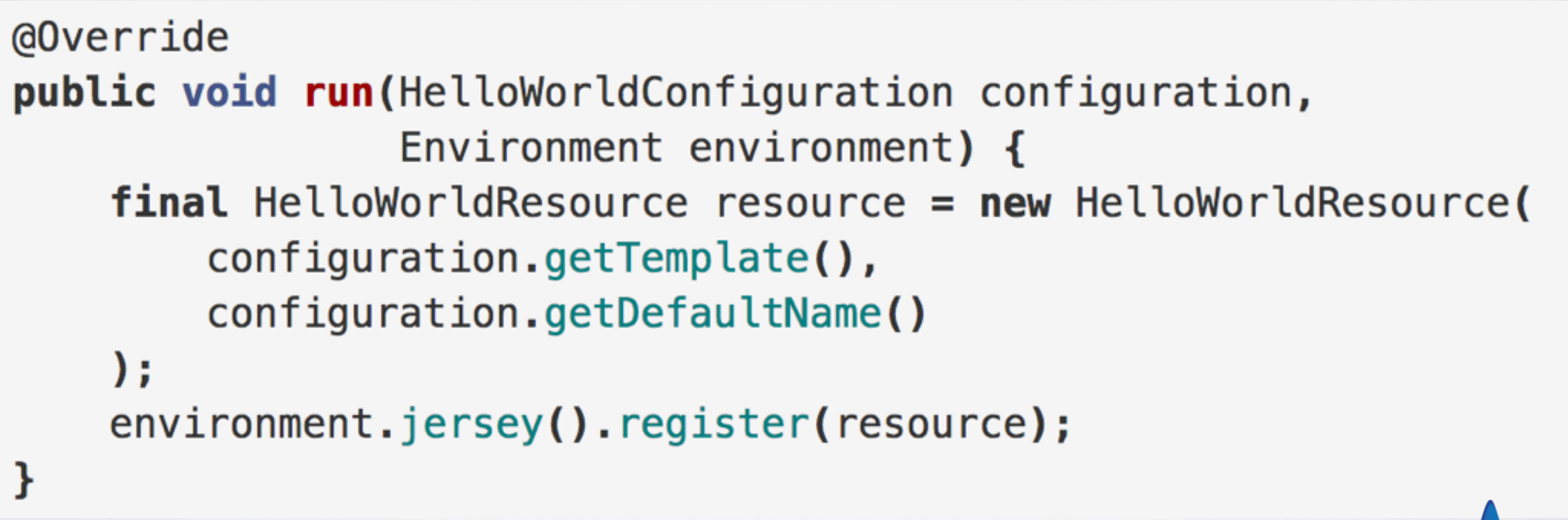
```java
public class Saying {
    private long id;

    @Length(max = 3)
    private String content;

    public Saying() {
        // Jackson deserialization
    }

    public Saying(long id, String content) {
        this.id = id;
        this.content = content;
    }

    @JsonProperty
    public long getId() {
        return id;
    }

    @JsonProperty
    public String getContent() {
        return content;
    }
```

