Getting started with JBoss

Slides and Workshop
(Postgres + JBoss + Azure)

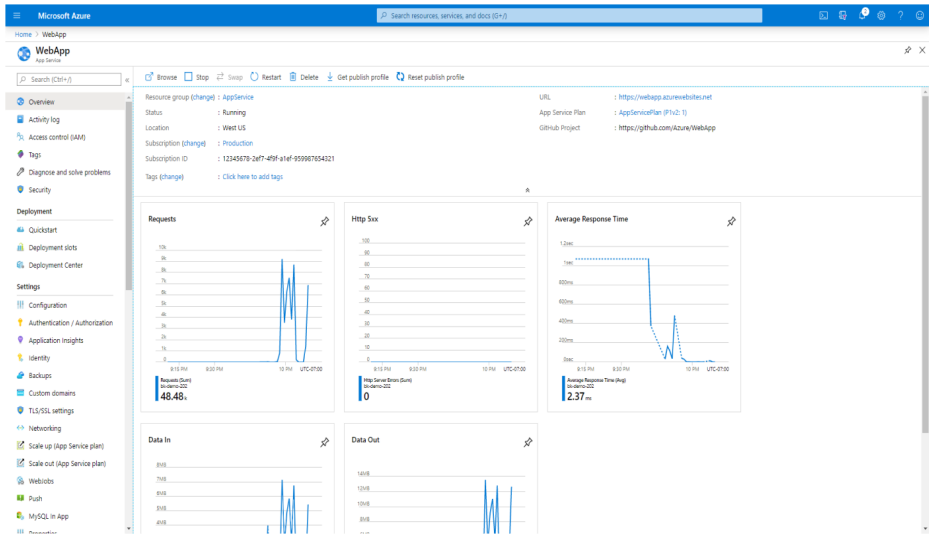
https://aka.ms/jboss-azure

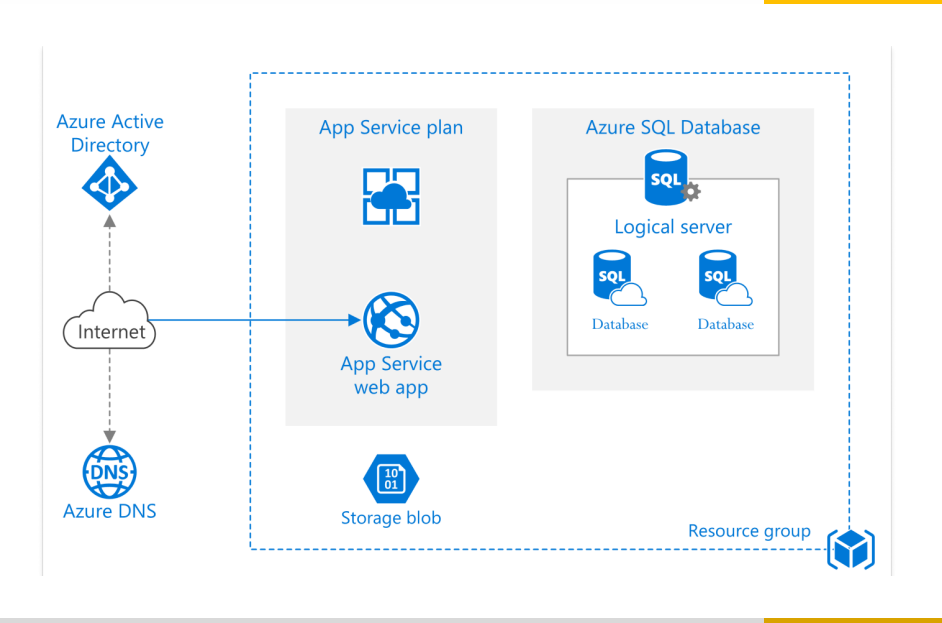
Agenda

- JBoss On Azure App Service
- MicroProfile
- Demo
- Resources

Azure App Service on Linux

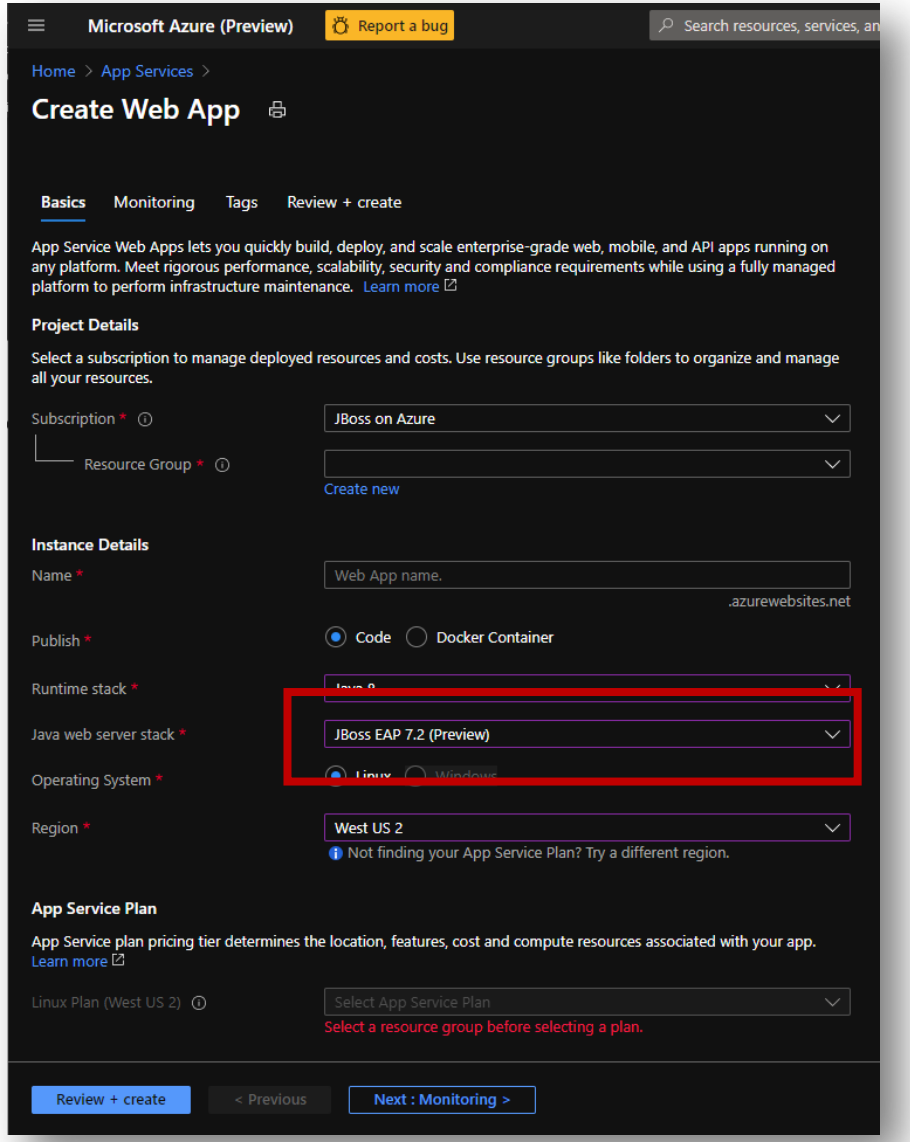
- **Managed** platform with built-in infrastructure maintenance and security patching
- Built-in **CI/CD** integration and zero-downtime deployments
- Integration with **virtual networks** and ability to run in an **isolated** and dedicated App Service environment
- **Security** and **compliance**
- **Scale** apps on an enterprise grade platform
- Simplify operations with built-in monitoring



