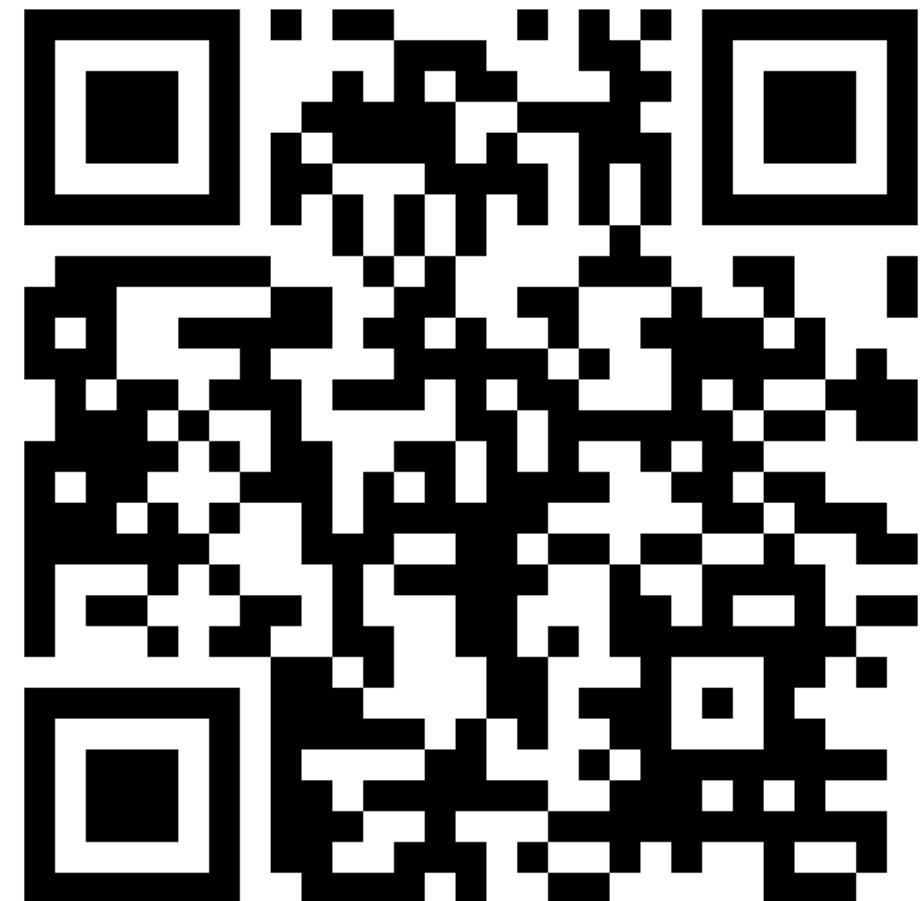


Getting started with JBoss

Slides and Workshop
(Postgres + JBoss + Azure)



<https://aka.ms/jboss-azure>



Agenda

JBoss On Azure App Service

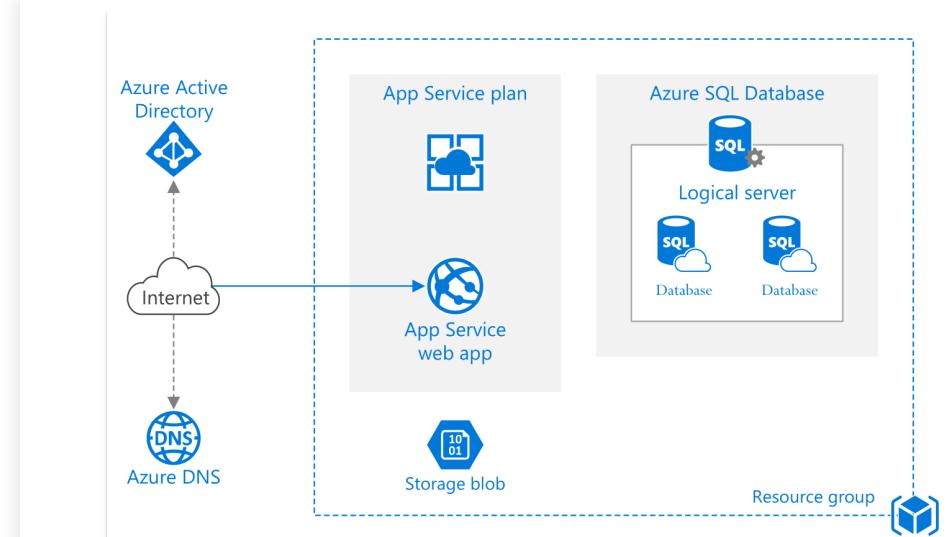
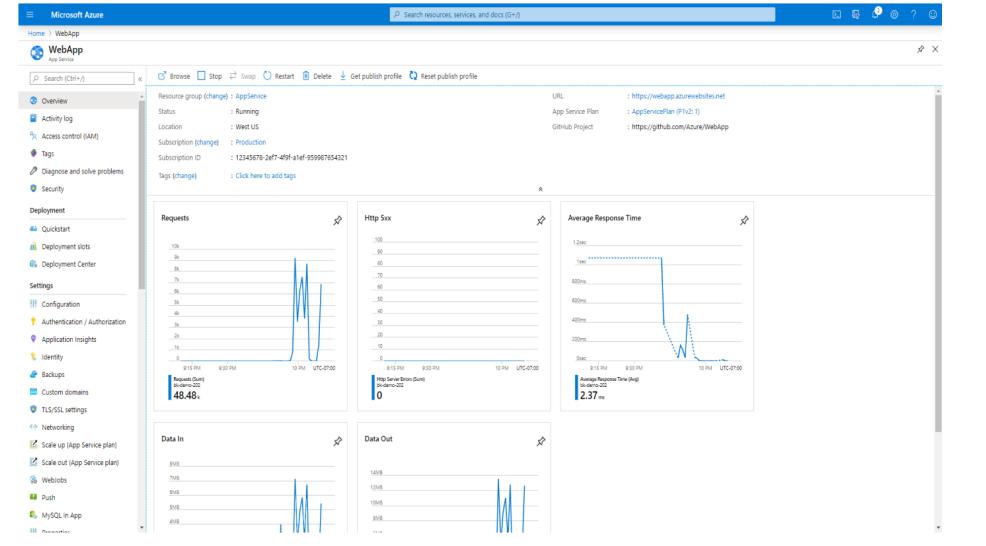
MicroProfile

