
AWS
re:Invent

SVS317

Observability, logging, and more with AWS Lambda extensions

Julian Wood

Senior Developer Advocate,
AWS Serverless

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What I am covering today

Introducing AWS Lambda extensions

How do extensions work

Using an extension

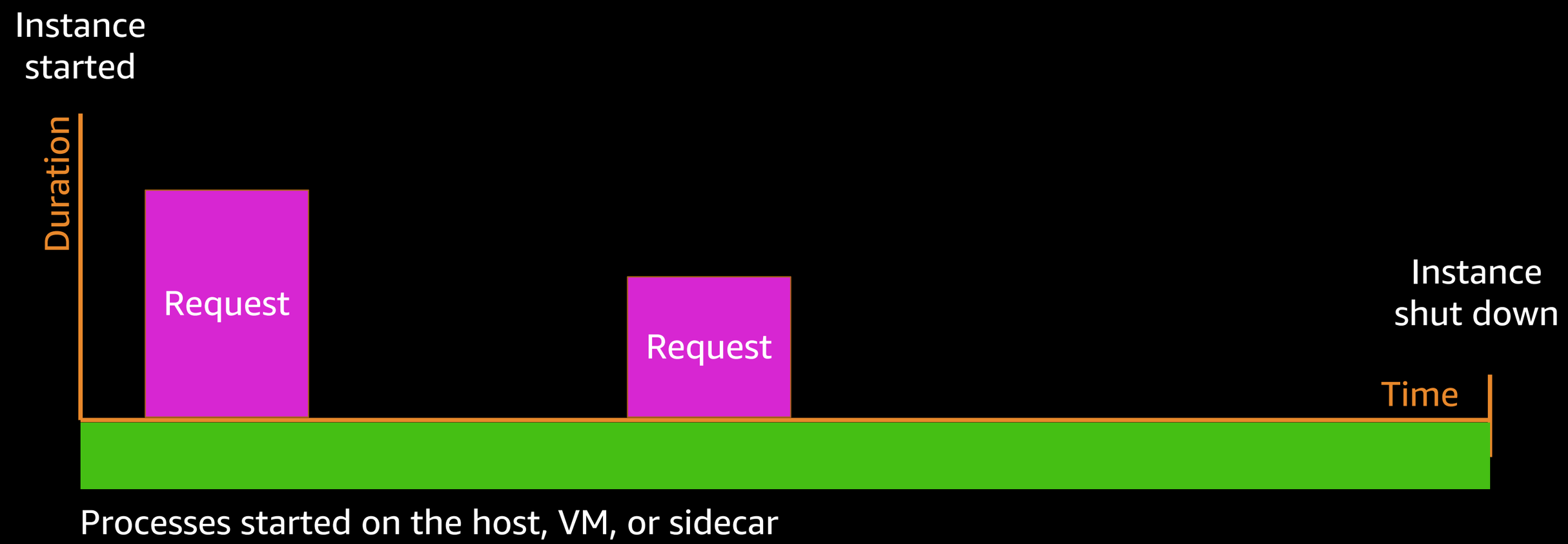
The Lambda lifecycle

Using extensions in container images

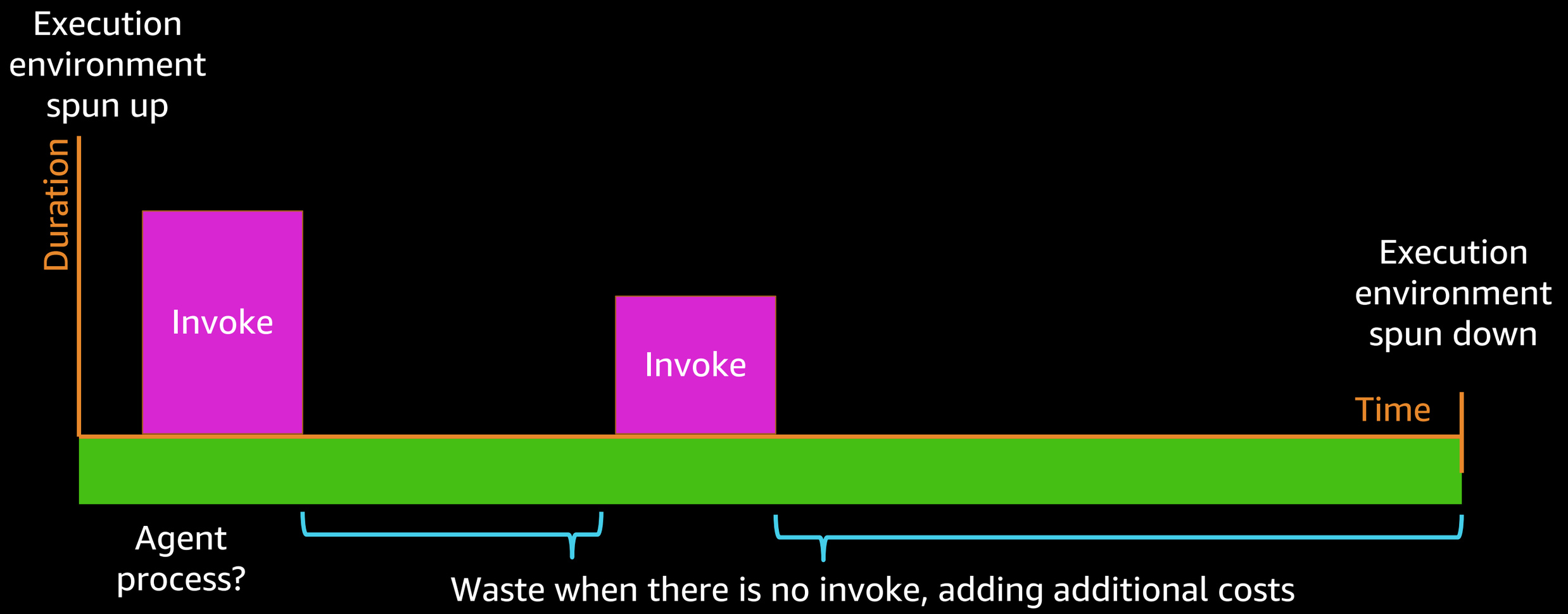


© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Traditional compute



Lambda invocation lifecycle



Introducing Lambda extensions

A new way to easily integrate Lambda with your favorite monitoring, observability, security, and governance tools



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What are Lambda extensions?