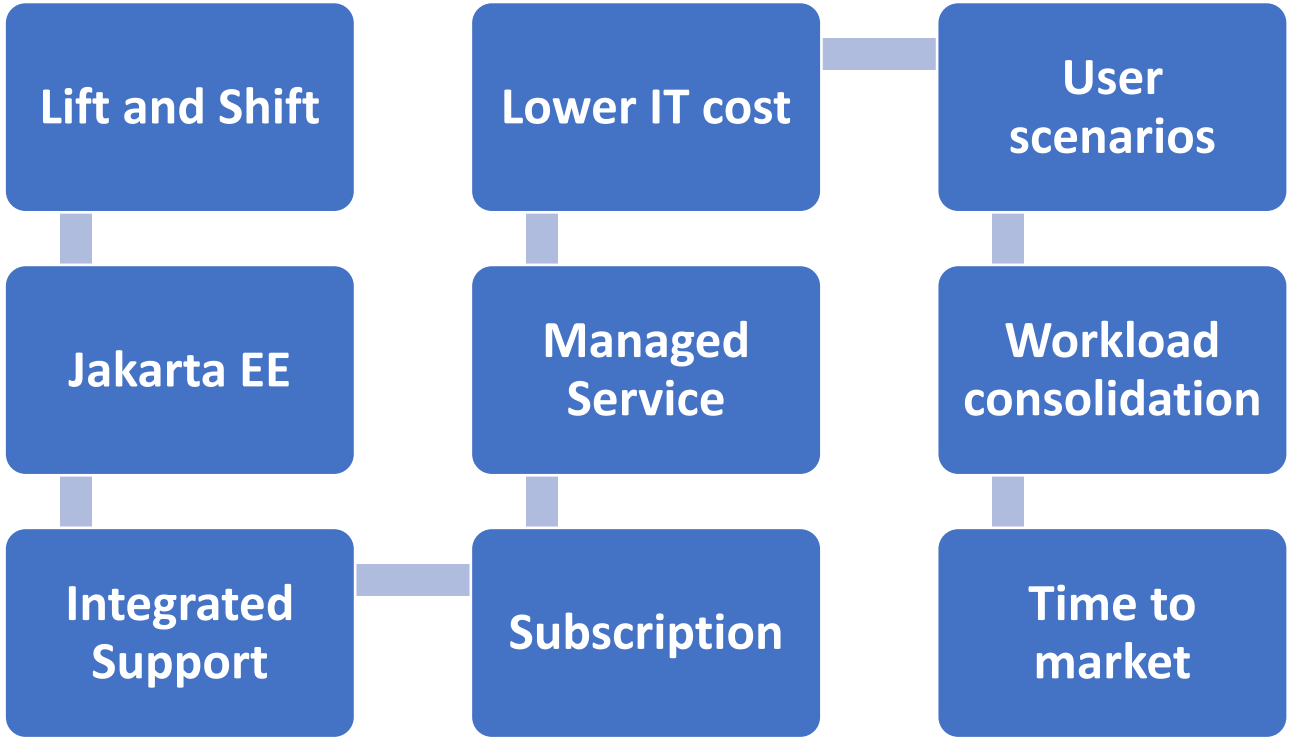


JBoss EAP on App Service Linux

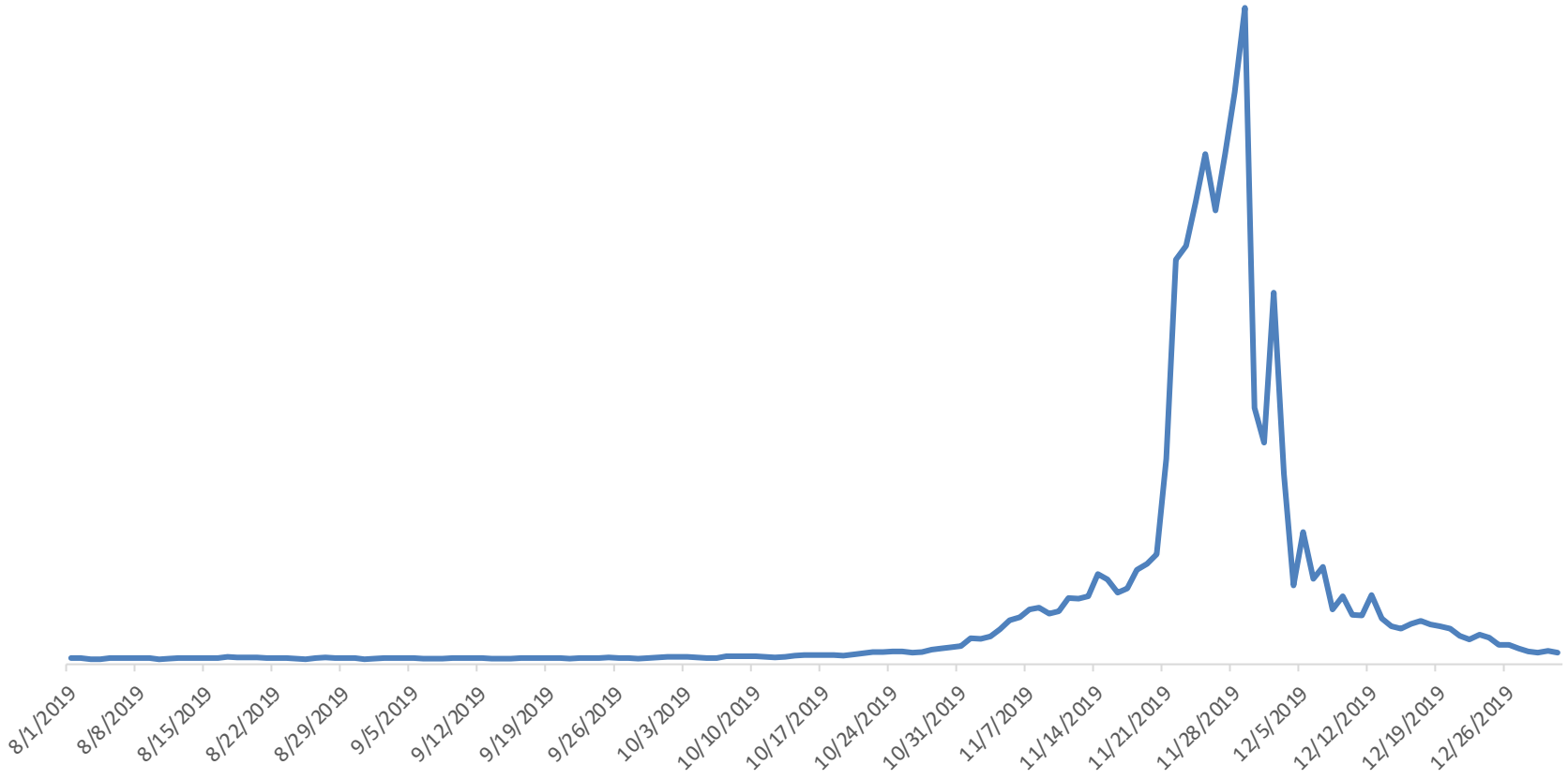
- **Public preview**
  - Standalone mode
  - Not recommended for production yet
- **App Service SKUs available at GA:**
  - Premium v2, v3
  - Isolated v1, v2
- **Offer includes:**
  - Red Hat JBoss Enterprise Application Platform (EAP)
  - Red Hat Enterprise Linux (RHEL)
  - Azure App Service
  - Integrated support (for GA only) – MS CSS + RH GSS
- **Pricing** – released at GA, PAYG only



Why JBoss EAP on App Service Linux?