RESOURCE CLASS



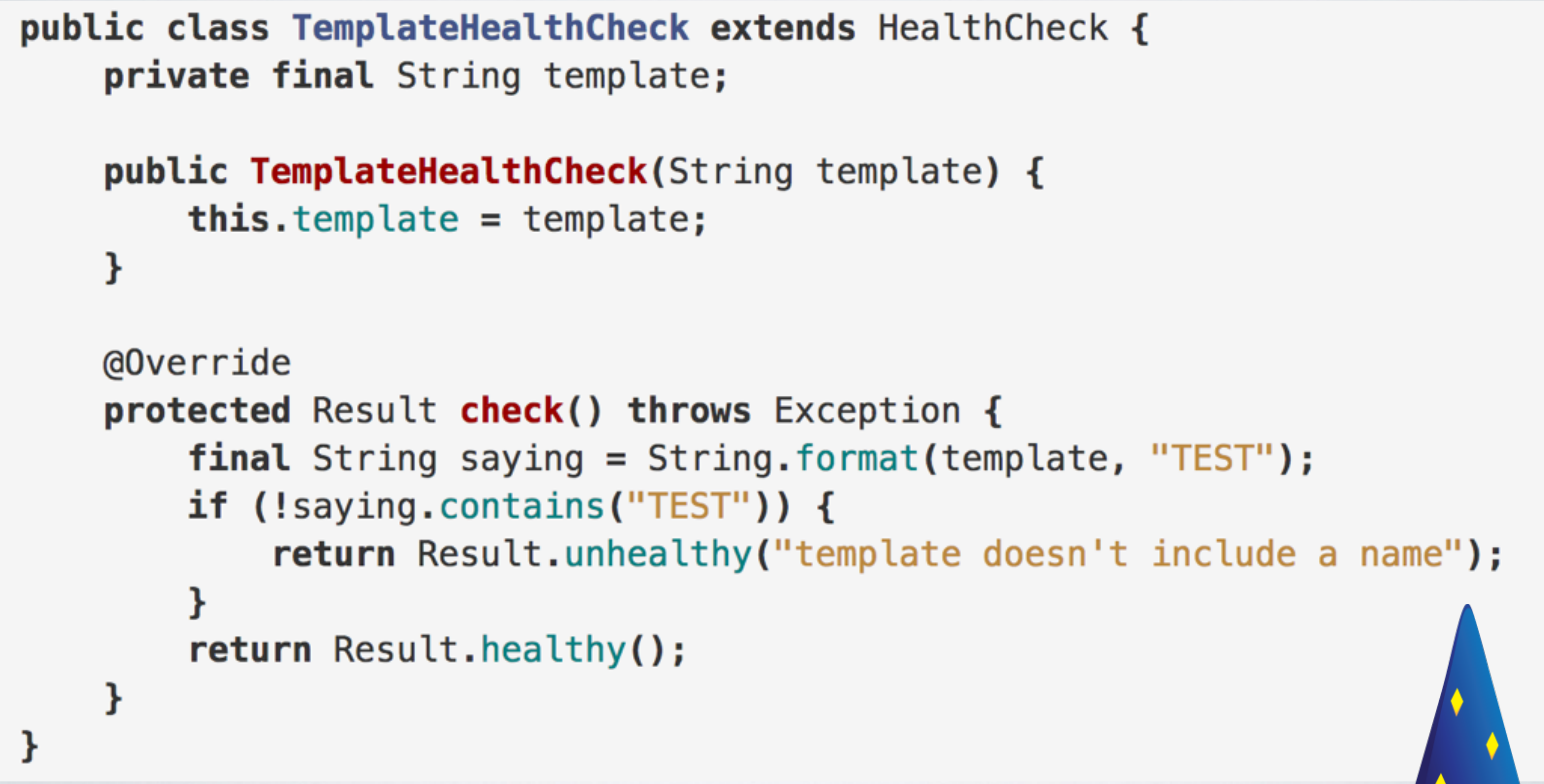
```java
@Path("/hello-world")
@Produces(MediaType.APPLICATION_JSON)
public class HelloWorldResource {
    private final String template;
    private final String defaultName;
    private final AtomicLong counter;

    public HelloWorldResource(String template, String defaultName) {
        this.template = template;
        this.defaultName = defaultName;
        this.counter = new AtomicLong();
    }

    @GET
    @Timed
    public Saying sayHello(@QueryParam("name") Optional<String> name) {
        final String value = String.format(template, name.or(defaultName)
        return new Saying(counter.incrementAndGet(), value);
    }
}
```