Demo

Resources

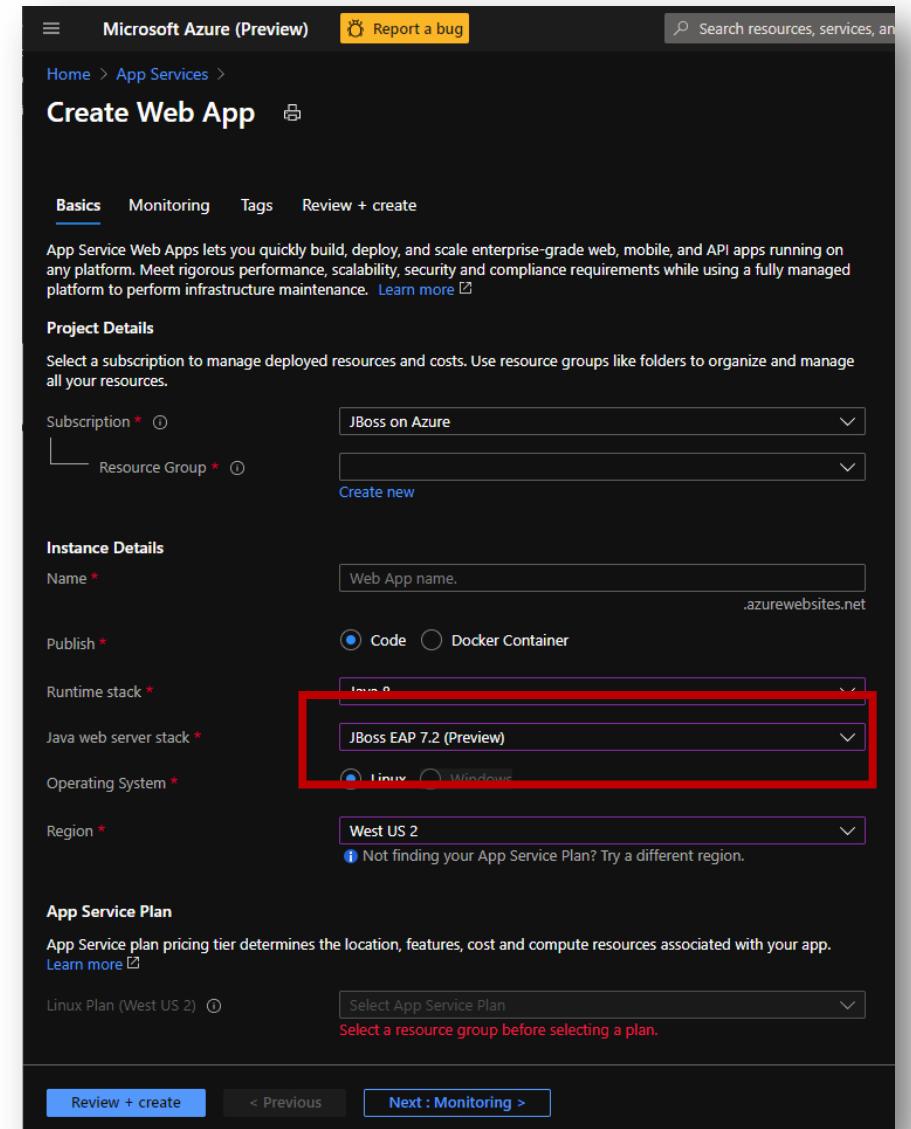
Azure App Service on Linux

- **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