Black Friday



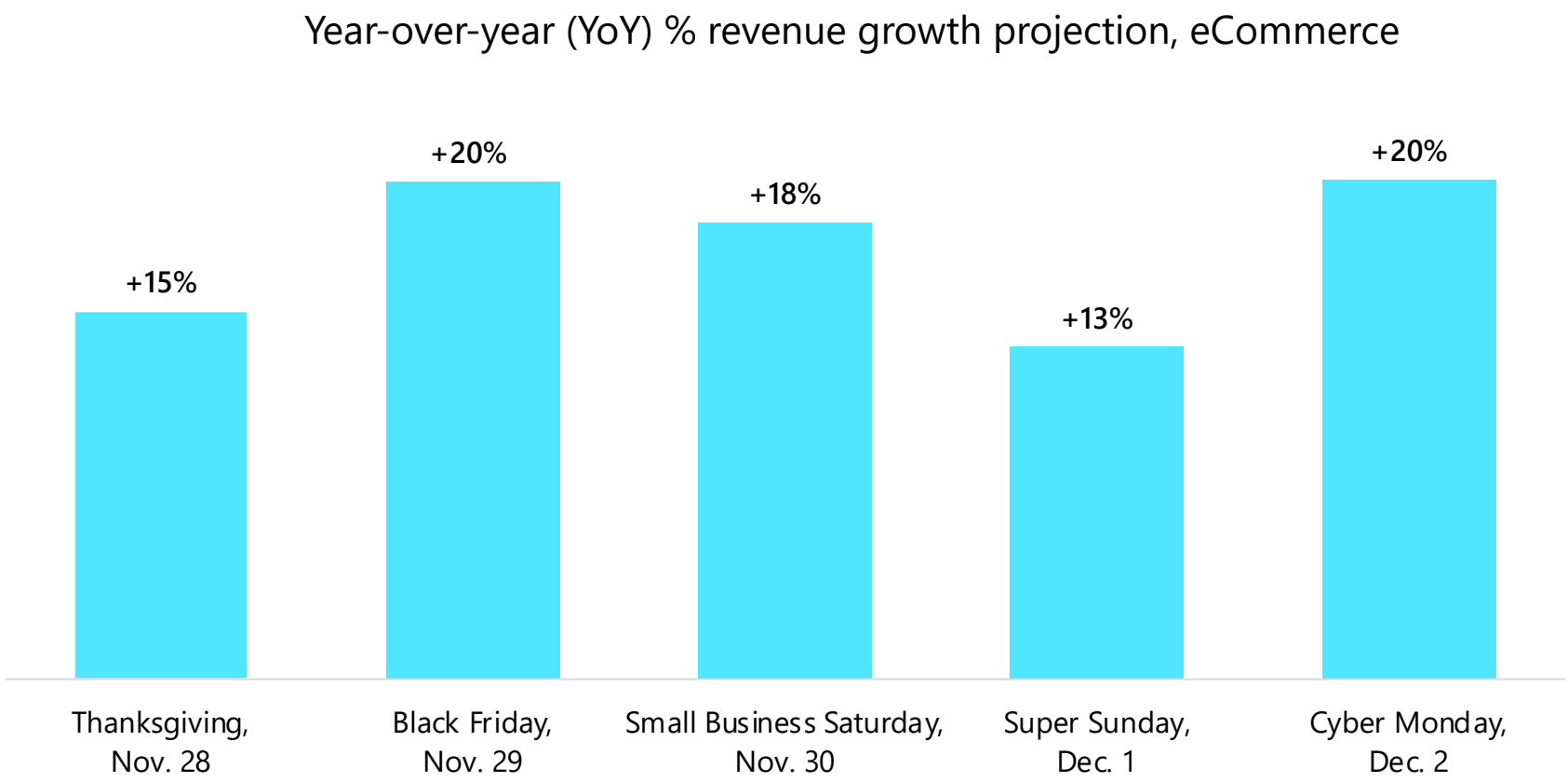
Holiday search volume change against August baseline:

September	-1%
October	62%
November	2,989%
December	949%

Source: Microsoft internal data, Dec. 2019 & May 2020

Microsoft

COVID-19 related eCommerce focus brings astronomical expectations for holiday's biggest weekend



- Source: eMarketer 'US Holiday 2019 and Looking Ahead to Holiday 2020 Planning'