A NEW WAY TO INTEGRATE WITH THE LAMBDA EXECUTION ENVIRONMENT

Use Lambda extensions to augment your Lambda functions

Previously, long-running processes, could not easily run on Lambda

Easier way to plug in your existing tools with Lambda

Tools can integrate deeply into the Lambda environment

No complex installation, configuration, or operational overhead

Helps reduce function code ☺

Extensions use cases

A NEW WAY TO INTEGRATE WITH THE LAMBDA EXECUTION ENVIRONMENT

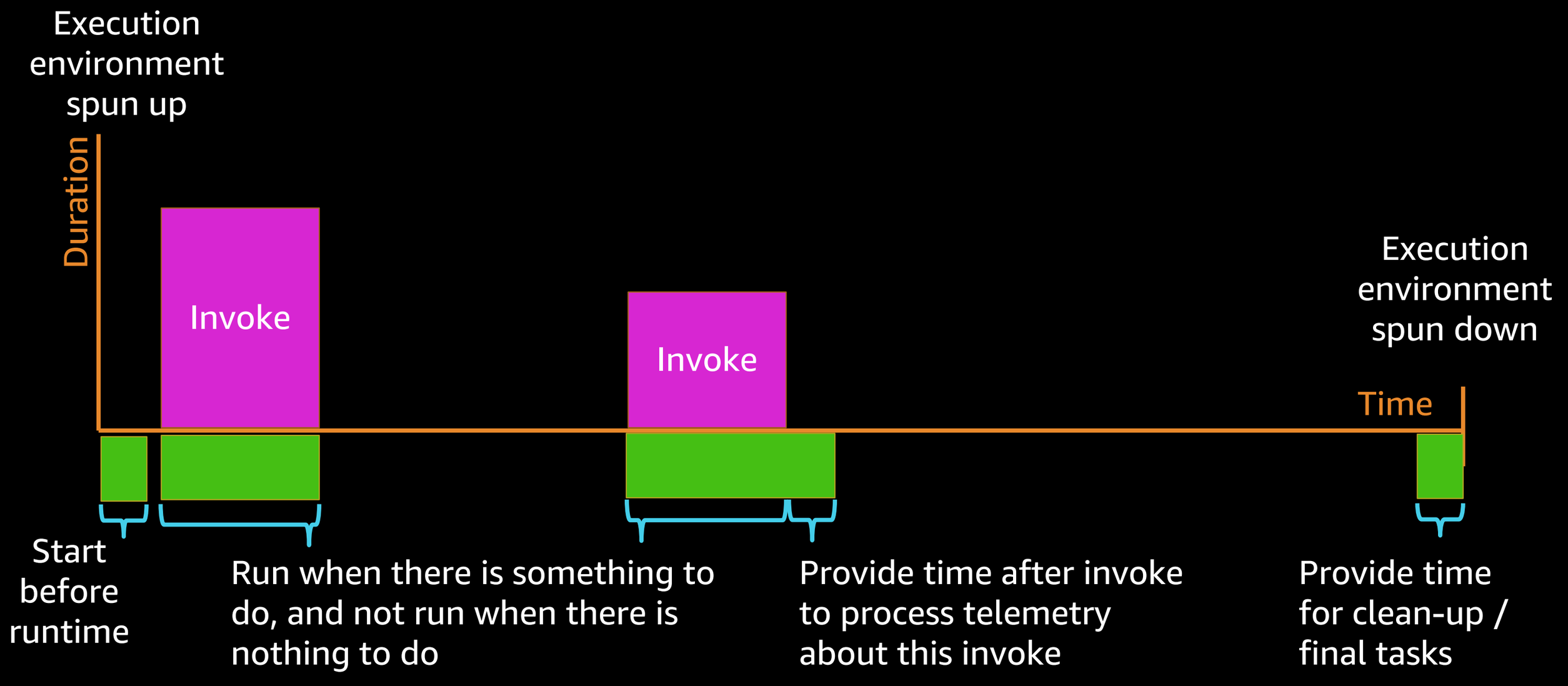
Capture diagnostic information before, during, and after function invocation

Automatically instrument your code without needing code changes

Fetch configuration settings or secrets before the function invocation

Detect and alert on function activity through hardened security agents

Extending Lambda's invocation lifecycle



Extensions available today

AWS LAMBDA READY PARTNER, AWS PARTNER, AND AWS TOOLS USING LAMBDA EXTENSIONS





















Amazon CloudWatch
Lambda Insights



AWS AppConfig











How do extensions work



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Understanding Lambda extensions

Use extensions

- Consume capabilities of an extension by configuring the options in your Lambda functions



https://s12d.com/extensions-intro

Extensions API

- Build your own extension
- Provides a way for tools to get greater control during function initialization, invocation, and shutdown



https://s12d.com/extensions-building

Logs API

- Consume platform and function logs directly from Lambda



https://s12d.com/extensions-logs

Most customers use extensions without needing to know about the capabilities of the Extensions API that enables them

How to deploy extensions

Deploy extensions as Lambda layers

- Using AWS Management Console, AWS CLI, AWS CloudFormation, AWS SAM, Serverless Framework, Terraform, Stackery

Include extensions in Lambda container image

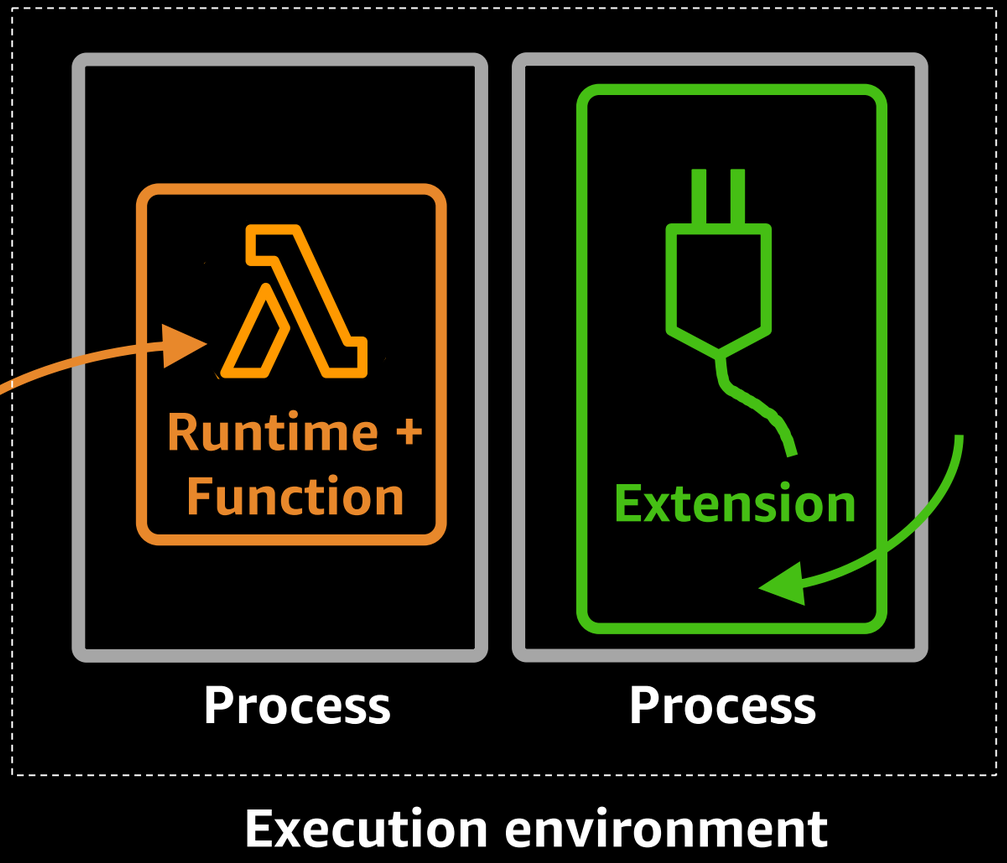
- Build image using Docker CLI/AWS SAM with a Dockerfile

Bootstrap

- Lambda starts any extensions found in the `/opt/extensions/` directory

Internal and external extensions

Internal extensions run as part of the runtime process, in-process with your code



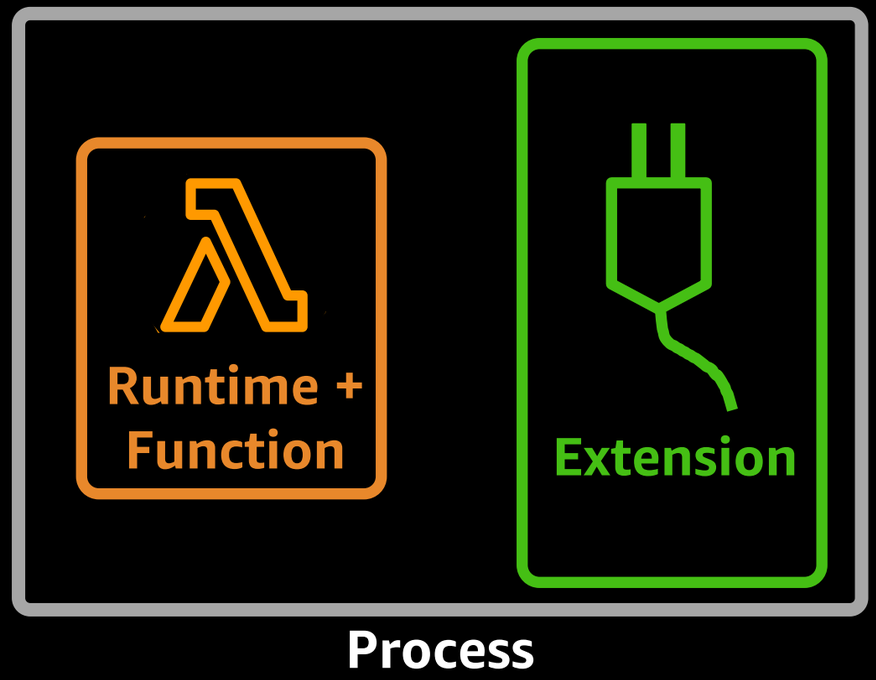
External extensions run as separate processes from the runtime, but still within the same execution environment as the Lambda function

Internal extensions

Extensions that run within the runtime, in-process with your code as a separate thread

Modify runtime process startup:

Language-specific environment variables
- JAVA_TOOL_OPTIONS (Java Corretto 8 and 11)
- NODE_OPTIONS (Node.js 10 and 12)
- DOTNET_STARTUP_HOOKS (.NET Core 3.1)



Set via Lambda environment variable:

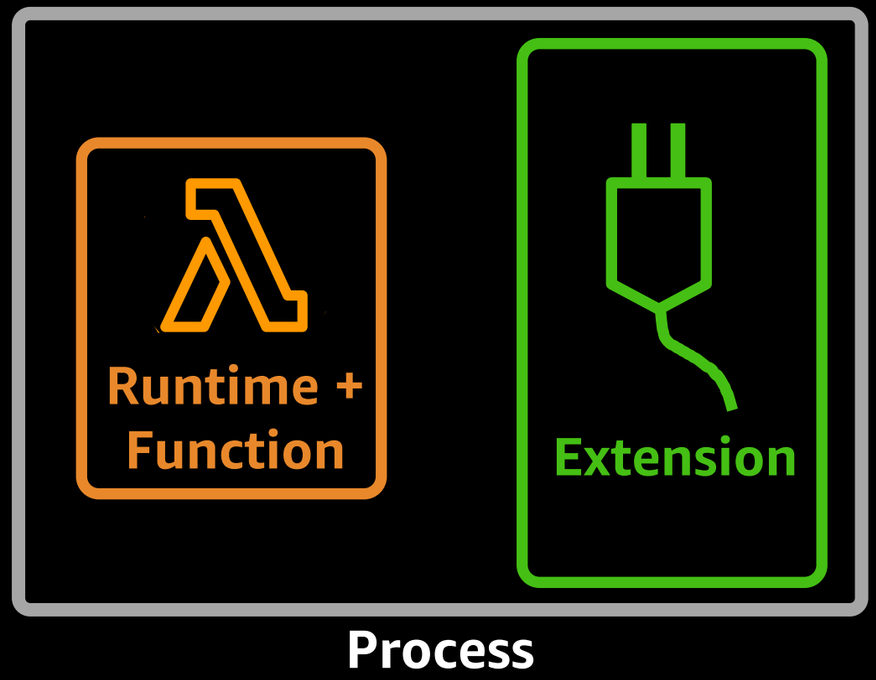
```
JAVA_TOOL_OPTIONS = -javaagent:"/opt/ExampleAgent-0.0.jar"
```

Internal extensions

Extensions that run within the runtime, in-process with your code as a separate thread

Modify runtime process startup:

Wrapper scripts
Delegate runtime startup to a script
Parameters that cannot be set through environment variables
- Node.js 10 and 12, Python 3.8, Ruby 2.7, Java 8 and 11, and .NET Core 3.1



Set via Lambda environment variable:

```
AWS_LAMBDA_EXEC_WRAPPER = /opt/wrapper_script
```

Code:

```
# the extra options we want to pass to the interpreter
extra_args = ["-X", "importtime"]
```

External extensions

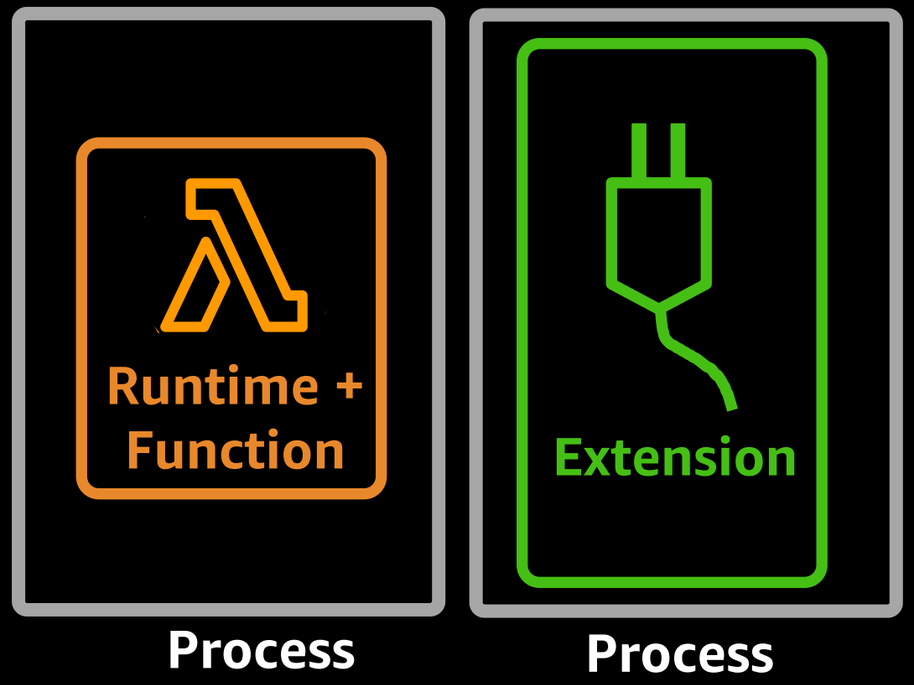
Extensions that run within the execution environment, but outside the runtime

Run as separate processes

Can start before the runtime process, and can continue after the runtime shuts down

Can be written in a different language to the function (recommend using a compiled language as a self-contained binary)

Node.js 10 and 12, Python 3.7 and 3.8, Ruby 2.5 and 2.7, Java Corretto 8 and 11, .NET Core 3.1, and custom runtimes



Resources and security

- Share resources such as CPU, memory, disk storage, and environment variables with the function and all extensions
- Share permissions, using the same AWS Identity and Access Management (IAM) role as the function
- Up to 10 extensions per function
  - Up to five layers at a time; multiple extensions can be included in a single layer
  - Include extensions in container image
- Size of the extensions counts towards the deployment package limit
  - Unzipped < 250 MB (ZIP archive functions) | < 10 GB (container image functions)

Performance

Extension performing compute-intensive operations:

- Can increase function execution duration

Initialized before the runtime is started:

- Can increase the delay before the function is invoked

Today, function invocation response returned after all extensions have completed:

- Can increase the delay before the function response is returned

Add more memory to a function also proportionally increases CPU and network throughput

Function and all extensions must complete within the function's configured timeout setting, which applies to the entire invoke phase

Performance

Measure additional extension time after function invocation

- Use the new `PostRuntimeExtensionsDuration`

Understand impact of a specific extension

- Use `Duration` and `MaxMemoryUsed` Amazon CloudWatch metrics
- Run different versions of your function with and without the extension

Pricing

Extensions share the same billing model as Lambda functions

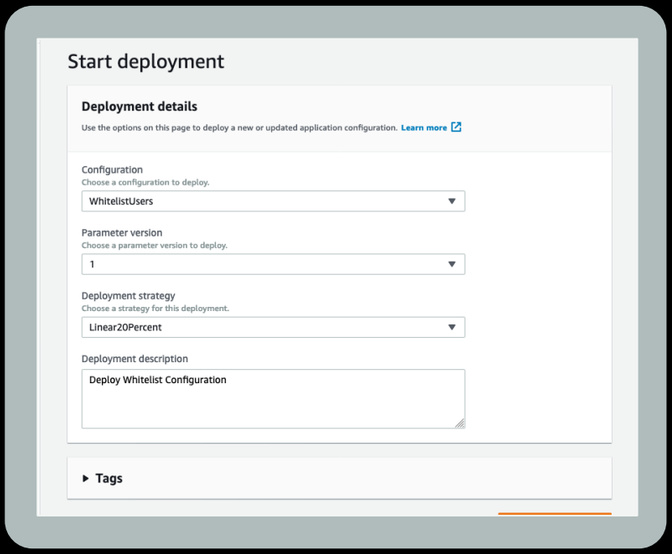
You pay for requests served and the combined compute time used to run your code and all extensions, in 1 ms increments