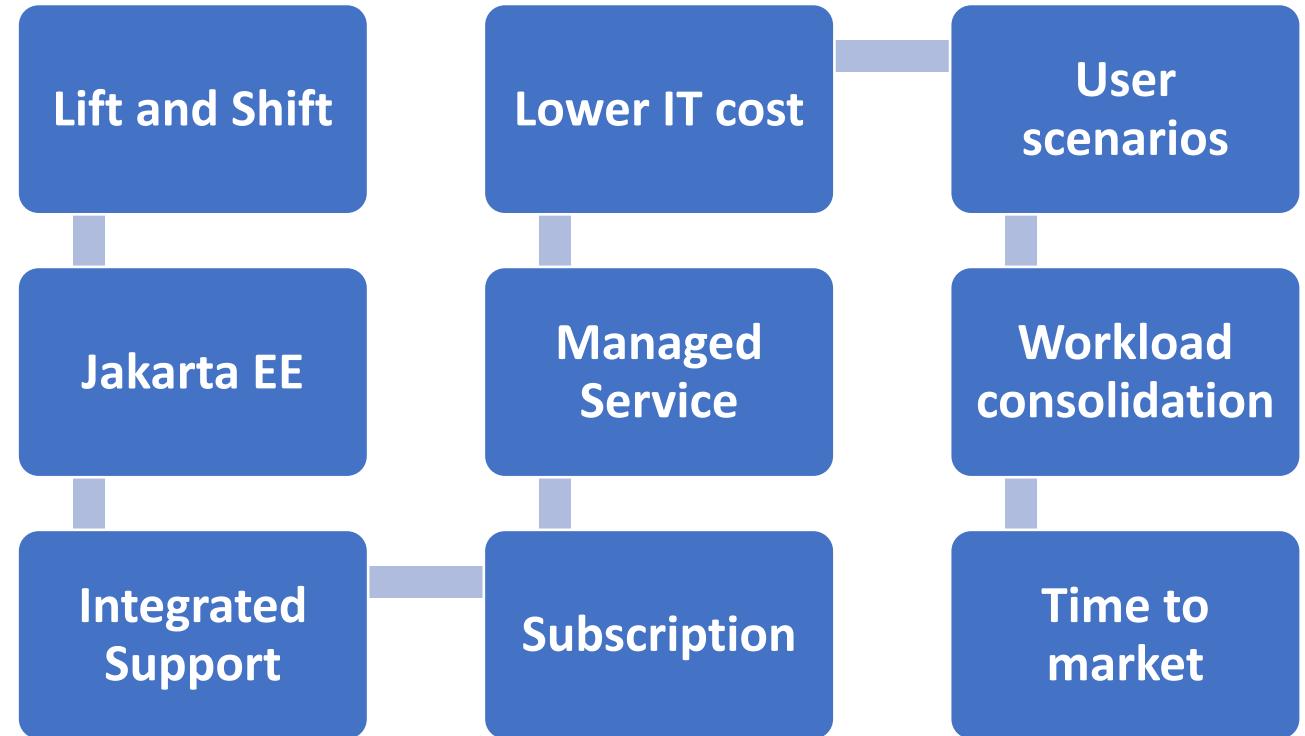
JBoss EAP on App Service Linux

- **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

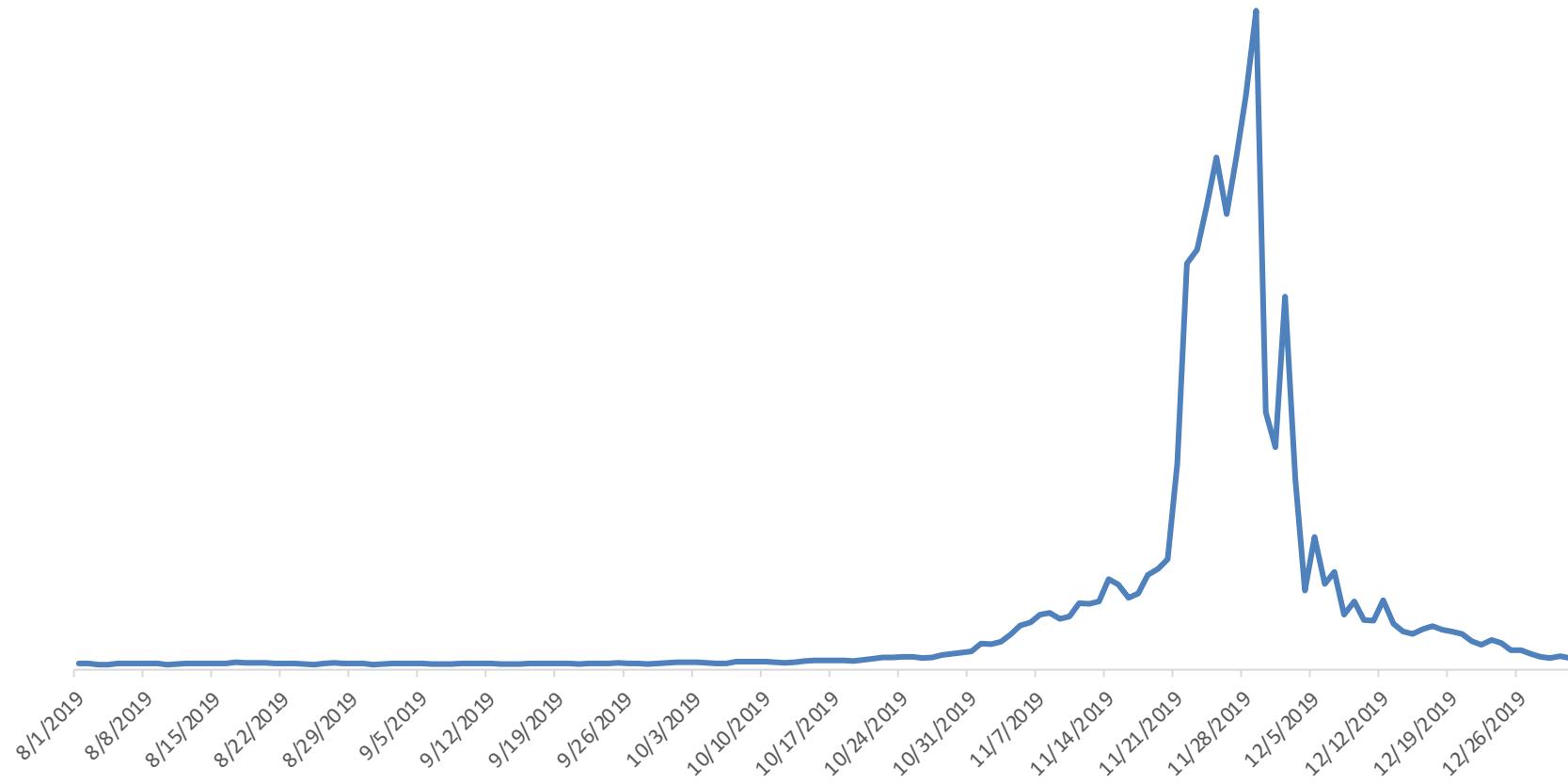




