WEBINAR

Modernizing Java Apps
with AWS Lambda

James Eastham

Senior Cloud Architect
AWS

Jon Bevan

Engineering Team Lead
Script Runner

Serverless is more than compute

Compute

AWS Lambda

AWS Fargate

Data stores

Amazon Simple Storage Service (Amazon S3)

Amazon Aurora Serverless

Amazon DynamoDB

Integration

Amazon API Gateway

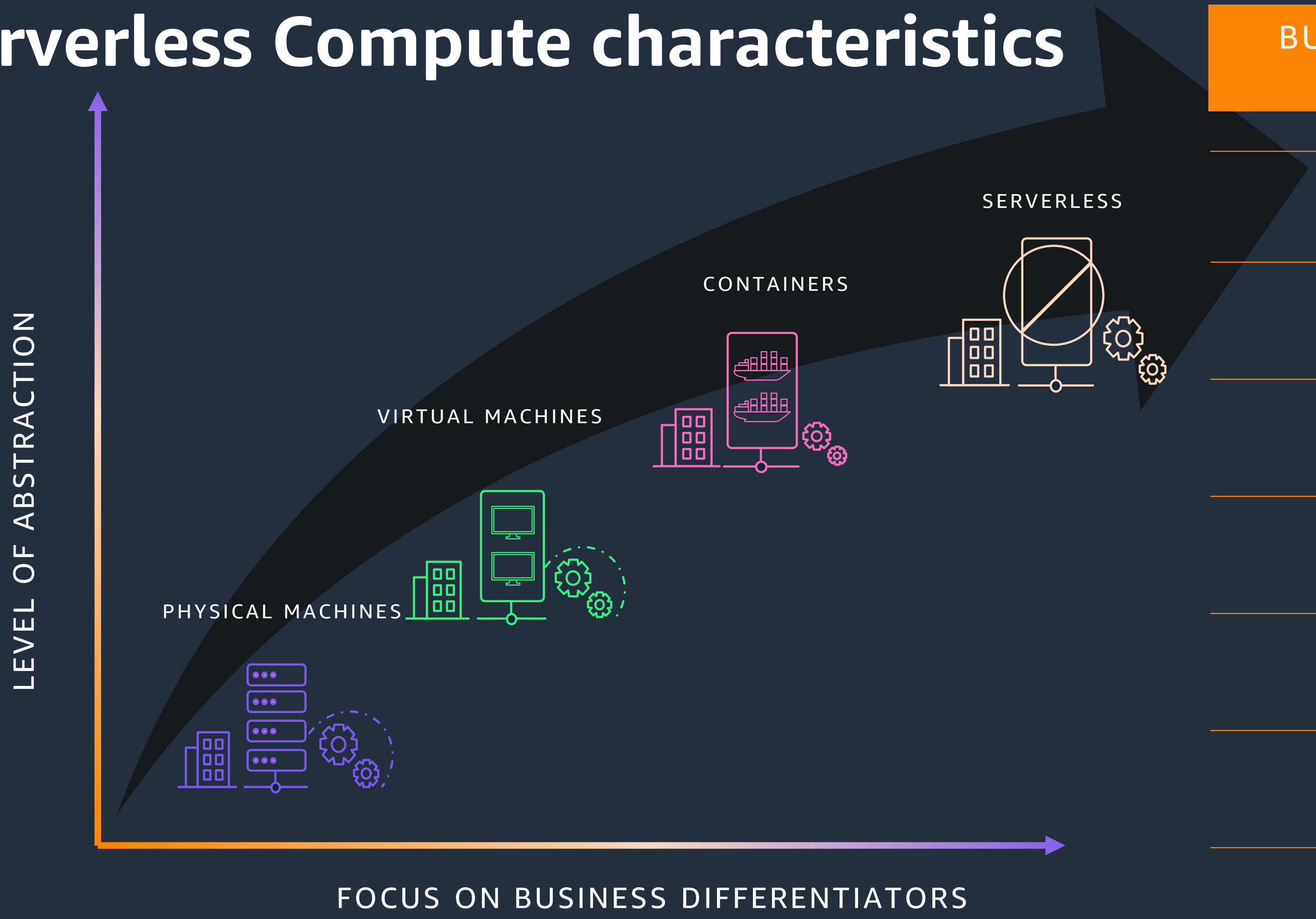
Amazon SQS

Amazon SNS

AWS Step Functions

AWS AppSync

Serverless Compute characteristics



BUILT A CULTURE OF INNOVATION

- Application
- Orchestration
- Storage & Data
- Networking
- Compute
- Infrastructure

Decision heuristics

CONTAINERS

Amazon Elastic Container Service (Amazon ECS)

AWS Fargate

Amazon Elastic Kubernetes Service (Amazon EKS)

AWS App Runner

When you need . . .

- Support for long-running compute jobs (> 15 minutes)
- Predictable scaling OR longer start times are acceptable
- Application with a non-HTTP/S listener
- Low single-digit millisecond response times to your requests

When you want . . .

- Managed container environment
- Management and control

Decision heuristics

AWS LAMBDA

AWS Lambda

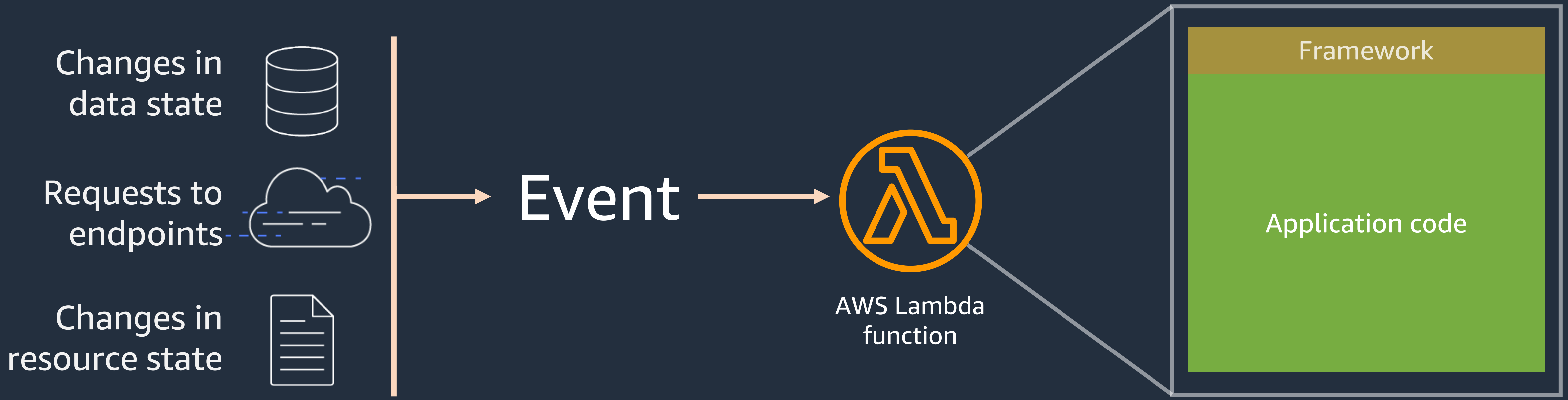
When you need . . .