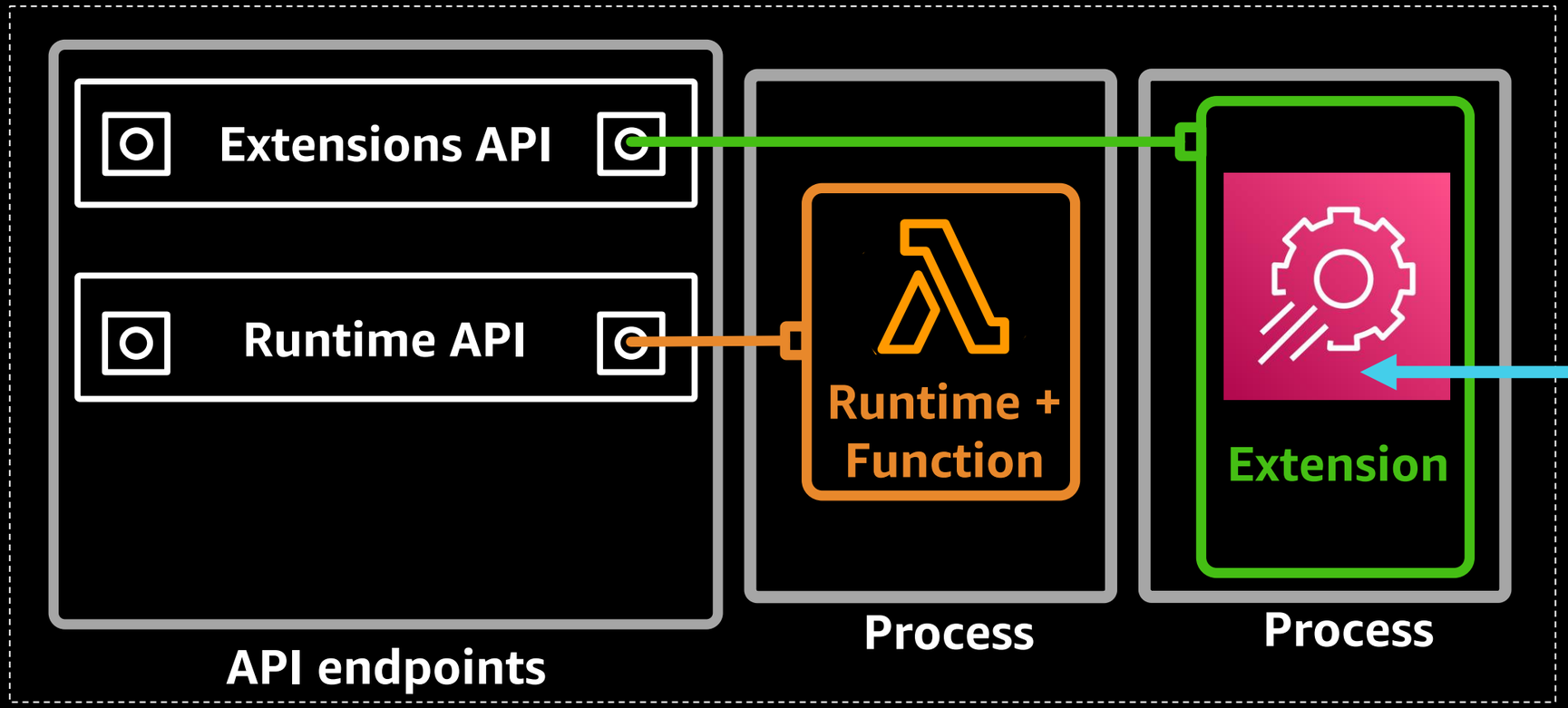
AWS AppConfig extension

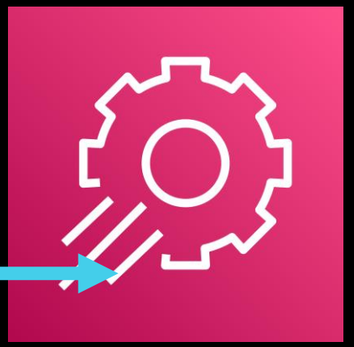
CREATE, MANAGE, AND DEPLOY APPLICATION CONFIGURATIONS

Deploy application configuration changes at runtime

Controlled deployment, validation, rollback







AWS AppConfig

CloudFormation - Stack s3-log | awsappconfig-extension-dem | Systems Manager Managemen | aws-lambda-extensions/awsap

https://github.com/aws-samples/aws-lambda-extensions/tree/main/awsappconfig-extension-demo 140%

Why GitHub? Team Enterprise Explore Marketplace Pricing Search / Sign in Sign up

aws-samples / aws-lambda-extensions

Watch 6 Star 80 Fork 13

Code Issues 2 Pull requests 1 Actions Security Insights

main aws-lambda-extensions / awsappconfig-extension-demo /

julianwood Updates README

..

src	Add awsappconfig-extension-demo
README.md	Updates README
template.yml	Added outputs to SAM template for AWS Config Extension Demo

https://s12d.com/extensions-gh

README.md

AWS AppConfig Extension Demo

This is a demo of AWS Lambda extensions using AWS AppConfig as explained in the blog post Introducing Lambda Extensions.

AWS AppConfig has an available extension to further integrate Lambda and AWS AppConfig. The extension runs a separate local process to

The Lambda lifecycle



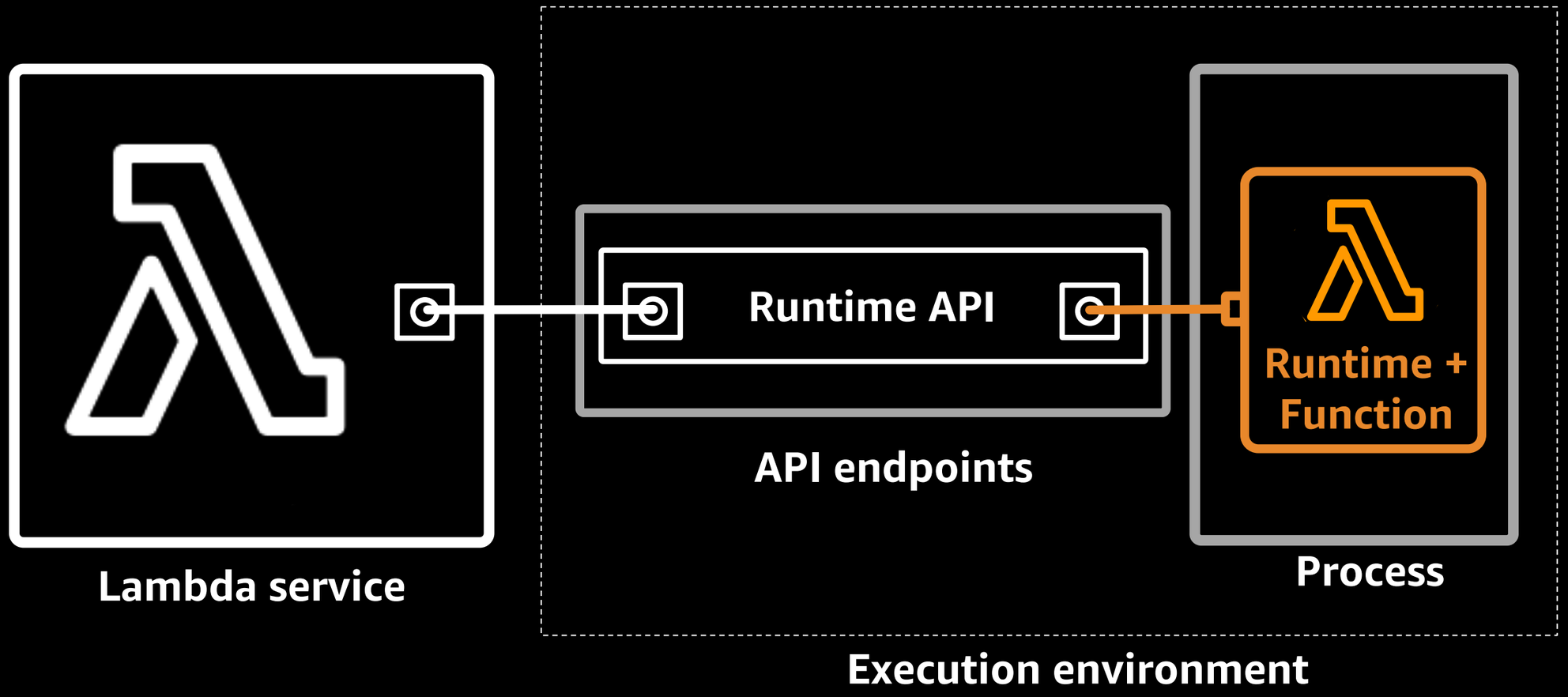
AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Lambda execution environment