REGISTER THE RESOURCE



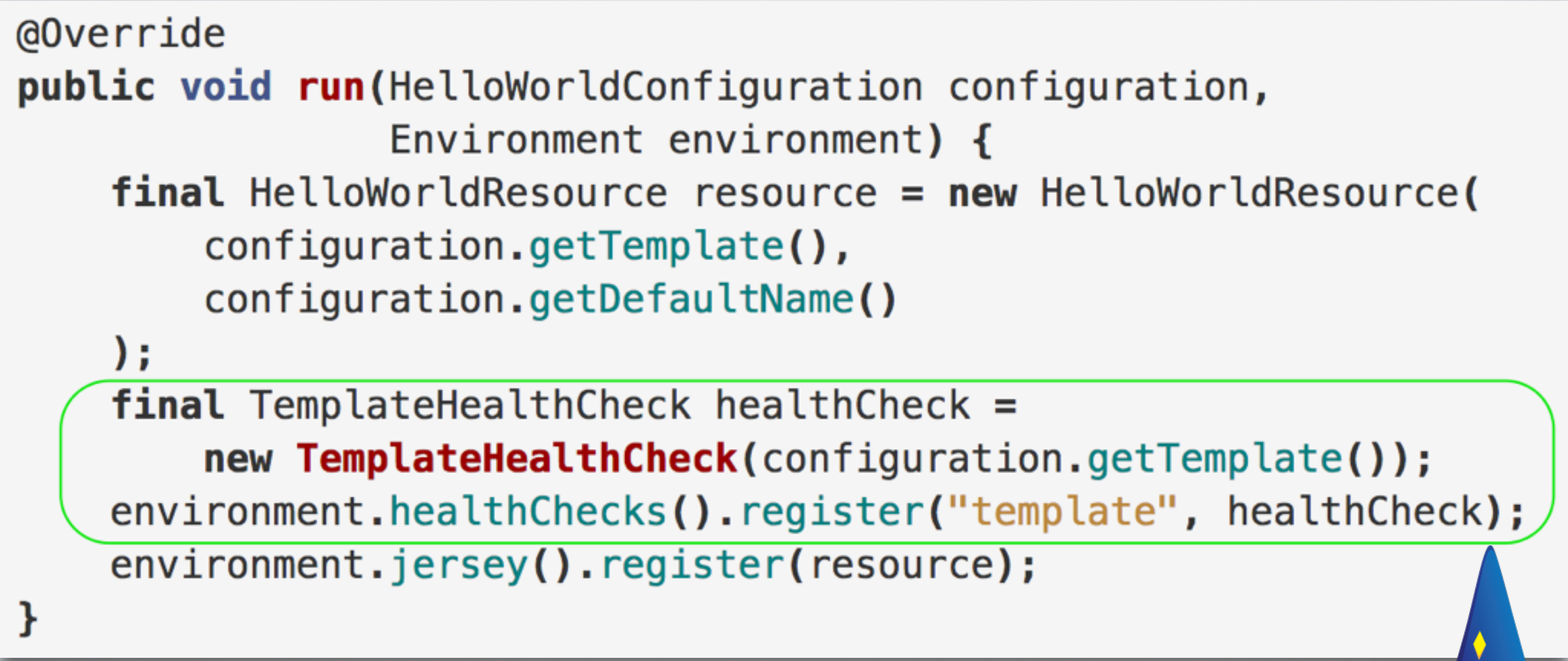
```java
@Override
public void run(HelloWorldConfiguration configuration,
                Environment environment) {
    final HelloWorldResource resource = new HelloWorldResource(
        configuration.getTemplate(),
        configuration.getDefaultName()
    );
    environment.jersey().register(resource);
}
```

HEALTHCHECK



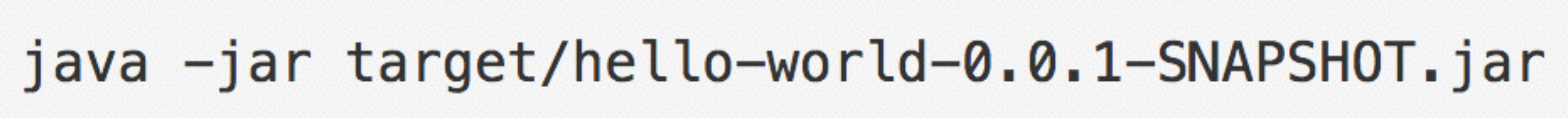
```
public class TemplateHealthCheck extends HealthCheck {
    private final String template;

    public TemplateHealthCheck(String template) {
        this.template = template;
    }

    @Override
    protected Result check() throws Exception {
        final String saying = String.format(template, "TEST");
        if (!saying.contains("TEST")) {
            return Result.unhealthy("template doesn't include a name");
        }
        return Result.healthy();
    }
}
```