- To trigger action on an event
- Ability to handle unknown demand
- Lighter-weight, application-focused stateless computing

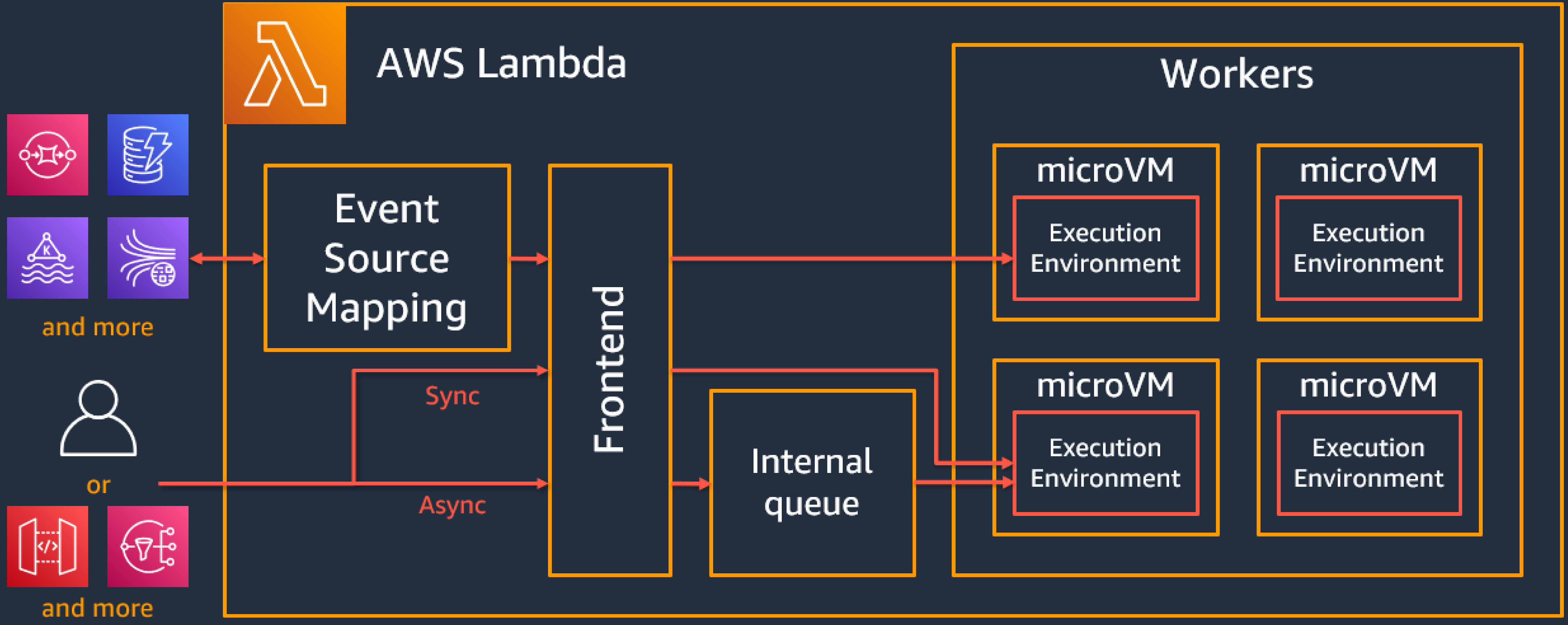
When you want . . .

- Focus on business logic and not IT operations
- Enable simplified IT automation
- Enable real-time data processing
- To hand operational complexity to AWS