Why JBoss EAP on App Service Linux?



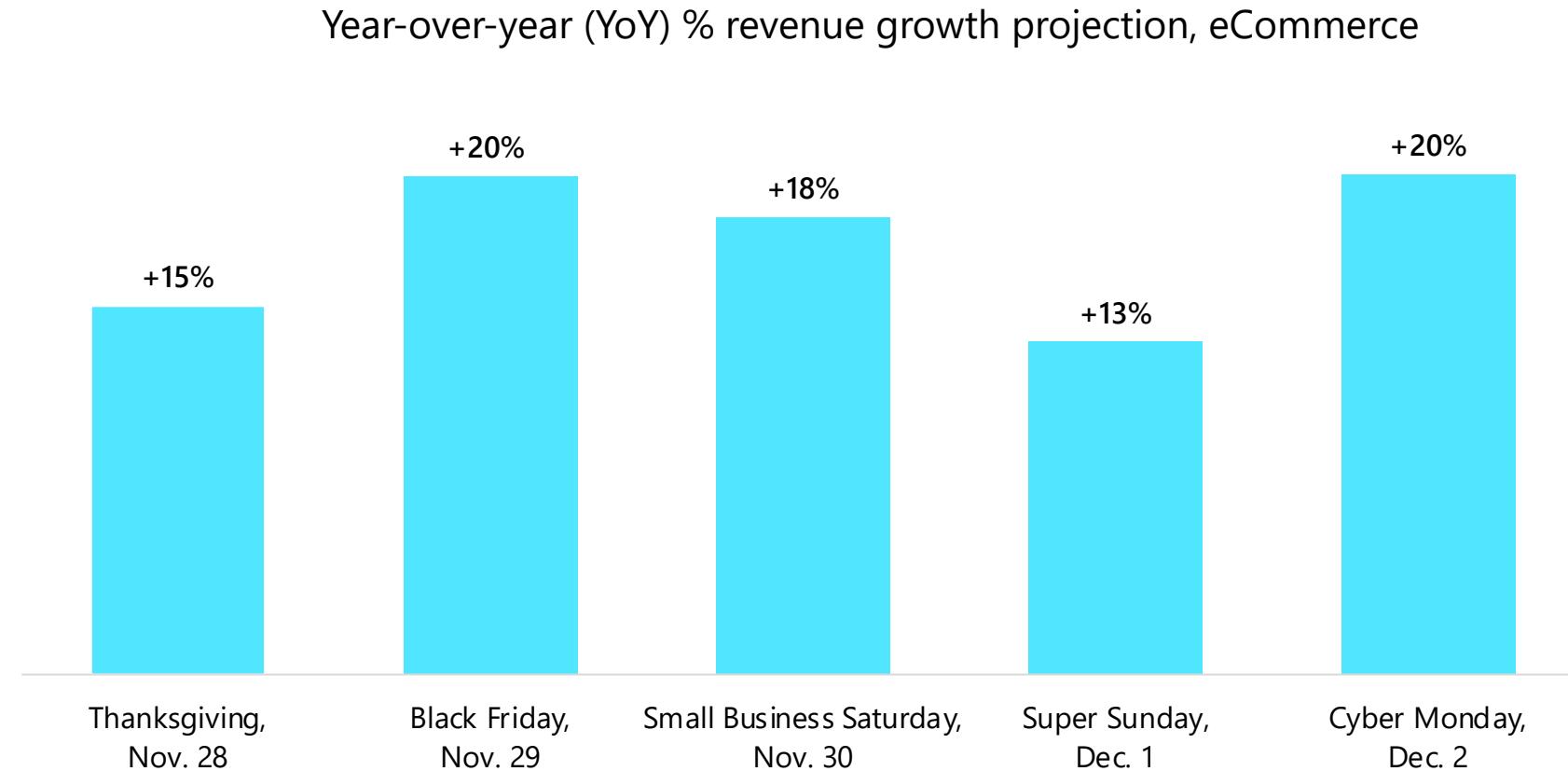
Black Friday



Holiday search volume
change against August
baseline:

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

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'

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

Custom autoscale

Autoscale setting name	jbosseaprz-Autoscale-628
Resource group	jbosseaprz
Instance count	1

Default* Auto created scale condition [Edit](#) [Delete](#)

Scale mode	<input checked="" type="radio"/> Scale based on a metric <input type="radio"/> Scale to a specific instance count				
Rules	<table border="1"><tr><td>Scale out</td></tr><tr><td>When jbosseaprz (Average) CpuPercentage > 6 Increase count by 1</td></tr><tr><td>Scale in</td></tr><tr><td>When jbosseaprz (Average) CpuPercentage <= 4 Decrease count to 1</td></tr></table> + Add a rule	Scale out	When jbosseaprz (Average) CpuPercentage > 6 Increase count by 1	Scale in	When jbosseaprz (Average) CpuPercentage <= 4 Decrease count to 1
Scale out					
When jbosseaprz (Average) CpuPercentage > 6 Increase count by 1					
Scale in					
When jbosseaprz (Average) CpuPercentage <= 4 Decrease count to 1					
Instance limits	Minimum <input type="text" value="1"/> Maximum <input type="text" value="3"/> Default <input type="text" value="1"/>				
Schedule	<p>This scale condition is executed when none of the other scale condition(s) match</p>				

Auto created scale condition 1 [Edit](#) [Delete](#)

Scale mode	<input type="radio"/> Scale based on a metric <input checked="" type="radio"/> Scale to a specific instance count
Instance count*	<input type="text" value="1"/>
Schedule	<input checked="" type="radio"/> Specify start/end dates <input type="radio"/> Repeat specific days
Timezone	(UTC+02:00) E. Europe
Start date	<input type="text" value="11/10/2020"/> <input type="button" value="12:00:00 AM"/>
End date	<input type="text" value="11/10/2020"/> <input type="button" value="11:59:00 PM"/>

app-demo-201106152713 | Scale out (App Service plan) 

App Service

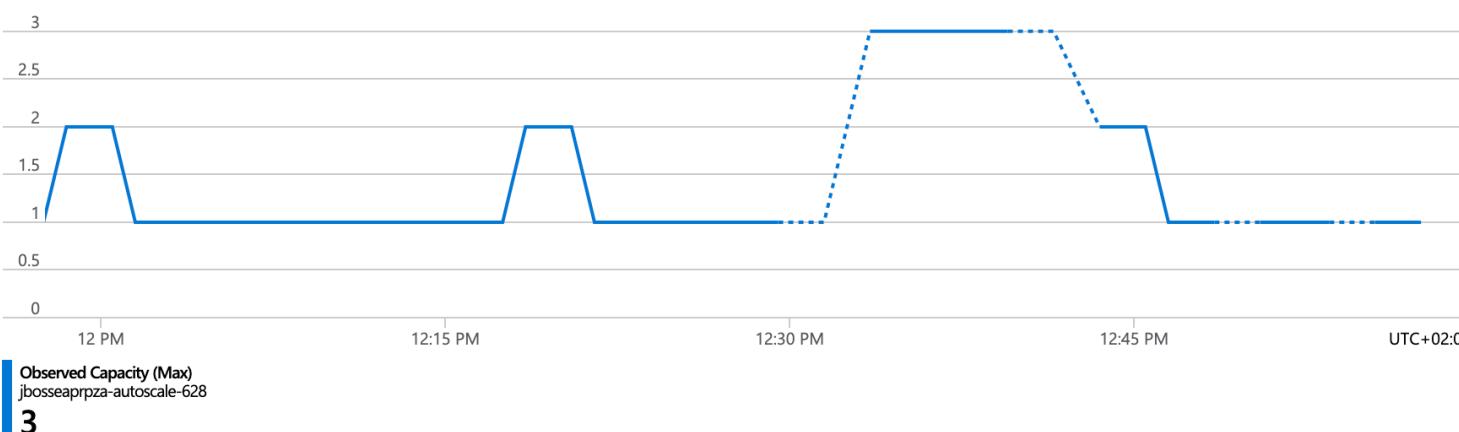
Save Discard Refresh Logs Provide feedback

Configure Run history JSON Notify Diagnostics settings

i Observed instance count - this chart plots the instance count as observed by the auto scale engine. If the chart is empty it either means auto scale is in period of time or auto scale was not configured.