HEALTHCHECK



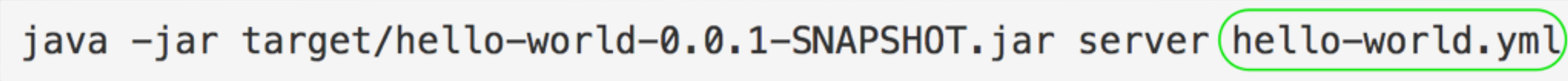
```
@Override
public void run(HelloWorldConfiguration configuration,
                Environment environment) {
    final HelloWorldResource resource = new HelloWorldResource(
        configuration.getTemplate(),
        configuration.getDefaultName()
    );
    final TemplateHealthCheck healthCheck =
        new TemplateHealthCheck(configuration.getTemplate());
    environment.healthChecks().register("template", healthCheck);
    environment.jersey().register(resource);
}
```

RUNNING

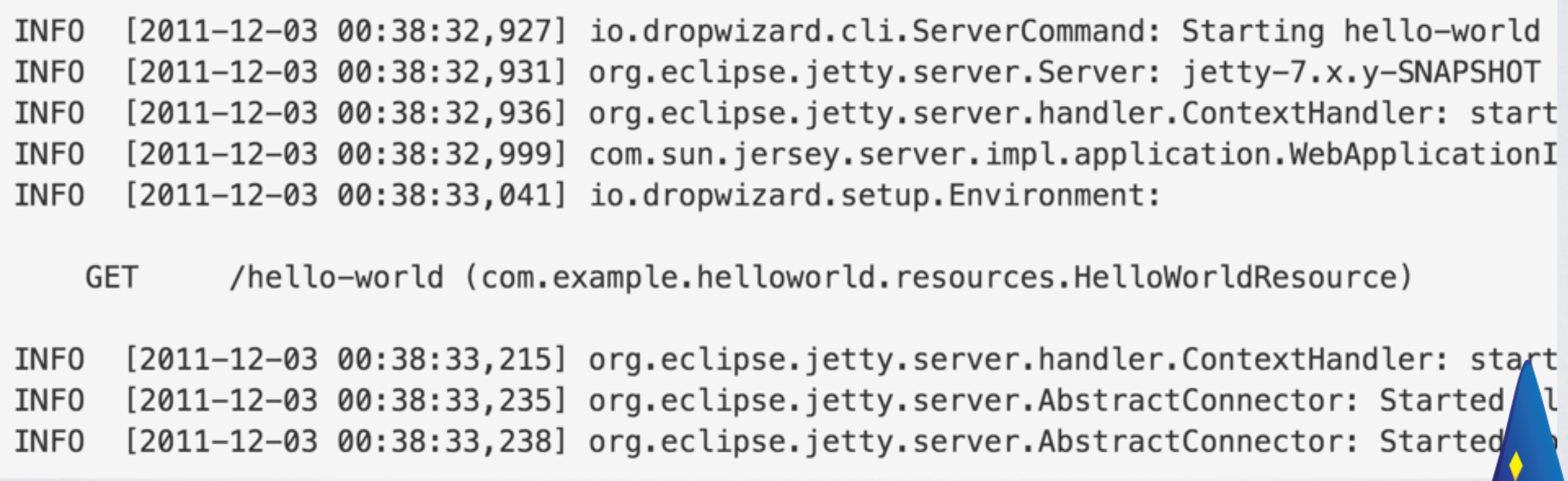


```
java -jar target/hello-world-0.0.1-SNAPSHOT.jar
```