Serverless Architecture



AWS Lambda under the hood



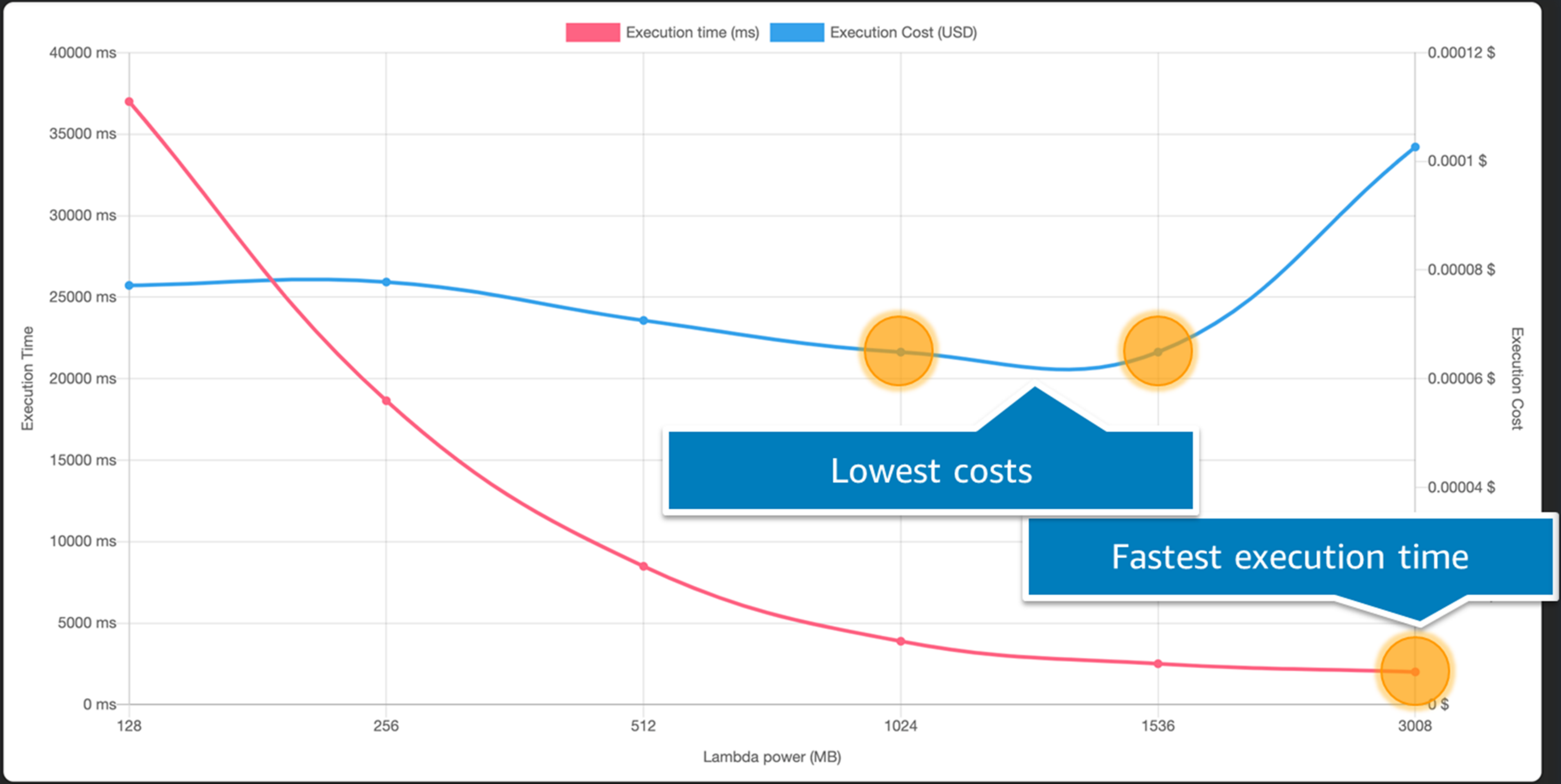
Resource allocation

Resource Allocation

MEMORY CONFIGURATION

Lambda allocates CPU power in proportion to the amount of memory configured.

Memory	vCPU share
128 MB	< 1 vCPU
1,769 MB	1 vCPU
10,240 MB	6 vCPU


Execution time (ms)
Execution Cost (USD)
40000 ms
35000 ms
30000 ms
25000 ms
20000 ms
15000 ms
10000 ms
5000 ms
0 ms
Execution Time
0.00012 $
0.0001 $
0.00008 $
0.00006 $
0.00004 $
0 $
Execution Cost
128
256
512
1024
1536
3008
Lambda power (MB)
Lowest costs
Fastest execution time

What does that mean for your Java application?

Java on Lambda

Micronaut

SpringBoot

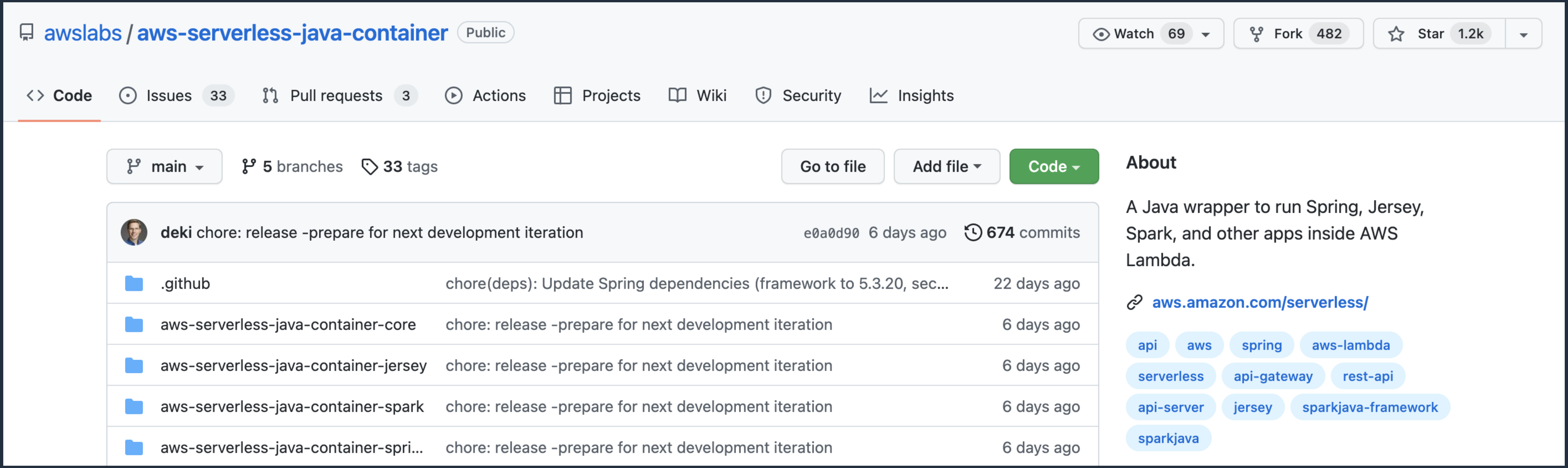
Quarkus

Kotlin

Groovy

Spring Boot on Lambda

AWS Serverless Java Container



awslabs / aws-serverless-java-container Public

Watch 69 | Fork 482 | Star 1.2k

Code | Issues 33 | Pull requests 3 | Actions | Projects | Wiki | Security | Insights

main | 5 branches | 33 tags | Go to file | Add file | Code

deki chore: release -prepare for next development iteration e0a0d90 6 days ago 674 commits

.github	chore(deps): Update Spring dependencies (framework to 5.3.20, sec...	22 days ago
aws-serverless-java-container-core	chore: release -prepare for next development iteration	6 days ago
aws-serverless-java-container-jersey	chore: release -prepare for next development iteration	6 days ago
aws-serverless-java-container-spark	chore: release -prepare for next development iteration	6 days ago
aws-serverless-java-container-spri...	chore: release -prepare for next development iteration	6 days ago

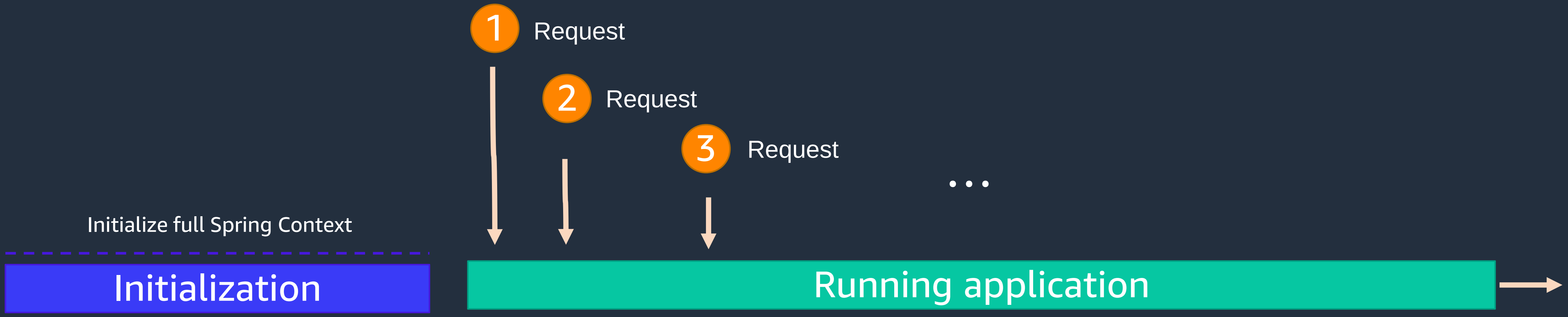
About

A Java wrapper to run Spring, Jersey, Spark, and other apps inside AWS Lambda.