BEFORE EXTENSIONS

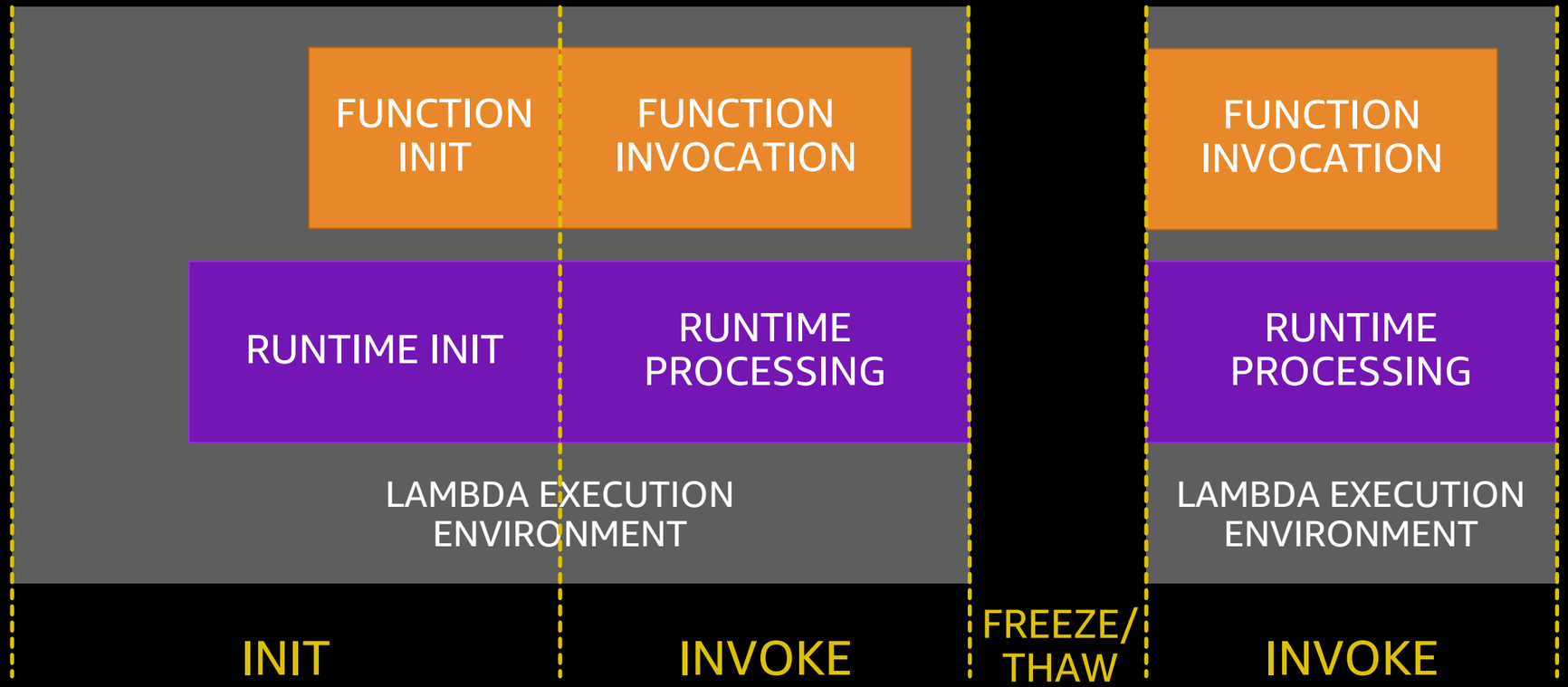
Lambda Runtime API, which enables you to bring custom runtimes to Lambda



Lambda execution environment

BEFORE EXTENSIONS

Lambda Runtime API, which enables you to bring custom runtimes to Lambda



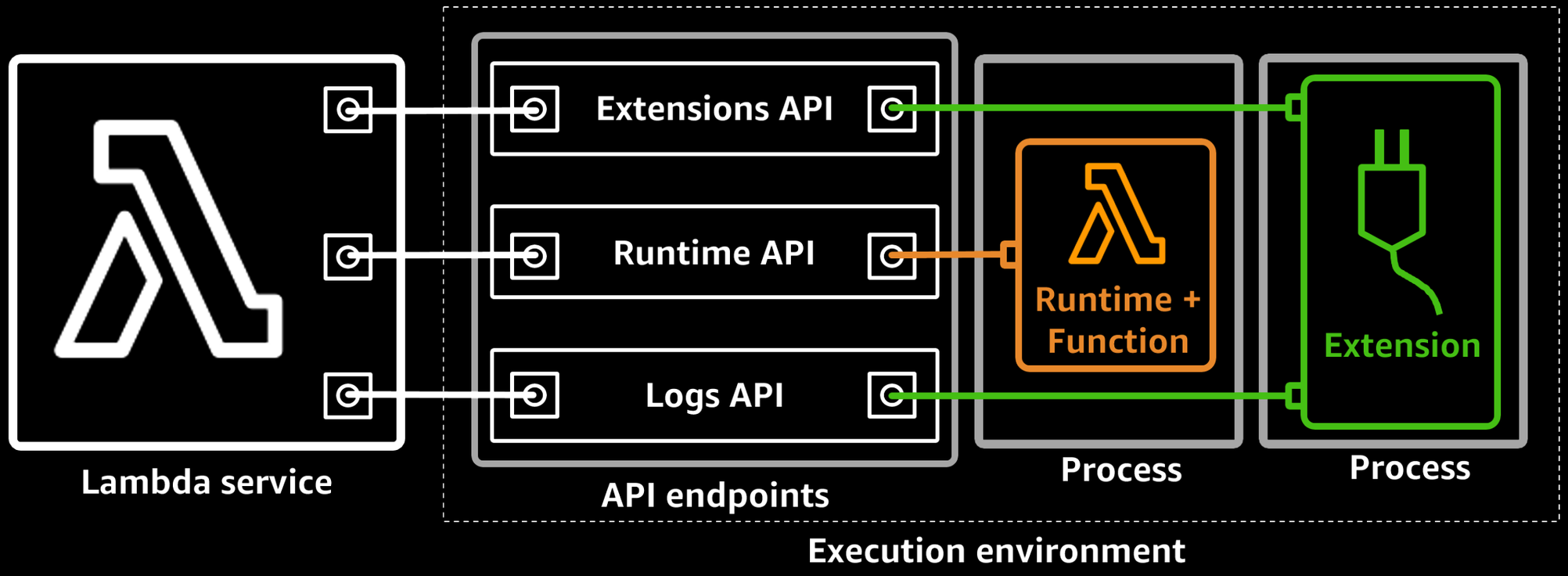
Lambda execution environment

WITH EXTENSIONS

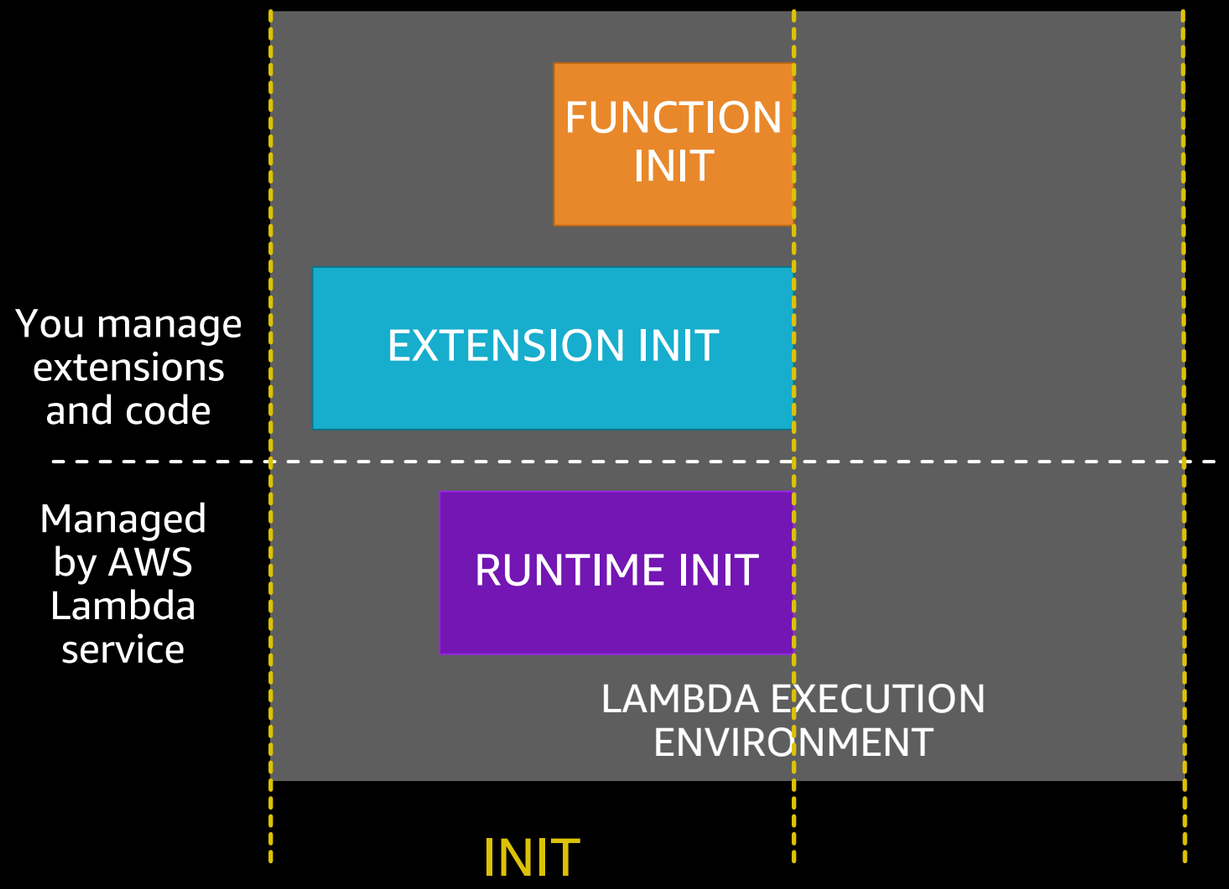
Extensions API builds on the existing Lambda Runtime API