Show data for last

1 hour 6 hours 12 hours 1 day 7 days Custom  Pin to dashboard



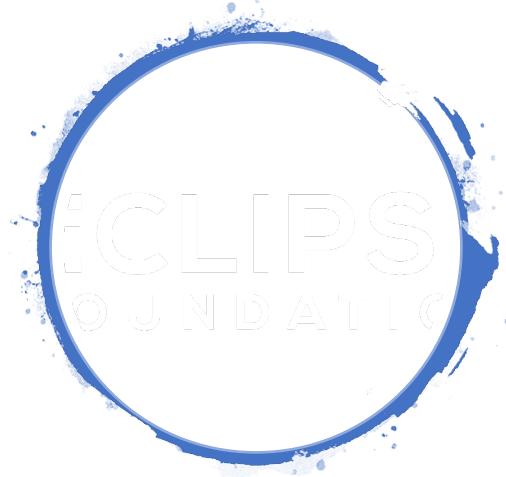
Observed Capacity (Max)
jbosssearpz-autoscale-628
3

Autoscale events for this time range [View more details in the Activity Log](#)

Operation name	Status	Time	Time stamp
> i Autoscale scale down completed	Succeeded	10 minutes ...	Mon Nov 09 2020 12:45:02 ...
i Flapping	Succeeded	11 minutes ...	Mon Nov 09 2020 12:43:59 ...
i Flapping	Succeeded	12 minutes ...	Mon Nov 09 2020 12:42:59 ...
i Flapping	Succeeded	12 minutes ...	Mon Nov 09 2020 12:42:15 ...
> i Autoscale scale down completed	Succeeded	14 minutes ...	Mon Nov 09 2020 12:40:04 ...
> i Autoscale scale up completed	Succeeded	22 minutes ...	Mon Nov 09 2020 12:32:11 ...

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



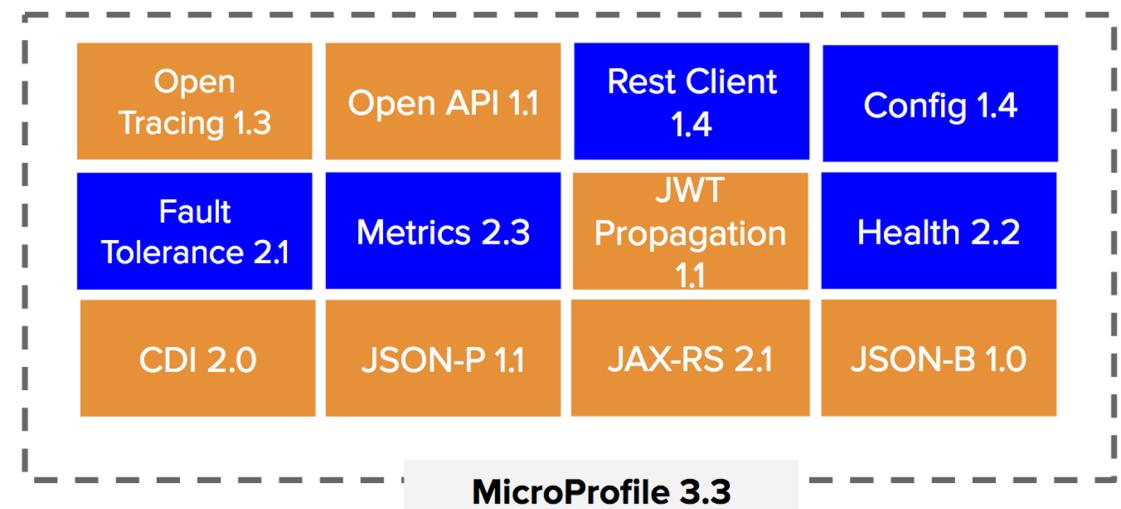
Eclipse MicroProfile 3.3 Released!

On February 18, 2020, MicroProfile 3.3 was released.

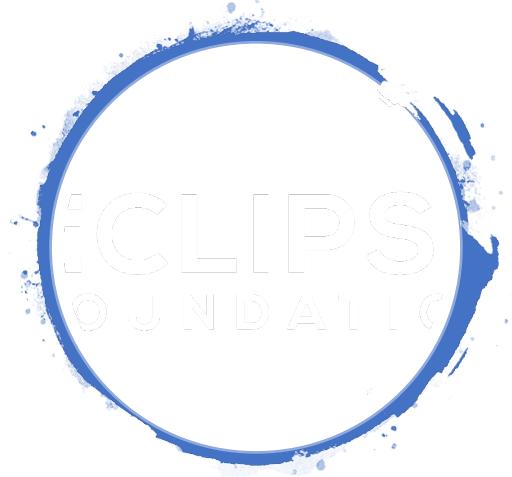
Offered in the release:

Tooling

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



Legend:
■ = New
■ = Updated
■ = No change from last release (MicroProfile 3.2)



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

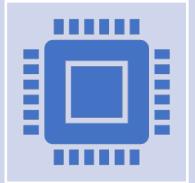
- Removes **Challenges** of Tracing
- **Standard** for instrumenting microservices for distributed tracing
 - Jaeger, Zipkin
- Enterprise Java binding to the Open Tracing specification