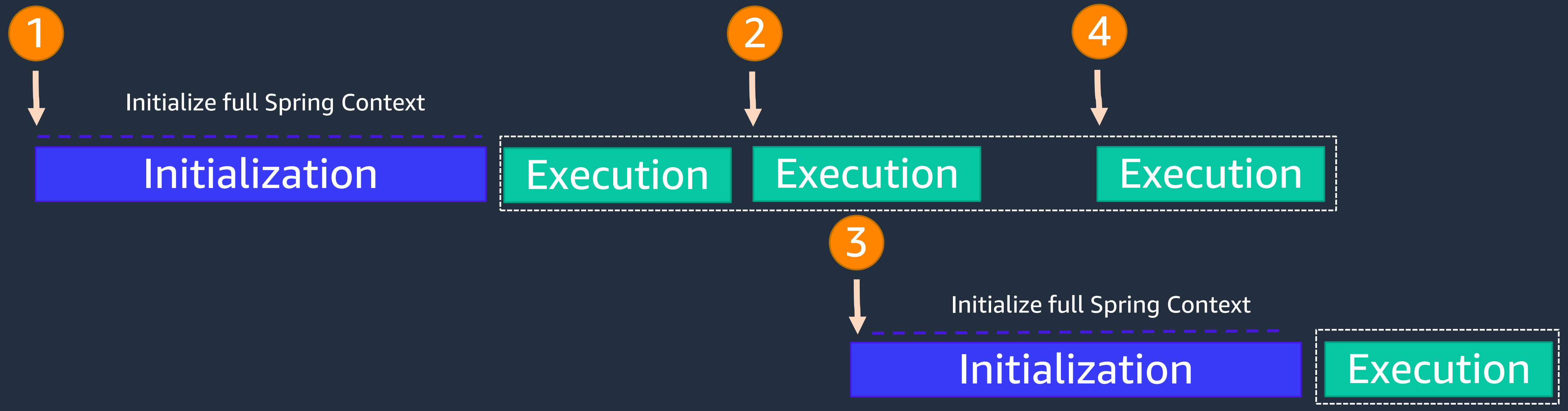
aws.amazon.com/serverless/

api aws spring aws-lambda serverless api-gateway rest-api api-server jersey sparkjava-framework sparkjava

Spring Boot on a server

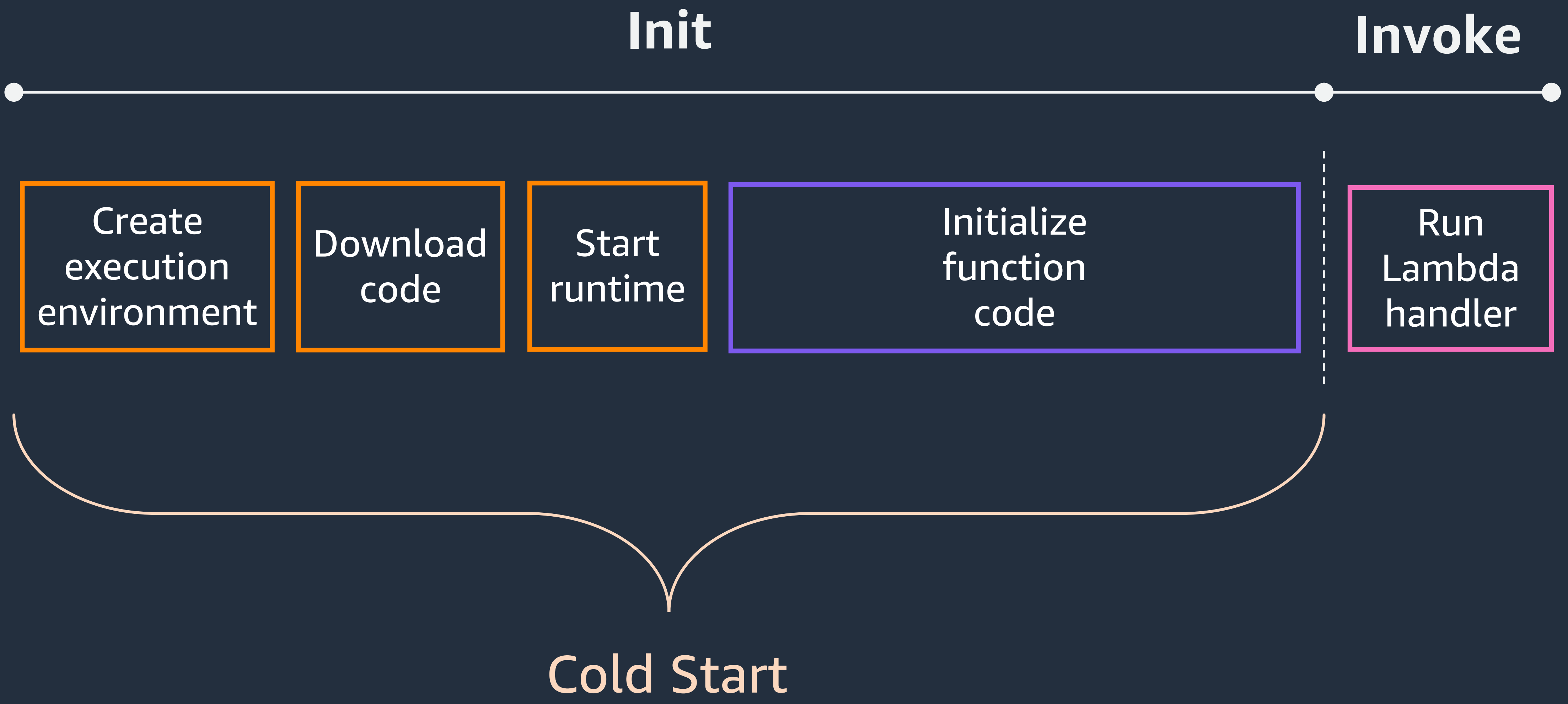


Spring Boot on AWS Lambda



Cold starts!