Microsoft

We need a Business Strategy

Higher utilization of resources

Elasticity without reserving capacity

Upper limits and lower limits are under our control

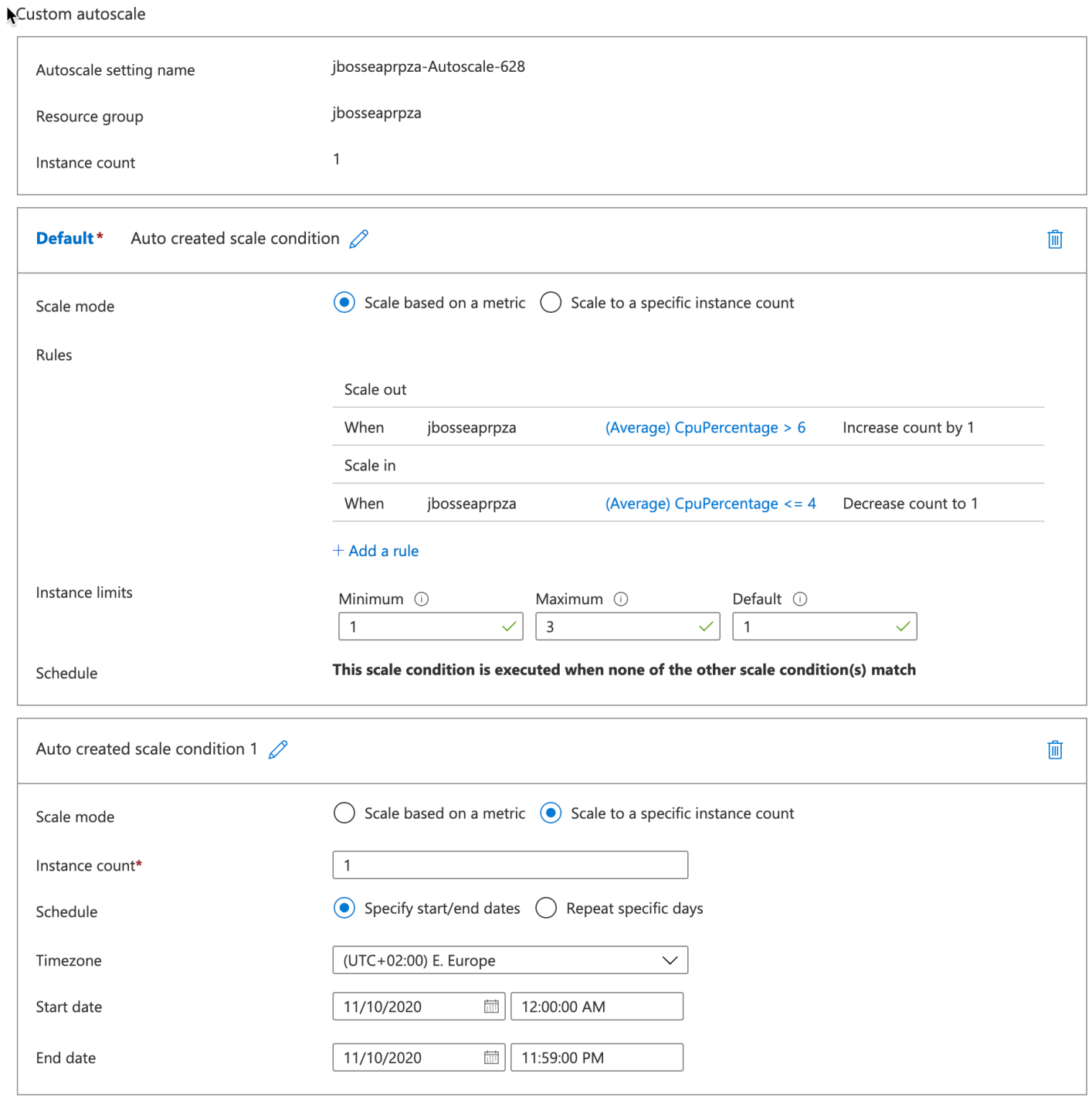
Pay-As-you-Go

Technical Approach

- I need to Scale specific apps
  - Scale the number of individual app instances
- Scale differently on weekends
  - Scale down to 1 instance on weekends.
- Scale differently during holidays
  - During holiday season, override the defaults and have more capacity at your disposal.
- Scale based on custom metrics
  - Scale an App based on CPU or memory