Open Tracing Example

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

JWT Propagation



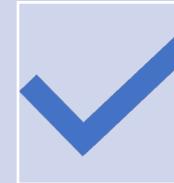
Security for Stateless REST Services



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



Think of it as **SAML** for REST



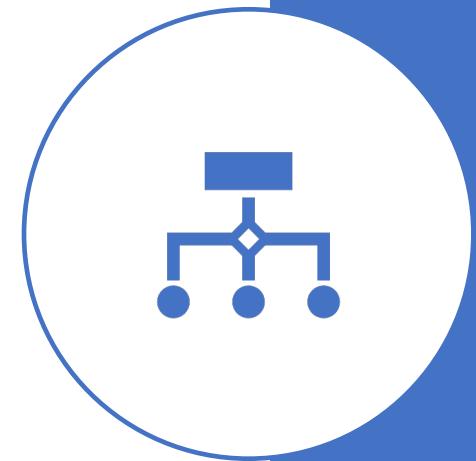
Enterprise **Java binding** for the JWT specification

JWT Propagation Example

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

Configuration

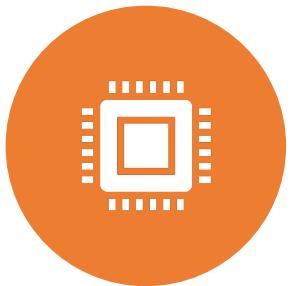
- Java EE mostly focuses on **static** configuration
- Applications may need to be configured based on a **running** environment
- **MicroProfile Config** allows modification of configuration values without repackaged
- Influenced by [DeltaSpike Config](#) and [Apache Tamaya](#)



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



Like JAX-WS/SOAP clients



Several JAX-RS implementations already exist

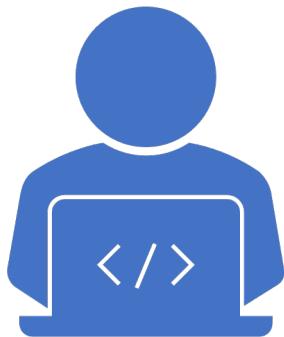


MicroProfile standardizes this capability

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)
```

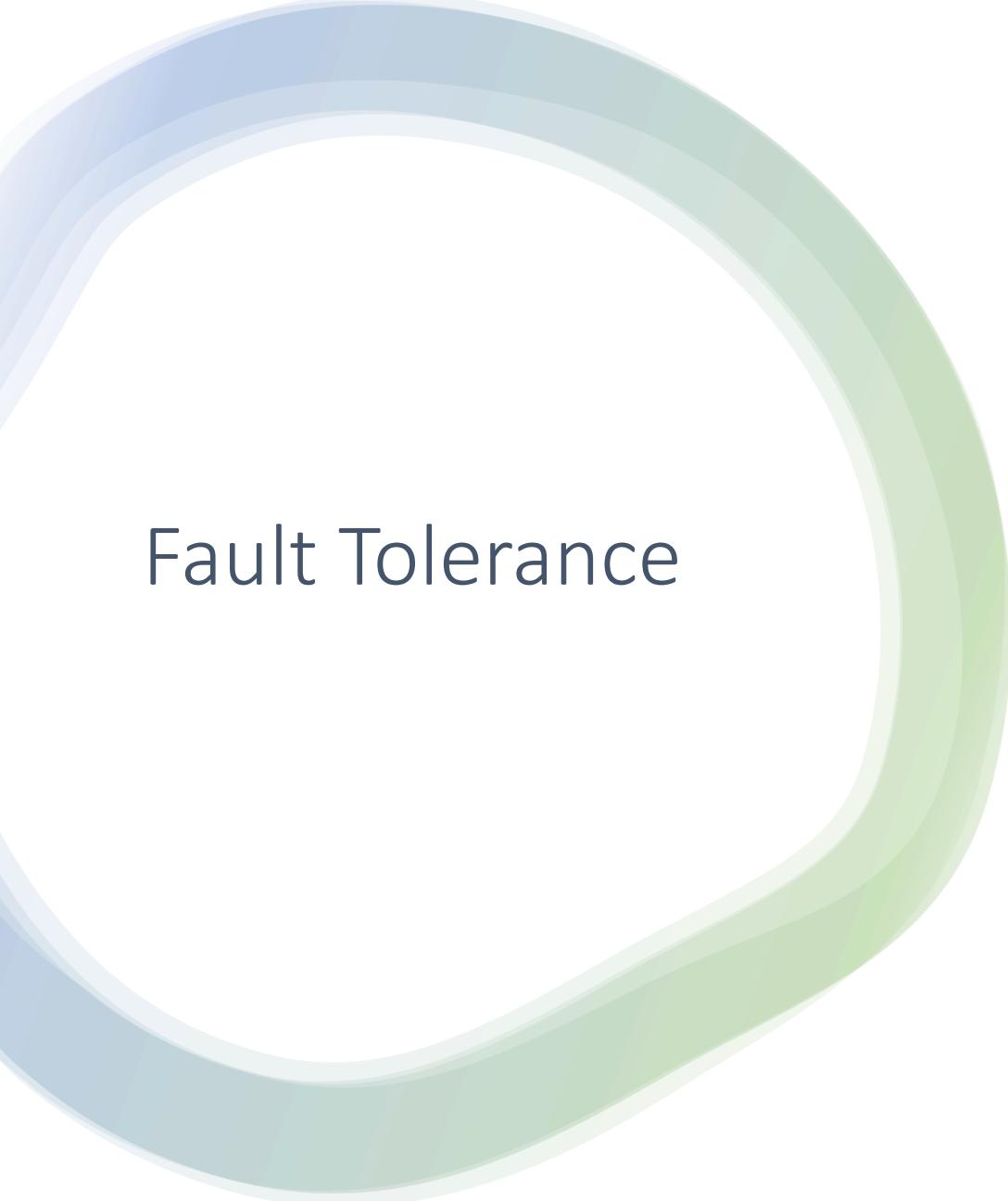
```
public List<Membership> getAllMemberships() {
```