Cold Starts

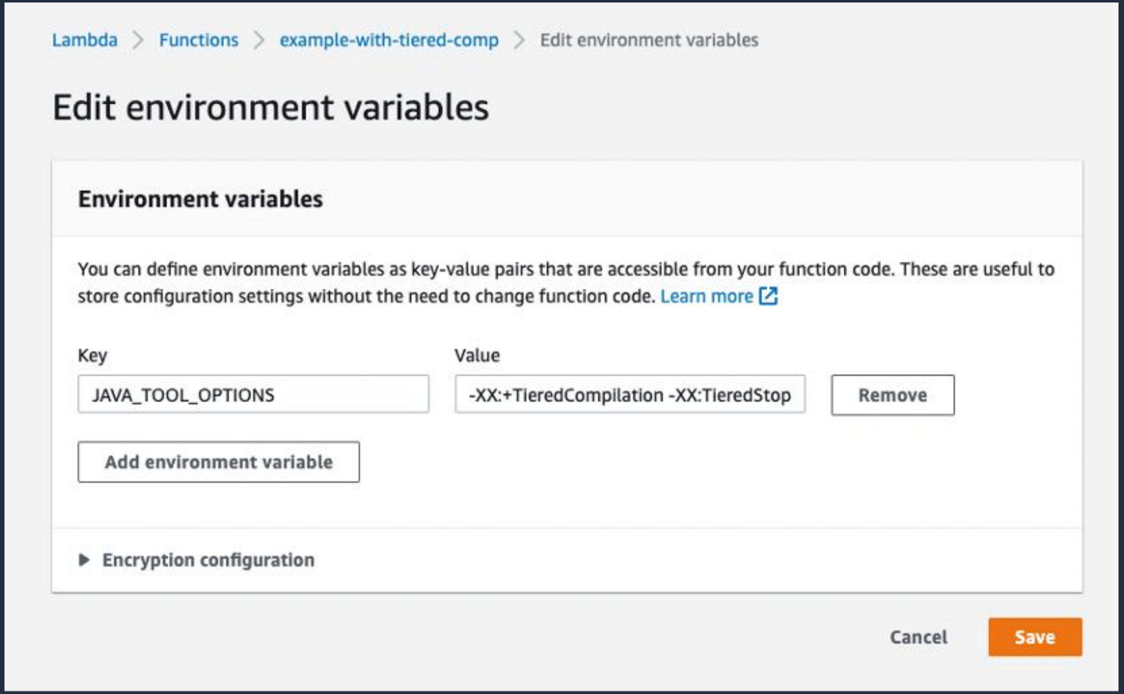

Init
Invoke
Create execution environment
Download code
Start runtime
Initialize function code
Run Lambda handler
Cold Start

Optimizations

Uber Jars containing all resources

Tiered compilation

- Use optimized C1 compiler for faster application start up
- Add an environment variable to change the compiler level



Lambda > Functions > example-with-tiered-comp > Edit environment variables

Edit environment variables

Environment variables

You can define environment variables as key-value pairs that are accessible from your function code. These are useful to store configuration settings without the need to change function code. Learn more

Key	Value
JAVA_TOOL_OPTIONS	-XX:+TieredCompilation -XX:TieredStop

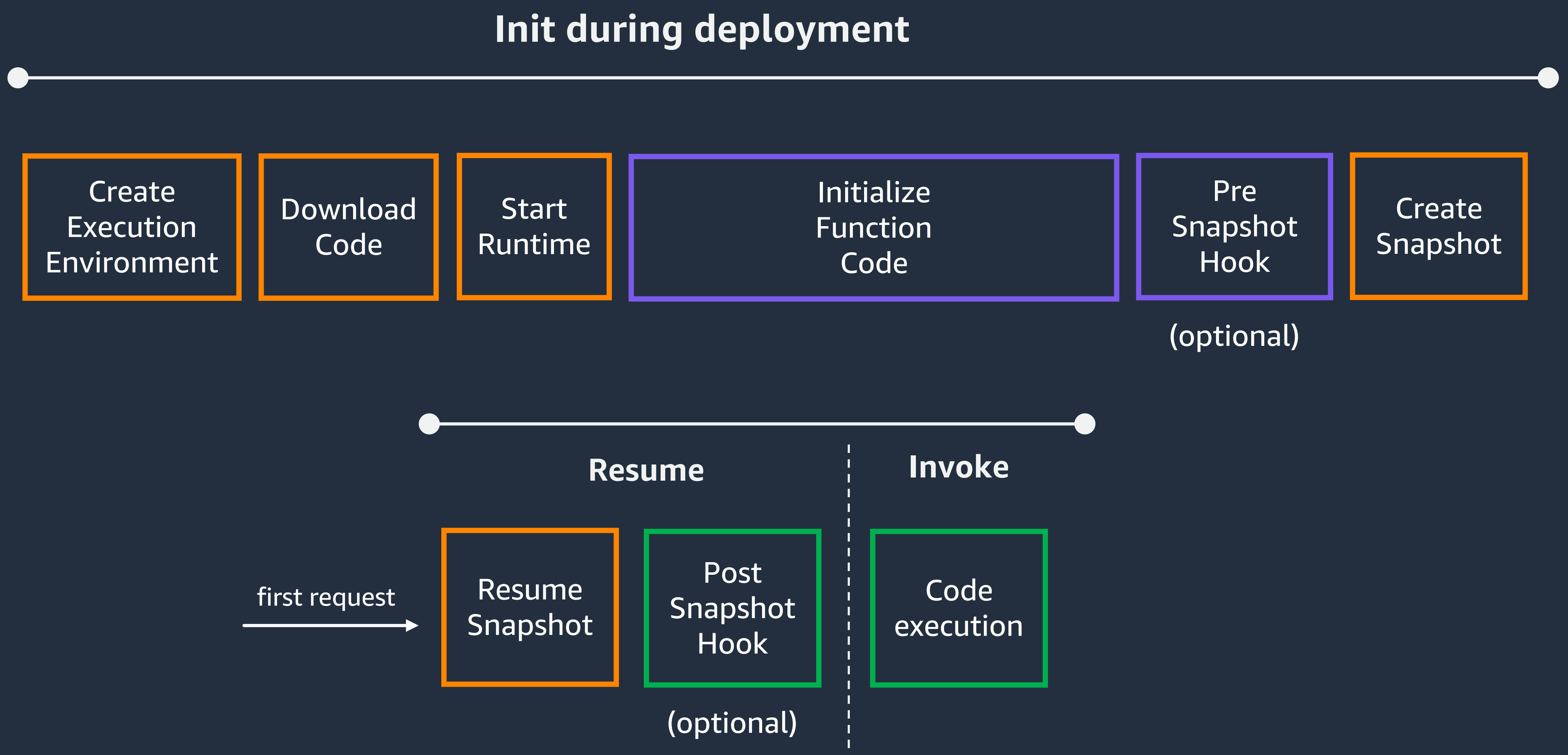
	Full Duration	
Without Tiered	11.378	
With Tiered	6.725	- 40 %

