Cloud Native Java EE with WildFly Swarm

Matthias Wessendorf – matzew @ redhat . com
@mwessendorf

From WAR to JAR to Linux Container to Cloud

Matthias Wessendorf – Red Hat | matzew AT redhat DOT com | @mwessendorf

Microservice....
like SOA, but different ...

- Microservices are differentprimarily due to innovations like:
- Linux containers,
- automated, elastic infrastructure, you know, the cloud
- plus wide adoption of CI, continuous integration
- and the growing adoption of DevOps principles & practices

Matthias Wessendorf – Red Hat | matzew AT redhat DOT com | @mwessendorf

What is Java EE anyway?

Matthias Wessendorf – Red Hat | matzew AT redhat DOT com | @mwessendorf

Perspectives on Java EE

- It’s different things to different people:
  - A collection of (useful) API’s
  - Technical capabilities of a system
  - A love/hate relationship (of the past)
  - (Existing) knowledge and expertise

Hello WildFly Swarm

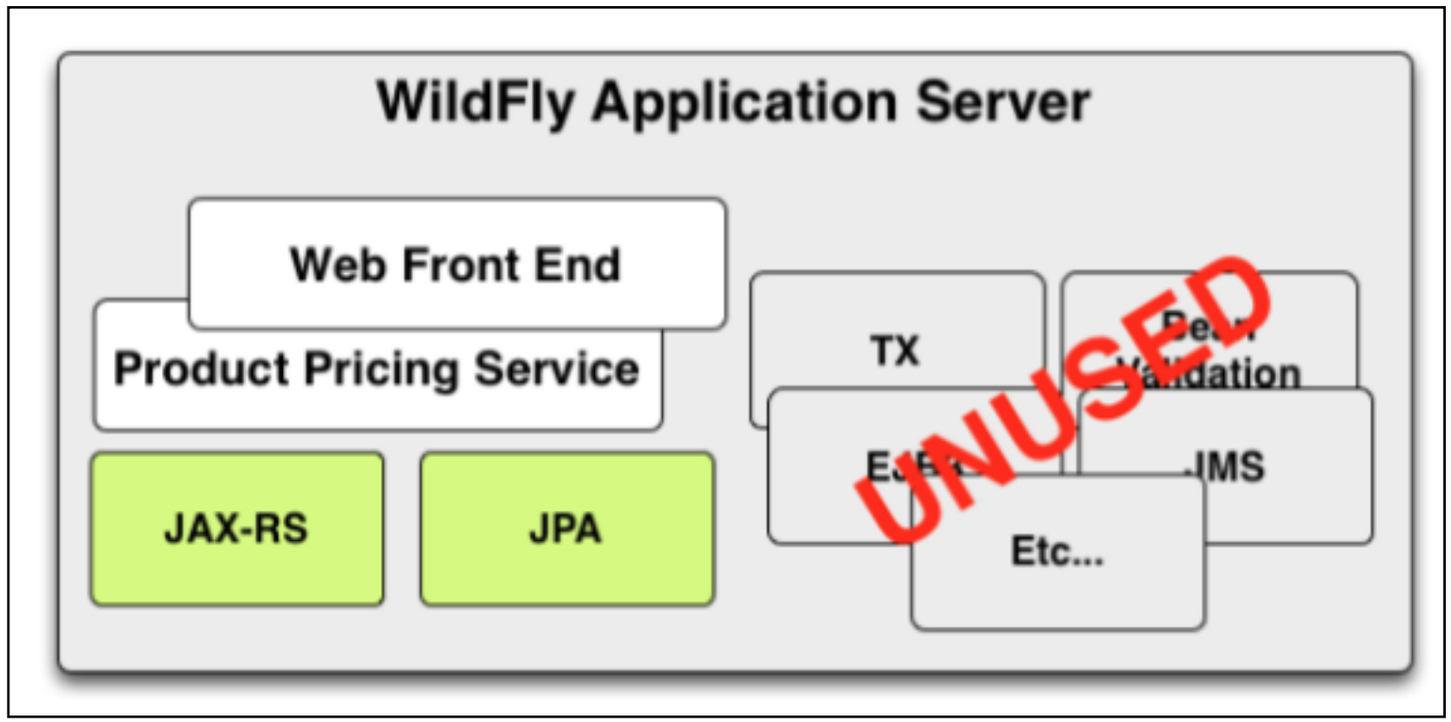
Matthias Wessendorf – Red Hat | matzew AT redhat DOT com | @mwessendorf