Register for function and execution environment lifecycle events

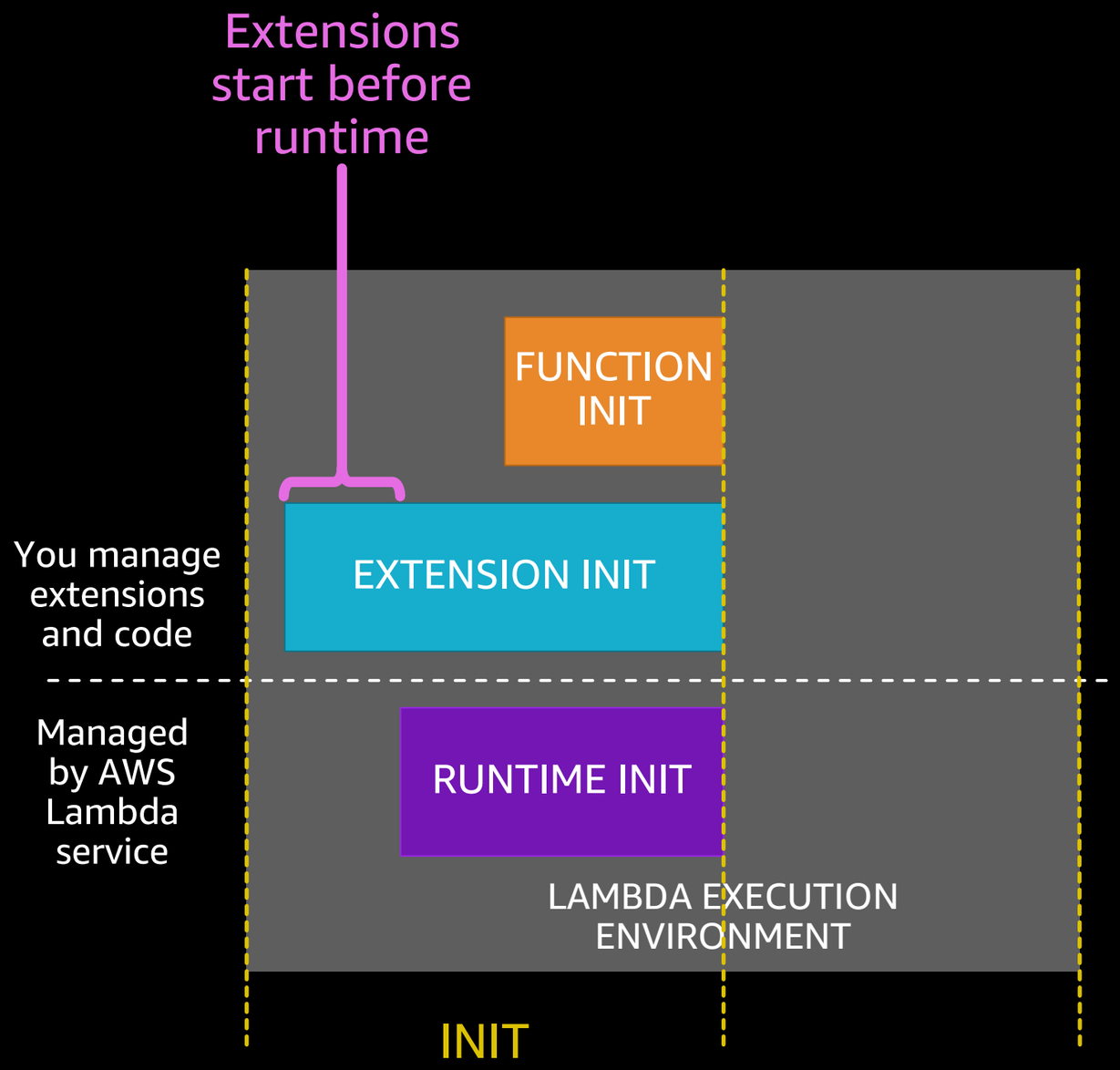
Extensions can use Logs API to register to receive logs directly from Lambda