Results will vary based on your application

AWS Lambda SnapStart

Up to 10x faster start-up performance

How SnapStart works



Example with hooks

```
import org.crac.Core;
import org.crac.Resource;

public class HelloHandler implements RequestHandler<String, String>, Resource {

    public HelloHandler() {
        Core.getGlobalContext().register(this);
    }

    public String handleRequest(String name, Context context) {
        return "Hello " + name;
    }

    public void beforeCheckpoint() {
        System.out.println("Before Snapshot");
    }

    public void afterRestore() {
        System.out.println("After Resume");
    }
}
```

Configuring SnapStart

```yaml
Transform: AWS::Serverless-2016-10-31
Description: Lambda Function with SnapStart and versions.

Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: <code-location>
      Handler: <handler>
      Runtime: java11
      AutoPublishAlias: live
      SnapStart:
        ApplyOn: PublishedVersions
```

Provisioned Concurrency

Reduction in Lambda function cold starts. Avoid burst throttling. Save costs on workloads with consistent traffic.

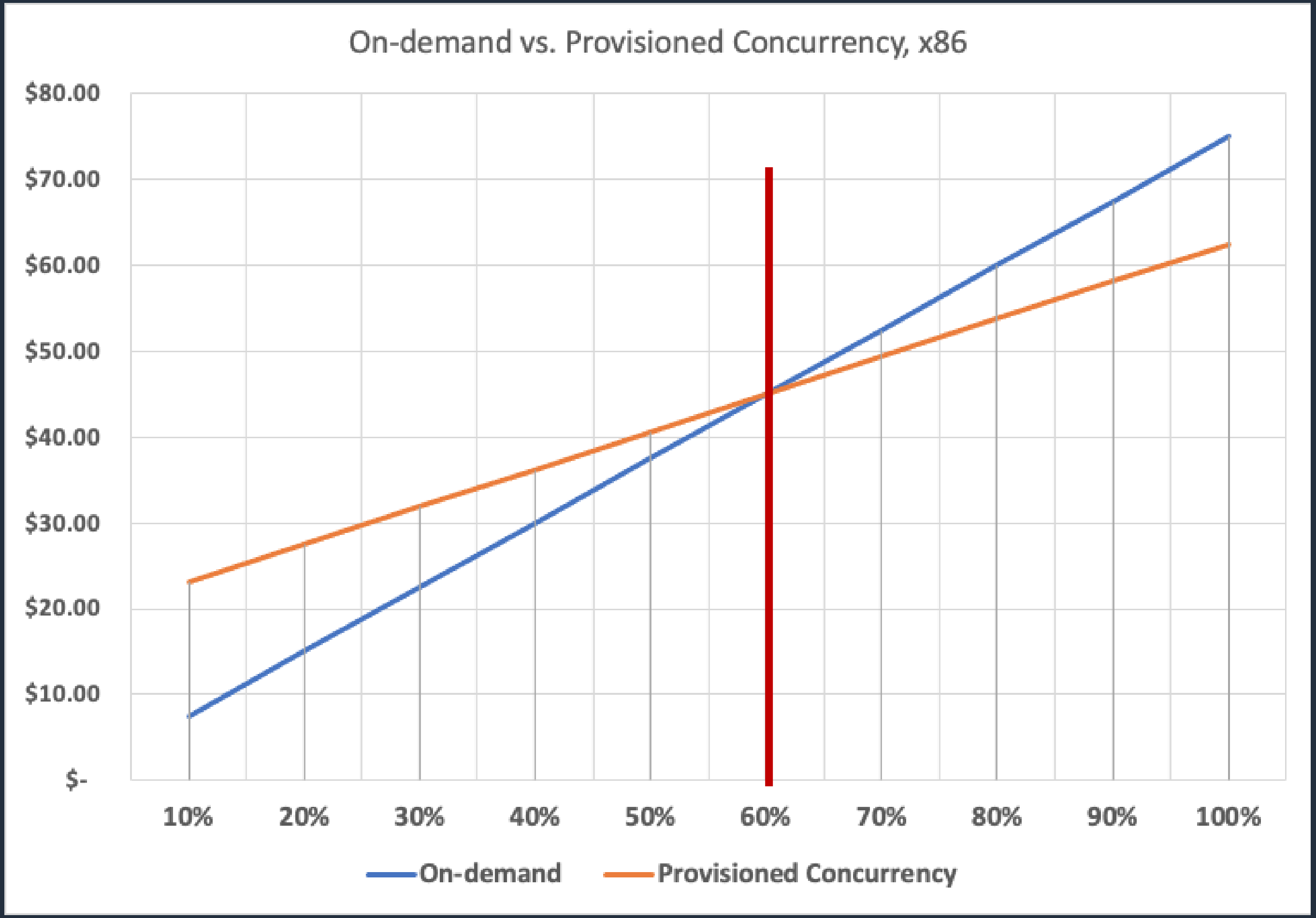
Provisioned Concurrency

	On-demand pricing (us-east-1)
Requests	$0.20 per 1M requests
Invocation Duration	$0.0000166667 per GB-second

	Provisioned Concurrency pricing (us-east-1)
Requests	$0.20 per 1M requests
Provisioned Concurrency	$0.0000041667 per GB-second
Invocation Duration	$0.0000097222 per GB-second

Total: $0.0000138889
(~16% cheaper than on-demand if fully utilized)