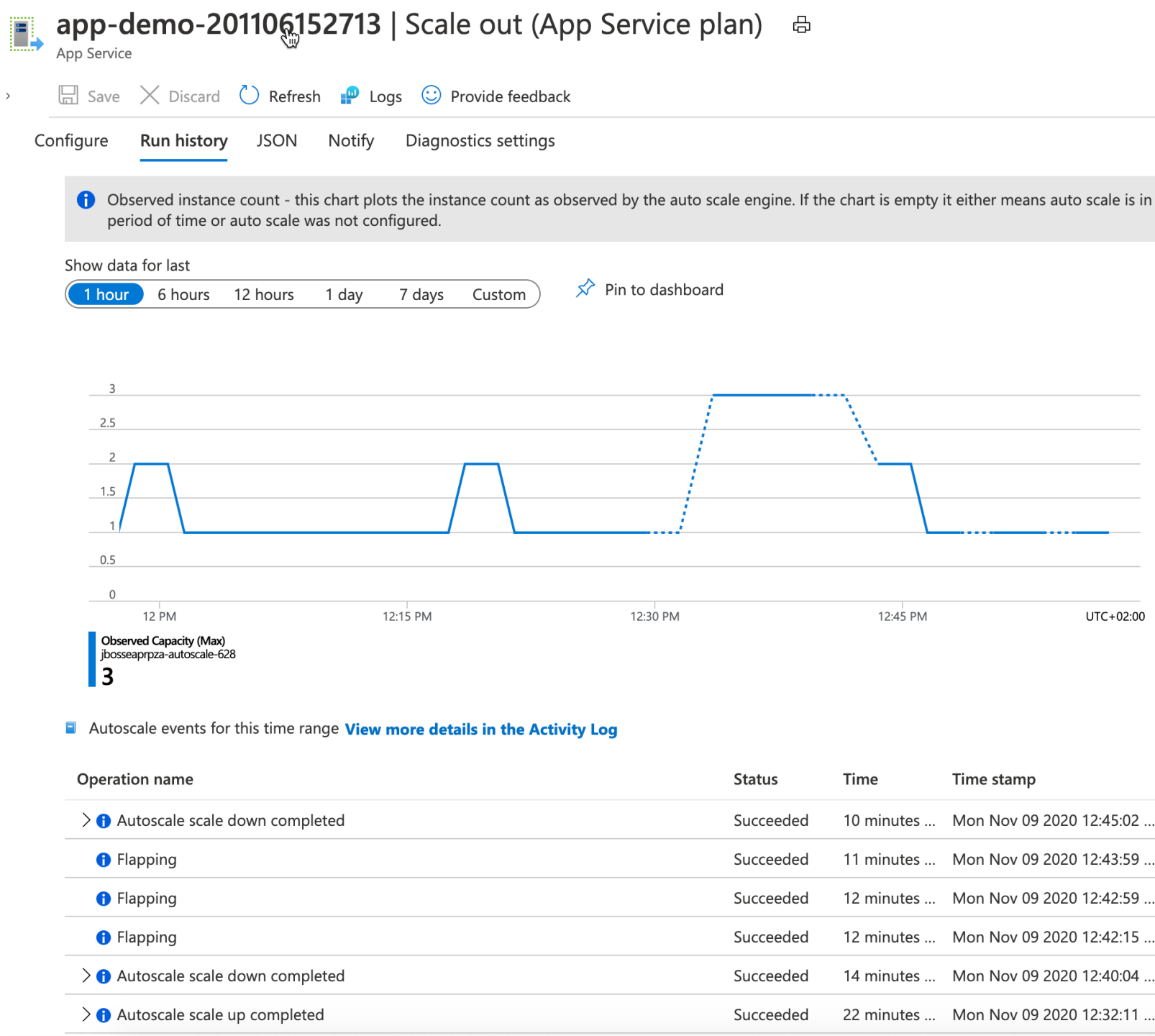
Autoscale

- Metric driven
- Scale Instance Count
- Can be Scheduled
- Executed in Workflow of matching conditions



Autoscale Settings

- Run history
- Activity log
- Tracing
- Notifications and Webhooks



Eclipse MicroProfile

Eclipse MicroProfile

Eclipse MicroProfile is an open-source community specification for Enterprise Java microservices

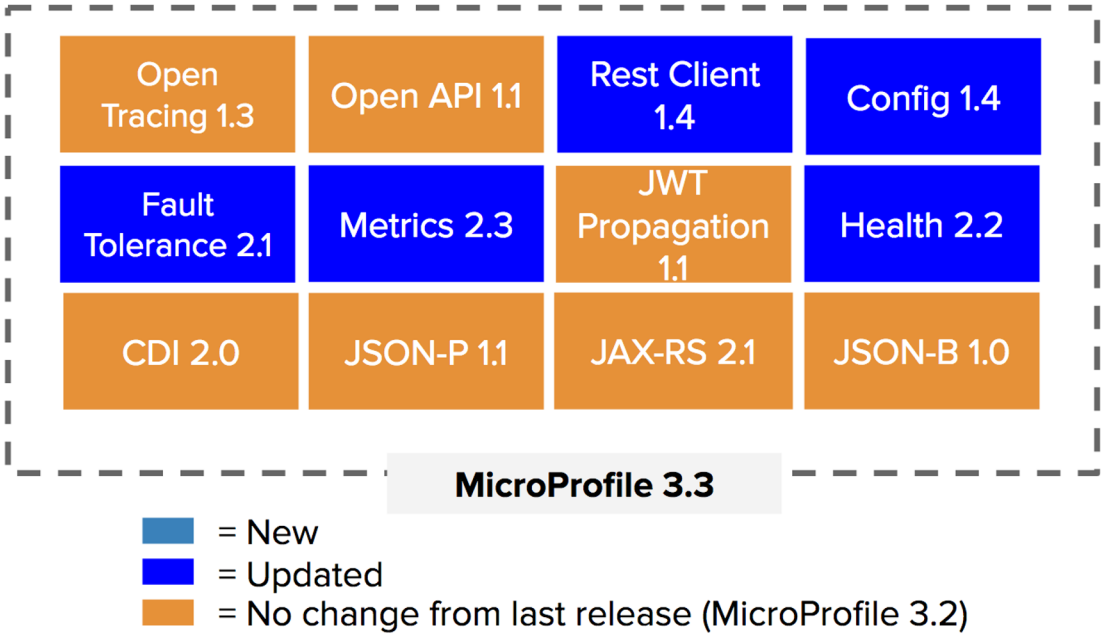
A community of individuals, organizations, and vendors collaborating within an open source (Eclipse) project to bring microservices to the Enterprise Java community

Community - individuals, organizations, vendors

Current MicroProfile implementations

Eclipse MicroProfile 3.3 Released!