WildFly Swarm

- OSS Project sponsored by Red Hat
  http://wildfly-swarm.io/
- Sidekick of Wildfly Application Server
- Small, but ambitious and friendly community
- Part of a bigger system of interrelated projects under the JBoss / Red Hat umbrella

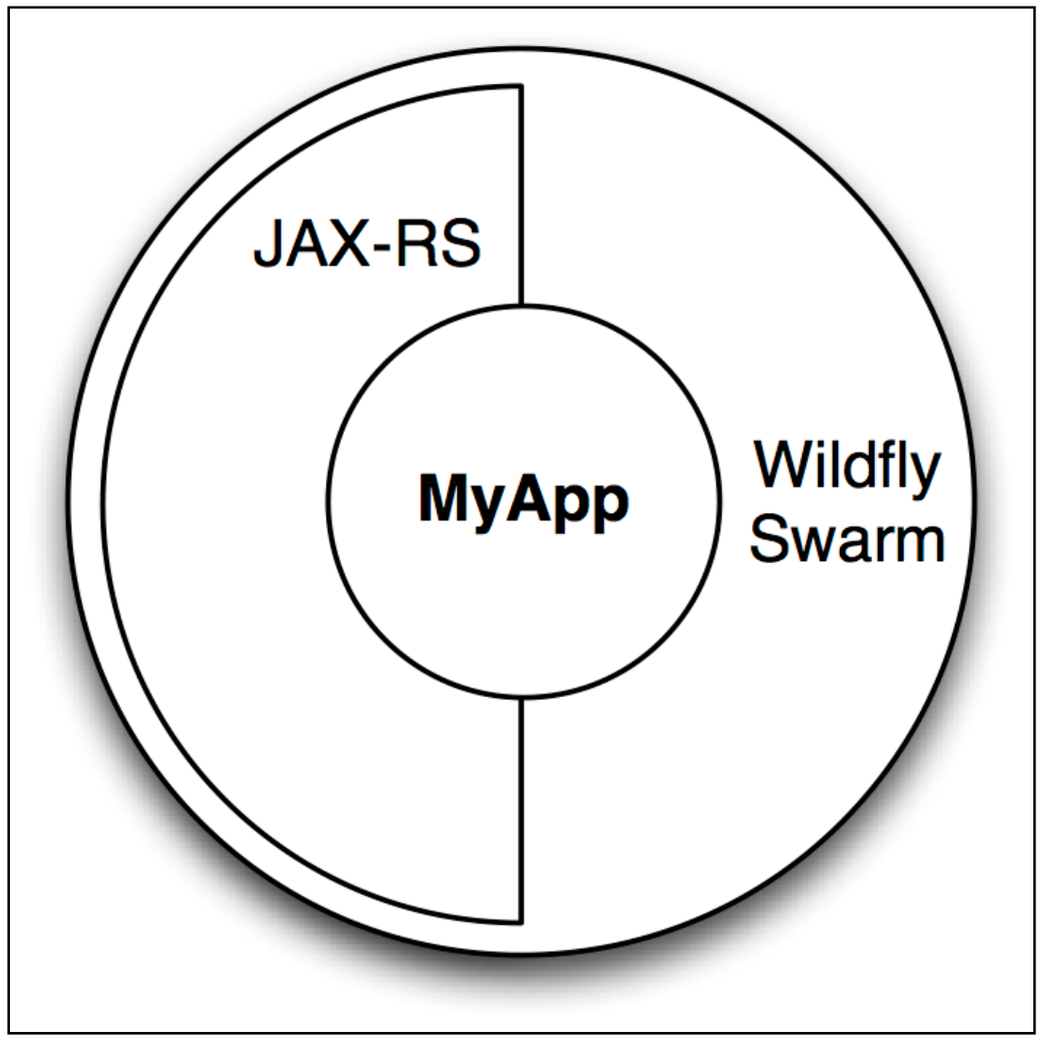
Just Enough App Server

- Use the API’s you want
- Include the capabilities you need
- Wrap it up for deployment



Uberjar

- A single .jar file containing your application
- the portions of WildFly required to support it
- an internal Maven repository of dependencies,
- plus a shim to bootstrap it all



Fractions

- A well-defined collection of application capabilities.
  - May map directly to a WildFly subsystem,
  - or bring in external capabilities such as Netflix Ribbon.