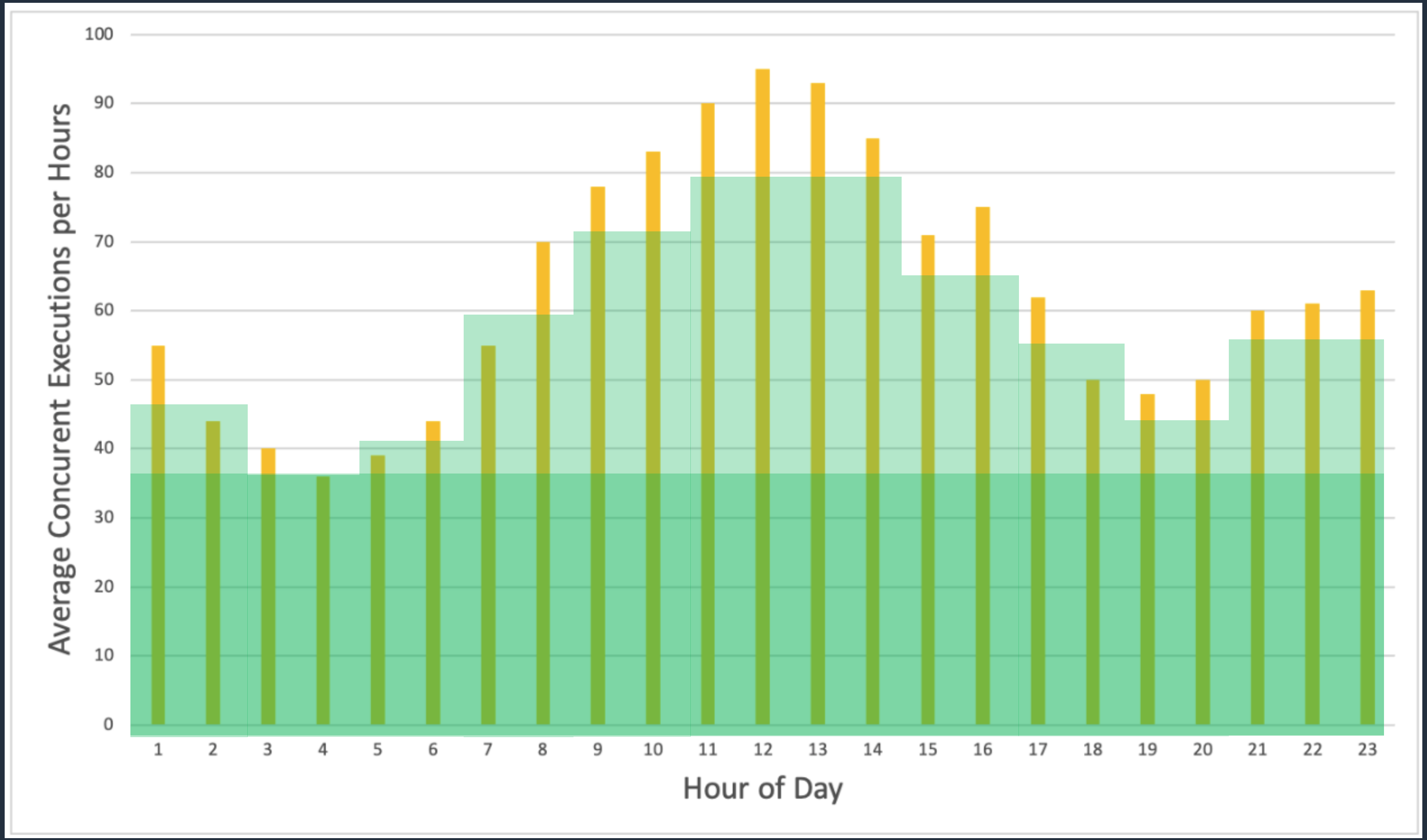
Provisioned Concurrency



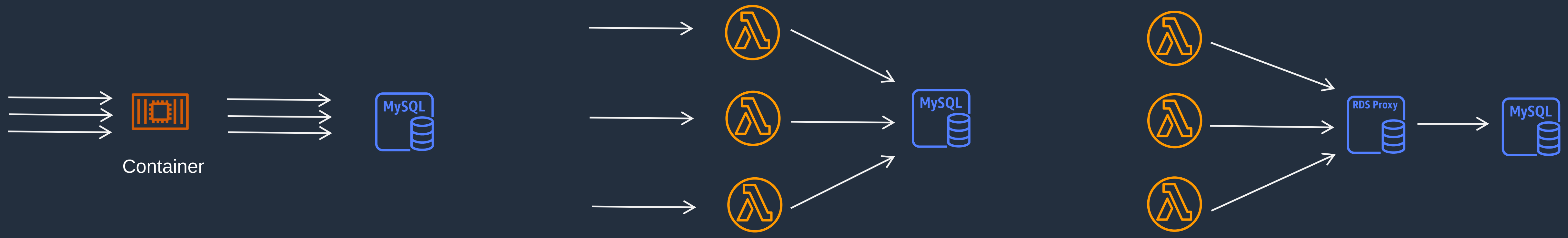
(us-east-1. Pricing differs per region, different regions will have different thresholds)

Provisioned Concurrency

1. Analyze concurrent executions patterns
2. Start with static Provisioned Concurrency
3. Evolve to dynamic Provisioned Concurrency with application auto-scaling



Database connection pooling



The Adaptavist Group

The Adaptavist Group

ScriptRunner

Stretch Jira, Confluence and Bitbucket beyond their limits; for teams of any size, in any industry.

New and unlimited possibilities for automation, customisation and integration

ScriptRunner

How ScriptRunner Cloud works

Atlassian Cloud

An administrator writes a script

End-users make a change in Jira or Confluence

Atlassian sends us a **webhook**

ScriptRunner Cloud

We process the webhook using Java services running on **Fargate**, the common SNS and SQS fanout, and some **Typescript** Lambda functions