On February 18, 2020, MicroProfile 3.3 was released.
Offered in the release:



Tooling

- MicroProfile Starter 1.0 is now generally available
- https://start.microprofile.io VS code extension released

Feature Tour

Open API

- Generates Open API (**Swagger**) documents from JAX-RS endpoints
- Basically **WSDL for REST**
- Enabled by default in all MicroProfile applications

Open API Example

```
@GET
@Operation(description="Get all current memberships")
@APIResponses({
    @APIResponse(responseCode="200",
        description="Successful, returning memberships"),
    ...
})
public List<Membership> getAllMemberships() {
```

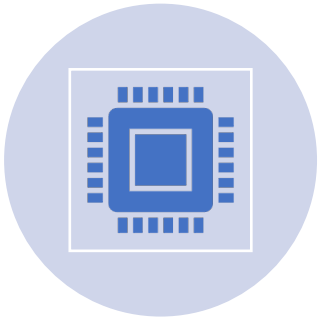
Open Tracing

- Removes **Challenges** of Tracing
- **Standard** for instrumenting microservices for distributed tracing
  - Jaeger, Zipkin
- Enterprise Java binding to the Open Tracing specification

Open Tracing Example

```
@GET
@Path("{id}")
@Traced(operationName="GetMembershipById", value=true)
public Membership getMembership(
    @NotNull @PathParam(value="id") int id) {
```

JWT Propagation



Security for Stateless REST Services

Basically what OAuth2, **OpenID** Connect (OIDC) and JSON Web Tokens (**JWT**) do

Think of it as **SAML** for REST

Enterprise **Java binding** for the JWT specification

JWT Propagation Example

```
@GET
@RolesAllowed({"admin"})
public List<Membership> getAllMemberships() {
```

Configuration

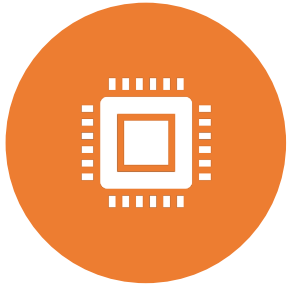
- Java EE mostly focuses on **static** configuration
- Applications may need to be configured based on a **running** environment

- **MicroProfile Config** allows modification of configuration values without repackaged
- Influenced by DeltaSpike Config and Apache Tamaya

Configuration Example

```
@Inject
@ConfigProperty(name="host.name", defaultValue="localhost")
private String hostname;
```

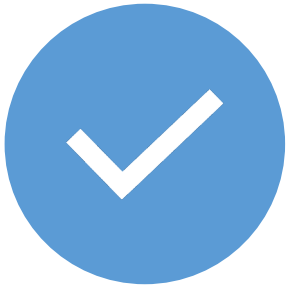
Rest Client



JAX-RS provides a powerful client API, but it can be **hard** to use and not really **type** safe

Several JAX-RS implementations already exist

Like JAX-WS/SOAP clients

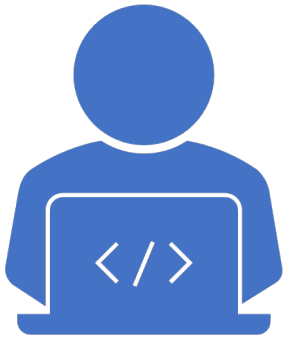


MicroProfile standardizes this capability

Rest Client Example

```
String apiUrl = "http://localhost:9080/movieReviewService";
MovieReviewService reviewService =
    RestClientBuilder.newBuilder()
            .baseUrl(apiUrl)
            .build(MovieReviewService.class);
Review review = new Review(3 /*stars*/, "Good Movie.");
reviewService.submitReview(movie, review);
```

Health Check



Probe state of a computing node from another machine (such as the Kubernetes service controller)

Automate reporting on the state of Cloud nodes

Health Check Example

```
@Health
@ApplicationScoped
public class CheckDiskSpace implements HealthCheck {
    ...
    public HealthCheckResponse call() {
        ...
    }
}
```

Metrics

- Designed for **distributed** systems
- Adds well-known monitoring **endpoints** and values for each process
- Targets cloud environments with distributed metrics registries/consoles

Metrics Example

```
@GET
@Timed(name="get_all_memberships_time", absolute=true,
    unit=MetricUnits.MICROSECONDS)
public List<Membership> getAllMemberships() {
...
@POST
@Counted(name="memberships_created", absolute=true,
monotonic=true)
public Membership createMembership(
```

Fault Tolerance

- Leverage different **strategies** to guide the execution
- **Retry policies, bulkheads, and circuit breakers**
- **Fallbacks** offer an alternative result
- Influenced by Hystrix/Failsafe

Fault Tolerance Example

```
@GET
@Path("{id}")
@CircuitBreaker(failOn=RuntimeException.class,
requestVolumeThreshold=1,
    failureRatio=1, delay=10,
delayUnit=ChronoUnit.SECONDS)
@Timeout(value=3, unit=ChronoUnit.SECONDS)
@Bulkhead(2)
public Membership getMembership(
```