```
    ...
```

```
@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(  
    @NotNull @PathParam(value = "id") int id) {
```

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
```

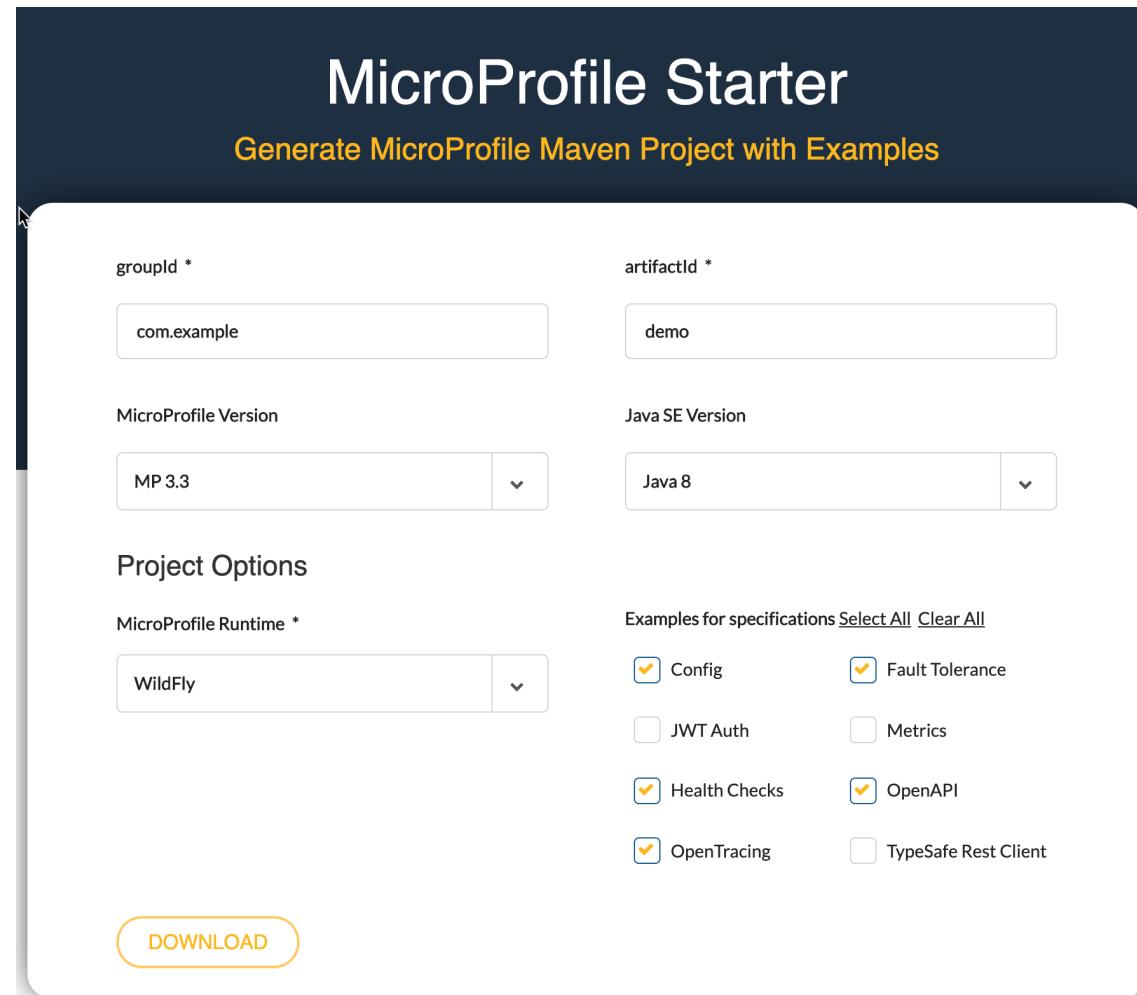
```
'https://start.microprofile.io/api/project?supportedServer=WILDFLY&groupId=com.example&artifactId=jboss&mpVersion=MP33&javaSEVersion=SE8&selectedSpecs=CONFIG&selectedSpecs=FAULT_TOLERANCE&selectedSpecs=HEALTH_CHECKS&selectedSpecs=OPEN_API&selectedSpecs=OPEN_TRACING'
```

- Unzip jboss.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 jboss-XXXXXX -g jboss-XXXXX-rg

3. Browse to “<https://xxxxxxxx.azurewebsites.net>”



<https://start.microprofile.io>

Slides and Workshop (Postgres + JBoss + Azure)

<https://aka.ms/jboss-azure>