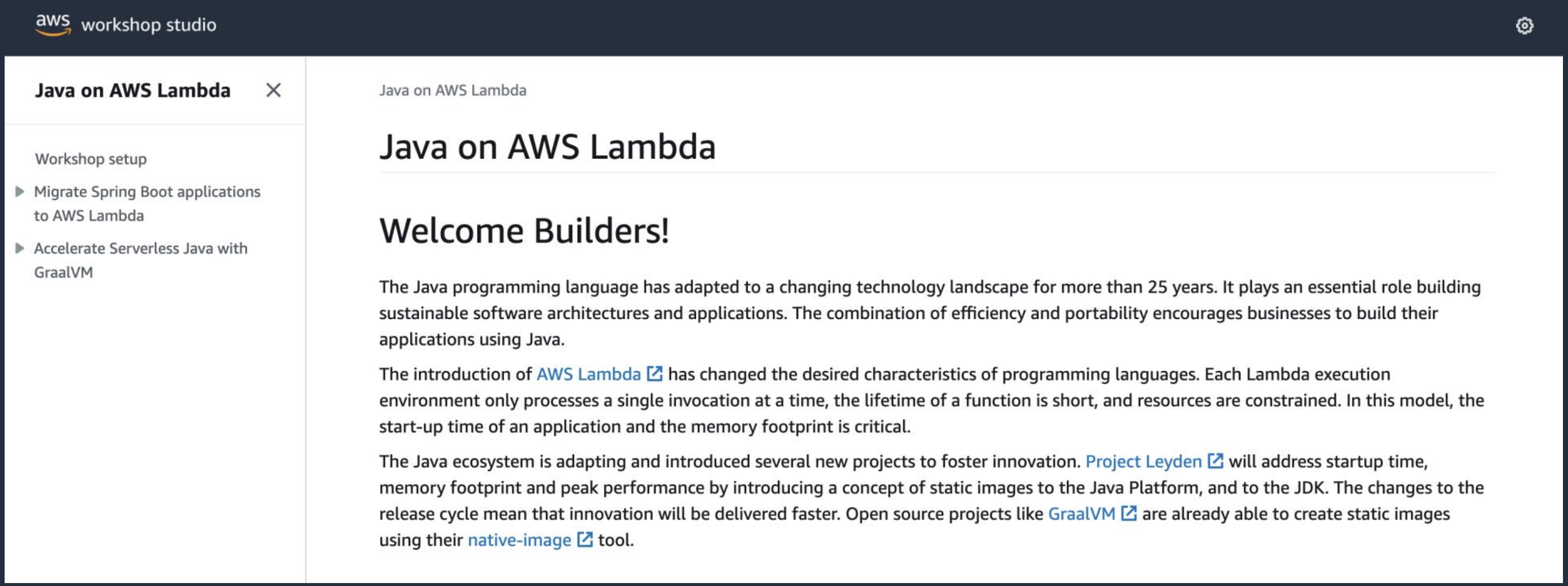
AWS Lambda

We compile and run the customer's Groovy script in an **isolated** Lambda function running on Java

ScriptRunner

The Script Runner Journey

Deep Dive: Java on AWS Lambda Workshop


aws workshop studio
Java on AWS Lambda
Workshop setup
Migrate Spring Boot applications to AWS Lambda
Accelerate Serverless Java with GraalVM

Java on AWS Lambda

Java on AWS Lambda

Welcome Builders!

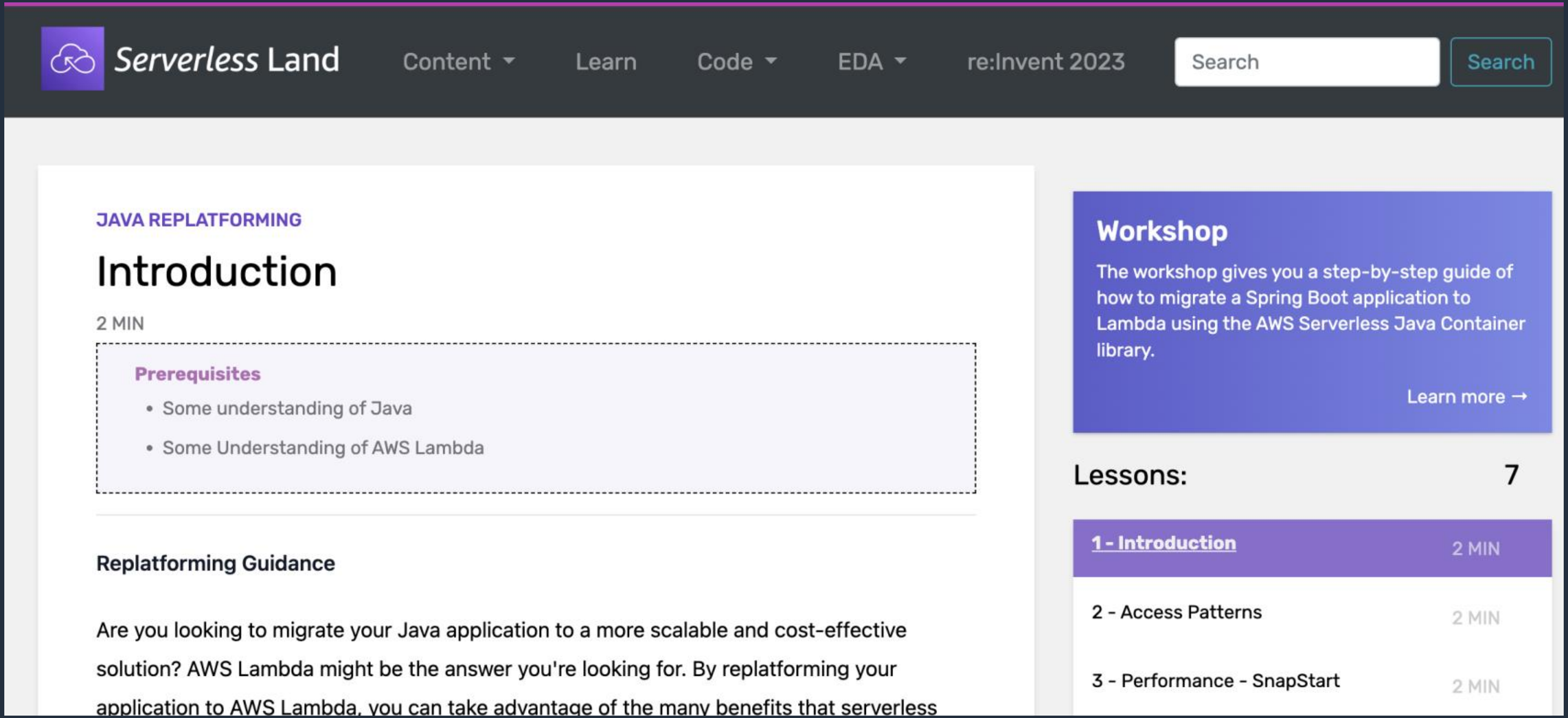
The Java programming language has adapted to a changing technology landscape for more than 25 years. It plays an essential role building sustainable software architectures and applications. The combination of efficiency and portability encourages businesses to build their applications using Java.

The introduction of AWS Lambda has changed the desired characteristics of programming languages. Each Lambda execution environment only processes a single invocation at a time, the lifetime of a function is short, and resources are constrained. In this model, the start-up time of an application and the memory footprint is critical.

The Java ecosystem is adapting and introduced several new projects to foster innovation. Project Leyden will address startup time, memory footprint and peak performance by introducing a concept of static images to the Java Platform, and to the JDK. The changes to the release cycle mean that innovation will be delivered faster. Open source projects like GraalVM are already able to create static images using their native-image tool.

https://catalog.workshops.aws/java-on-aws-lambda

Serverless Java Replatforming Guide



JAVA REPLATFORMING

Introduction

2 MIN

Prerequisites

- Some understanding of Java
- Some Understanding of AWS Lambda

Replatforming Guidance

Are you looking to migrate your Java application to a more scalable and cost-effective solution? AWS Lambda might be the answer you're looking for. By replatforming your application to AWS Lambda, you can take advantage of the many benefits that serverless

Workshop

The workshop gives you a step-by-step guide of how to migrate a Spring Boot application to Lambda using the AWS Serverless Java Container library.

Learn more →

Lessons: 7

1 - Introduction	2 MIN
2 - Access Patterns	2 MIN
3 - Performance - SnapStart	2 MIN

Thank you!

James Eastham
jamesuk@amazon.com
@plantpowerjames

Jon Bevan
jbevan@adaptavist.com