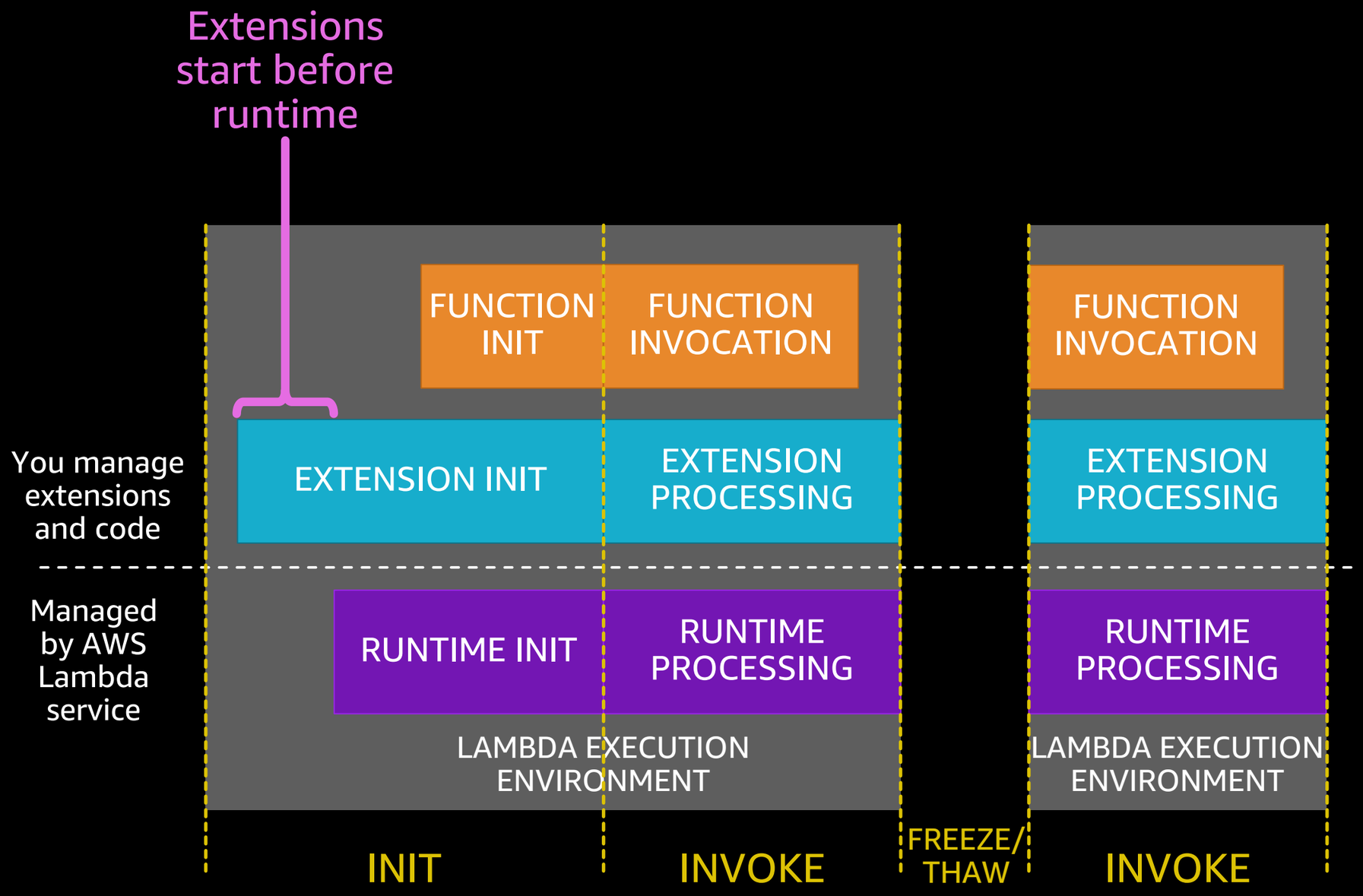
Lambda lifecycle with extensions



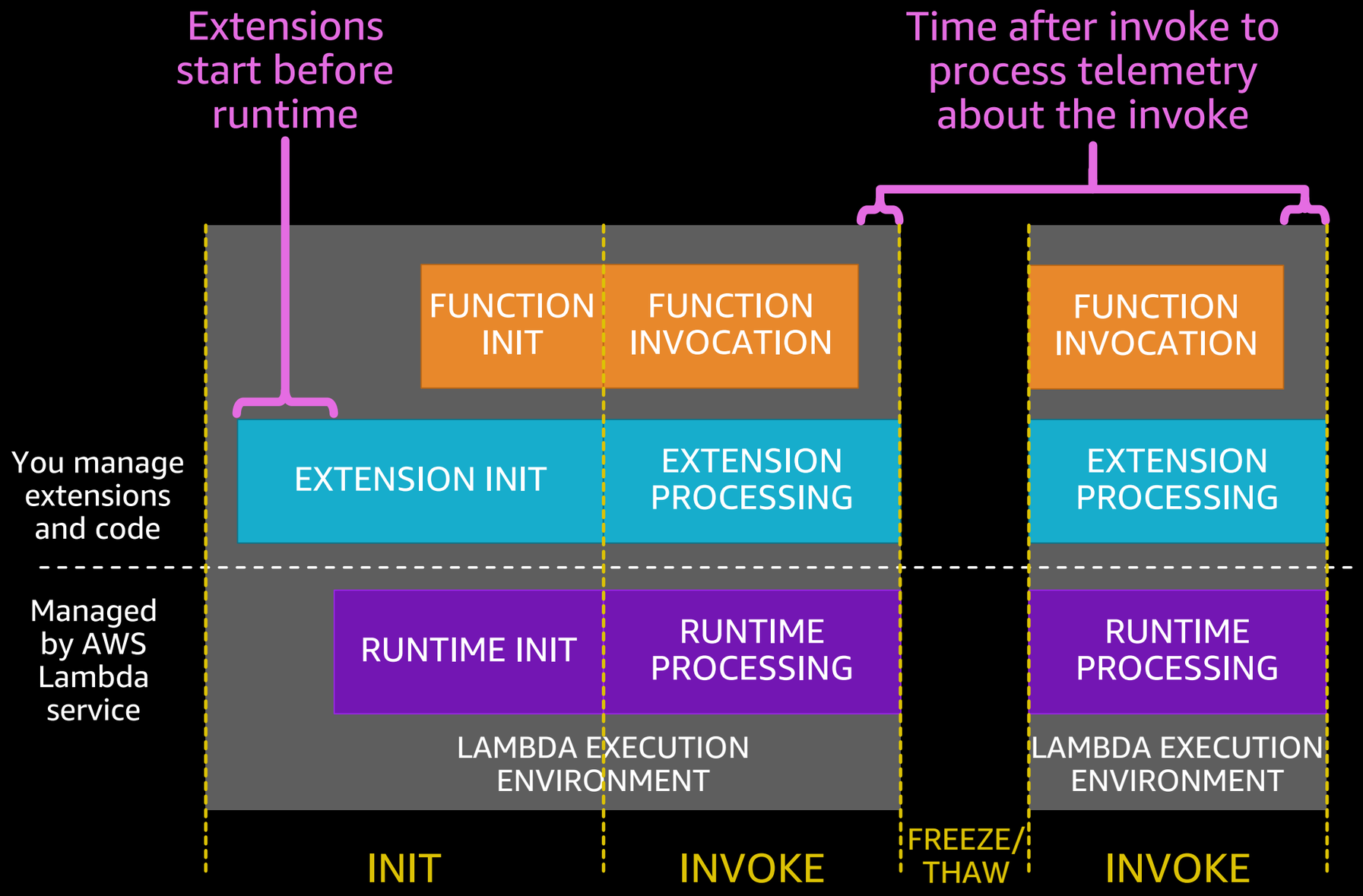
Lambda lifecycle with extensions



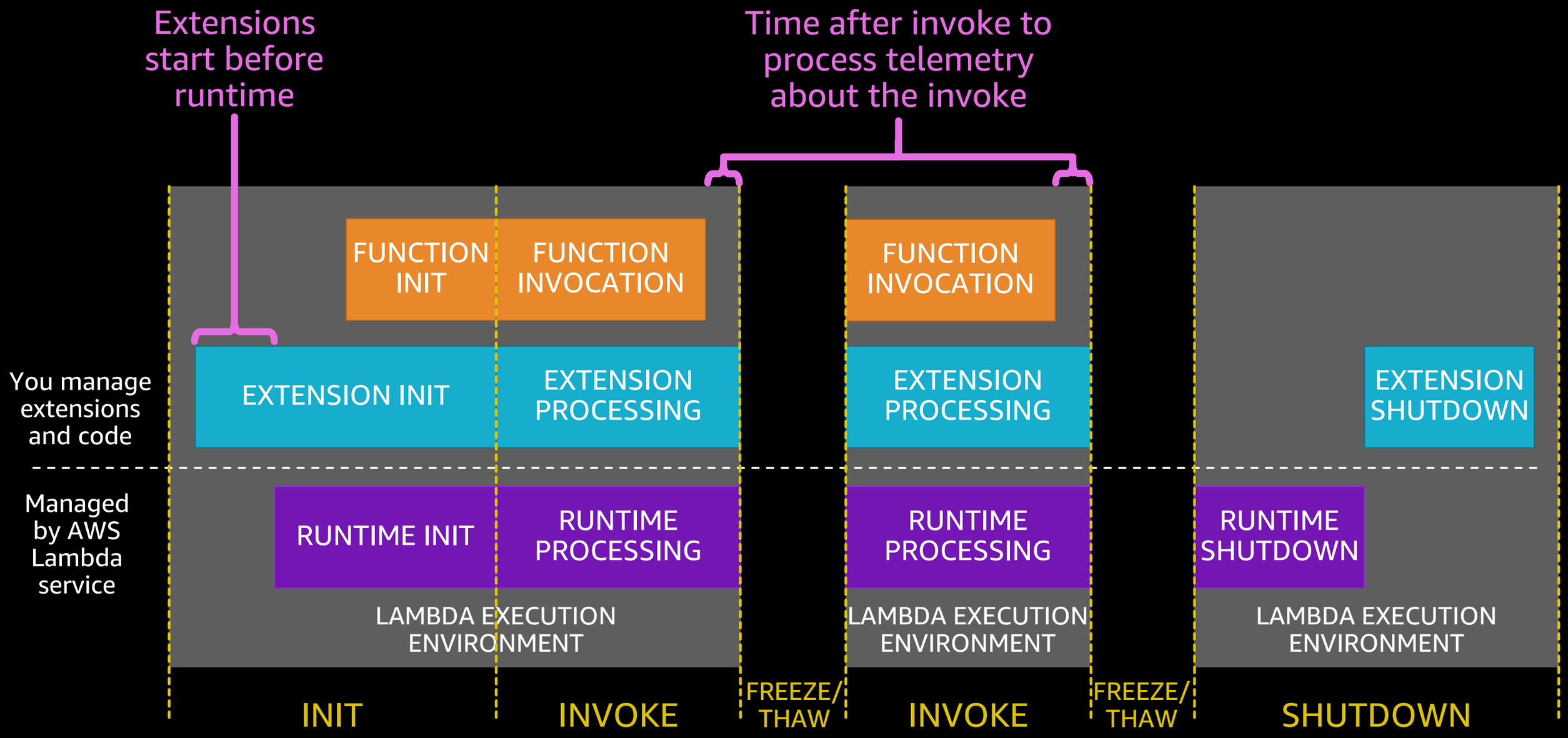