Summary

MicroProfile fills an important gap with Java EE and microservices

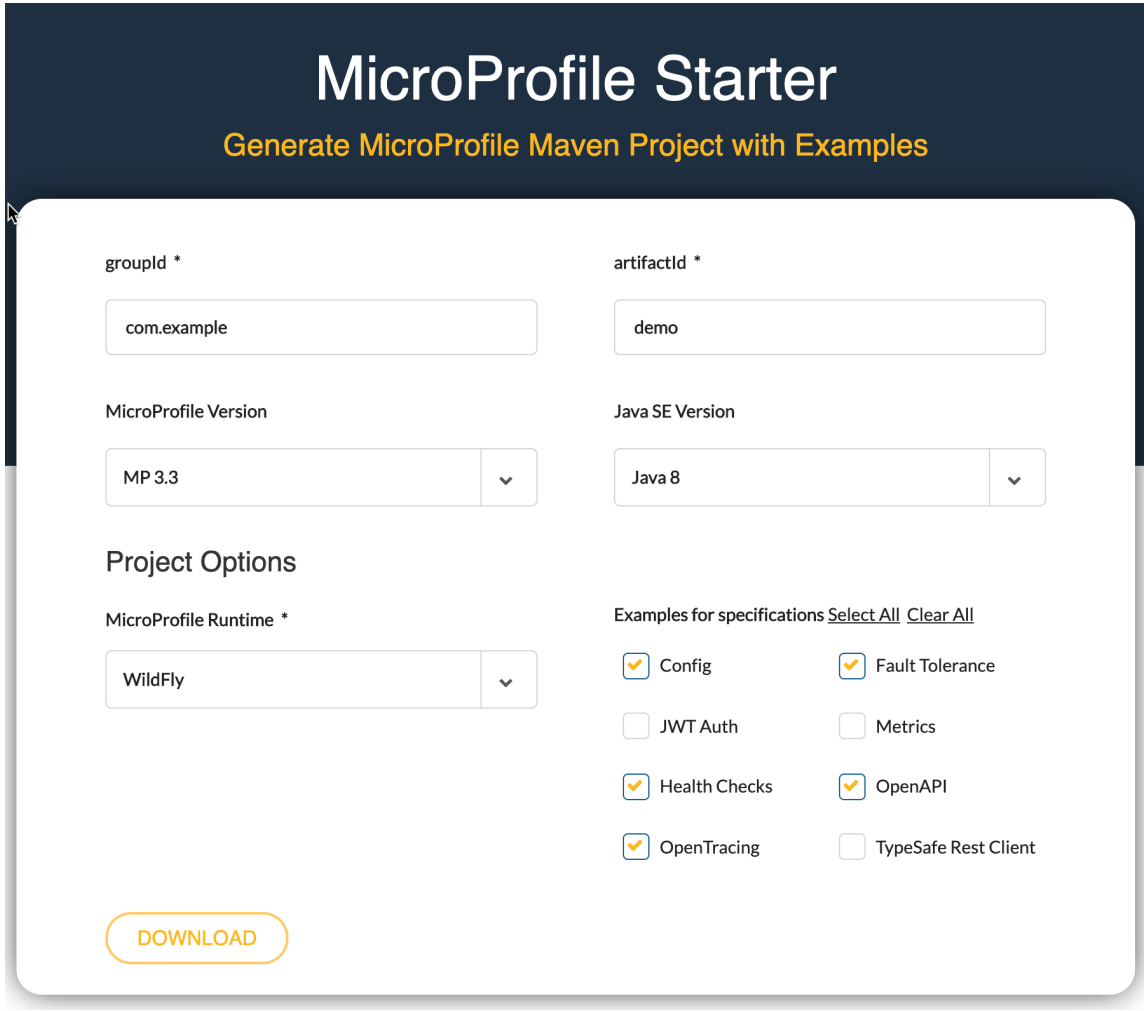
JBoss on App Service is in **Public preview**

AutoScale

Quick to start - Demo time!

Demo

1. Create project
   curl -O -J -L 'https://start.microprofile.io/api/project?supportedServer=WILDFLY&groupId=com.example&artifactId=demo&mpVersion=MP33&javaSEVersion=SE8&selectedSpecs=CONFIG&selectedSpecs=FAULT_TOLERANCE&selectedSpecs=HEALTH_CHECKS&selectedSpecs=OPEN_API&selectedSpecs=OPEN_TRACING'
   - Unzip demo.zip
2. Deploy
   - mvn com.microsoft.azure:azure-webapp-maven-plugin:1.12.0:config
   - mvn com.microsoft.azure:azure-webapp-maven-plugin:1.12.0:deploy
   - az webapp log tail --name demo-XXXXX -g demo-XXXX-rg
3. Browse to “https://xxxxxx.azurewebsites.net”



https://start.microprofile.io

Slides and Workshop (Postgres + JBoss + Azure)

https://aka.ms/jboss-azure