RUNNING

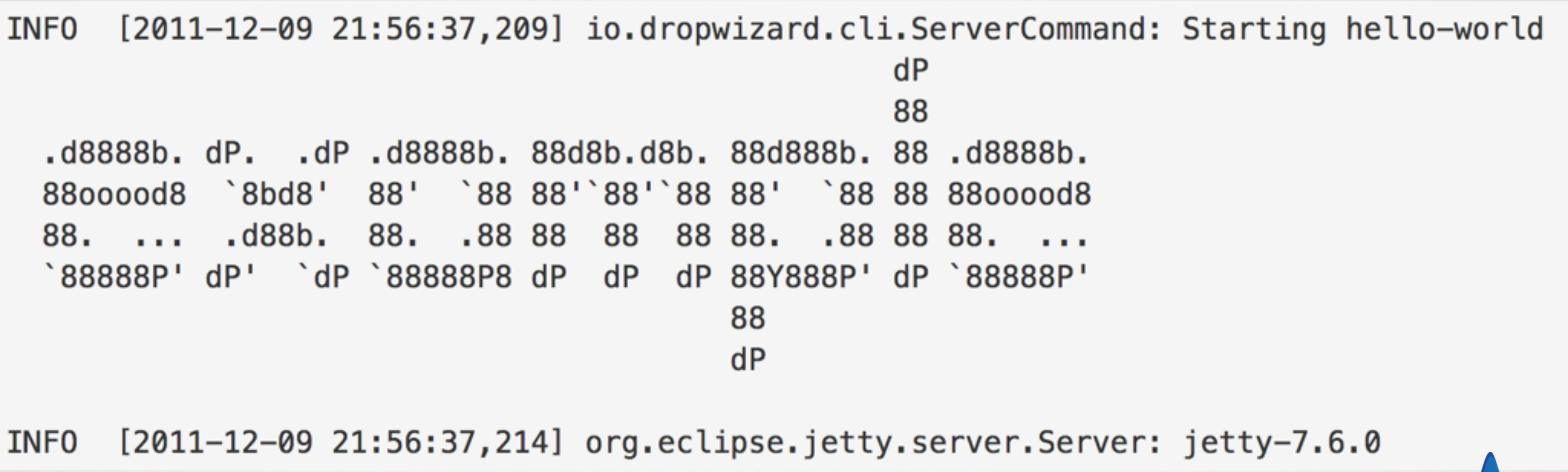


```
java -jar target/hello-world-0.0.1-SNAPSHOT.jar server hello-world.yml
```



```
INFO  [2011-12-03 00:38:32,927] io.dropwizard.cli.ServerCommand: Starting hello-world
INFO  [2011-12-03 00:38:32,931] org.eclipse.jetty.server.Server: jetty-7.x.y-SNAPSHOT
INFO  [2011-12-03 00:38:32,936] org.eclipse.jetty.server.handler.ContextHandler: start
INFO  [2011-12-03 00:38:32,999] com.sun.jersey.server.impl.application.WebApplicationI
INFO  [2011-12-03 00:38:33,041] io.dropwizard.setup.Environment:

    GET     /hello-world (com.example.helloworld.resources.HelloWorldResource)

INFO  [2011-12-03 00:38:33,215] org.eclipse.jetty.server.handler.ContextHandler: start
INFO  [2011-12-03 00:38:33,235] org.eclipse.jetty.server.AbstractConnector: Started l
INFO  [2011-12-03 00:38:33,238] org.eclipse.jetty.server.AbstractConnector: Started
```

ADD COOL BANNERS



```
INFO  [2011-12-09 21:56:37,209] io.dropwizard.cli.ServerCommand: Starting hello-world
                                                 dP
                                                 88
  .d8888b. dP.  .dP .d8888b. 88d8b.d8b. 88d888b. 88 .d8888b.
  88ooood8  `8bd8'  88'  `88 88'`88'`88 88'  `88 88 88ooood8
  88.  ...  .d88b.  88.  .88 88  88  88 88.  .88 88 88.  ...
  `88888P' dP'  `dP `88888P8 dP  dP  dP 88Y888P' dP `88888P'
                                        88
                                        dP

INFO  [2011-12-09 21:56:37,214] org.eclipse.jetty.server.Server: jetty-7.6.0
```

OPS FRIENDLY

- Now you can access the following URLs:
  - /ping
  - /healthcheck
  - /metrics
  - /threads

EXTRA

- Nice test support.
- Validation
- Servlets, Filters
- Custom Representations
- HTML Views

EXTRA

- Streaming output
- Caching
- Error Handling
- SSL
- Logging

EXTRA

- Metrics and Metrics Reporters
- Commands (CLI)
- Tasks (admin HTTP interface)
- Managed Objects

SUMMARY

- Pros:
  - Easy, simple
  - Performant
  - ops friendly
  - Opinionated
  - No web container
  - Community!

- Cons:
  - Opinionated
  - Eran will say more ;-)

REFS

- https://dropwizard.github.io/dropwizard/
- http://martinfowler.com/articles/microservices.html