What Fractions can do

- Enable WildFly subsystems (JAX-RS, Infinispan)
- Integrate additional system capabilities (Topology)
- Provide deployments (ribbon-webapp, jolokia)
- Alter deployments (keycloak)

DEMO

Code snippets:

https://gist.github.com/matzew/565ae0bb5f1df39cddeb36c94ecb3742

… from WAR to JAR to Linux Container to CLOUD !

javaee-samples / **javaee7-simple-sample**

Watch 4 | Star 13 | Fork 15

A simple Java EE 7 Sample

61 commits | 1 branch | 8 releases | 2 contributors

Branch: **master** **javaee7-simple-sample** / +

arun-gupta cleaning up whitespace — Latest commit `9ef152e` on Sep 19

src/main	adding 'all' beans.xml	8 months ago
.gitignore	adding gitignore	a year ago
README.asciidoc	reorganizing	8 months ago
pom.xml	cleaning up whitespace	a month ago

README.asciidoc

A simple Java EE 7 Sample

This is a trivial Java EE 7 sample.

Code
Issues 0
Pull requests 0
Wiki
Pulse
Graphs

SSH clone URL

`git@github.com:j`

You can clone with HTTPS, SSH, or Subversion.

Clone in Desktop

Download ZIP

Matthias Wessendorf – Red Hat | matzew AT redhat DOT com | @mwessendorf

Going beyond simple (and Java EE)

Matthias Wessendorf – Red Hat | matzew AT redhat DOT com | @mwessendorf

CUSTOM Config

```
public class Main {

    public static void main(String... args) throws Exception {

        Container container = new Container();

        container.fraction(new DatasourcesFraction()
                .jdbcDriver(new JDBCDriver("h2")
                        .driverName("h2")
                        .driverDatasourceClassName("org.h2.Driver")
                        .xaDatasourceClass("org.h2.jdbcx.JdbcDataSource")
                        .driverModuleName("com.h2database.h2"))
                .dataSource(new DataSource("LibraryDS")
                        .driverName("h2")
                        .jndiName("java:/LibraryDS")
                        .connectionUrl("jdbc:h2:./library;DB_CLOSE_ON_EXIT=TRUE")
                        .userName("sa")
                        .password( "sa" )));

        container.start();

    }
}
```

Keycloak protected resources

```
public class Main {

    public static void main(String... args) throws Exception {

        Container container = new Container();

        JAXRSArchive deployment = ShrinkWrap.create(JAXRSArchive.class);
        deployment.addPackage(Main.class.getPackage());
        deployment.as(Secured.class)
                .protect("/items")
                .withMethod("GET")
                .withRole("*");

        container.start();
        container.deploy(deployment);

    }
}
```

Publishing Service Interface Description

```java
@Path("/time")
@Api(value = "/time", description = "Get the time", tags = "time")
@Produces(MediaType.APPLICATION_JSON)
public class TimeResource {

    @GET
    @Path("/now")
    @ApiOperation(value = "Get the current time",
            notes = "Returns the time as a string",
            response = String.class
    )
    @Produces(MediaType.APPLICATION_JSON)
    public String get() {

        return String.format("{\"value\" : \"The time is %s\"}",
                new DateTime()
        );
    }
}
```

```
$ curl http://localhost:8080/swagger.json
```


```
{
  "basePath": "/",
  "paths": {
    "/time/now": {
      "get": {
        "description": "Returns the time as a string",
        "operationId": "get",
        "parameters": [],
        "produces": [
          "application/json"
        ],
        "responses": {
          "200": {
            "description": "successful operation",
            "schema": {
              "type": "string"
            }
          }
        },
        "summary": "Get the current time"
      }
    }
  }
}
```

Other Noteworthy Features

- Testing:
  - Arquillian (in container, web driver)
  - Consumer-Driven Contracts (expressing and asserting expectations of a provider contract)
- Logging & Monitoring
  - Simple REST interface on each node
  - Centralised Logging with Logstash
  - Push Runtime Data to Hawkular,Influx,etc
- Remote Management
  - CLI (full access to the server config and runtime state)

Thanks

Heiko Braun (Swarm Team) for original slides

Questions ?

Matthias Wessendorf – Red Hat | matzew AT redhat DOT com | @mwessendorf