Lambda lifecycle with extensions



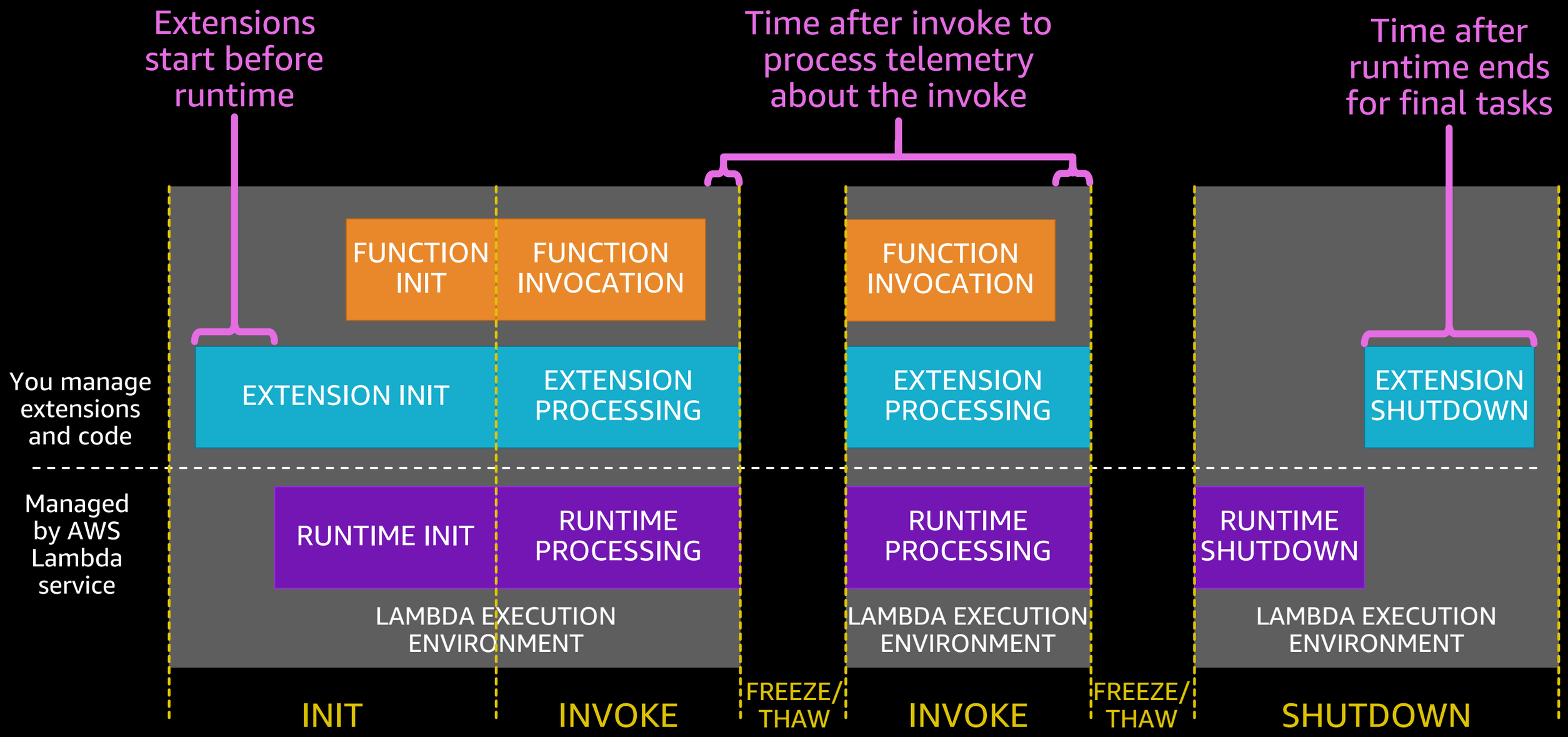
Lambda lifecycle with extensions



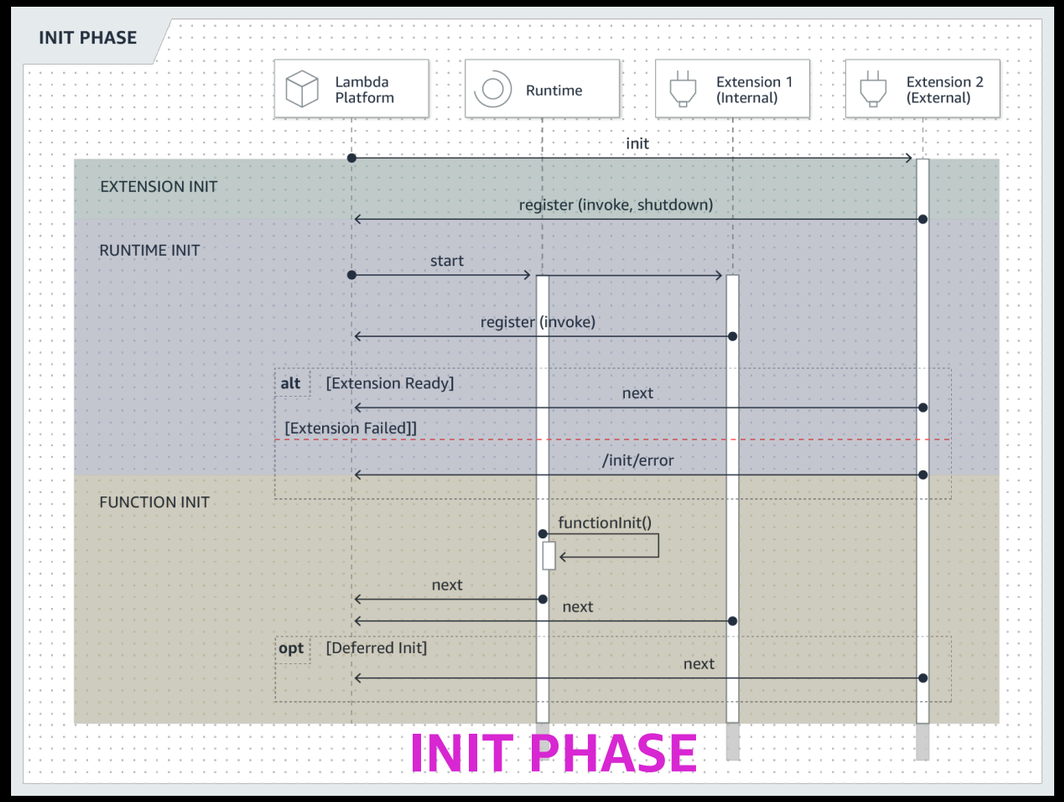
Lambda lifecycle with extensions

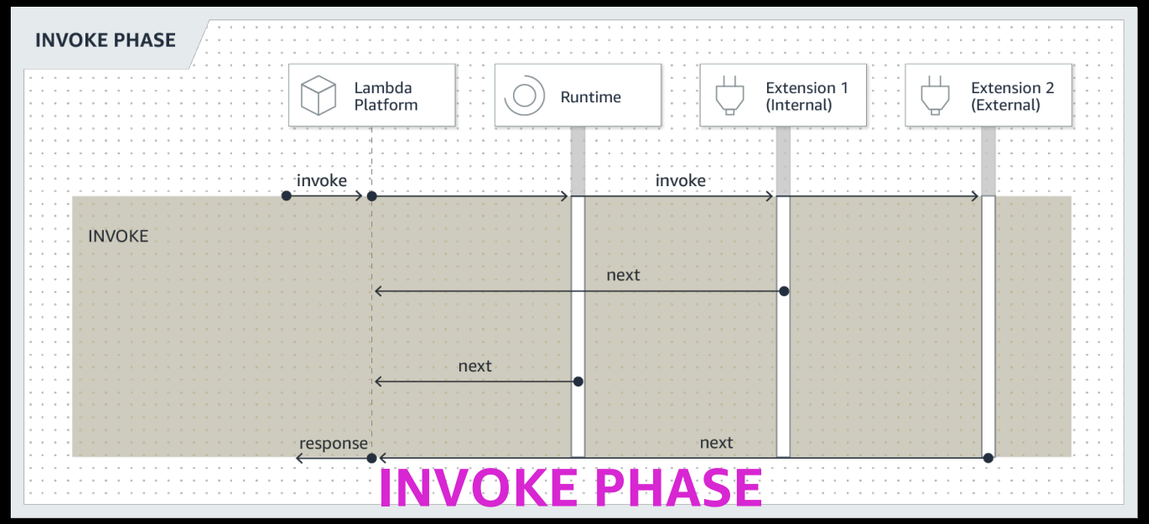


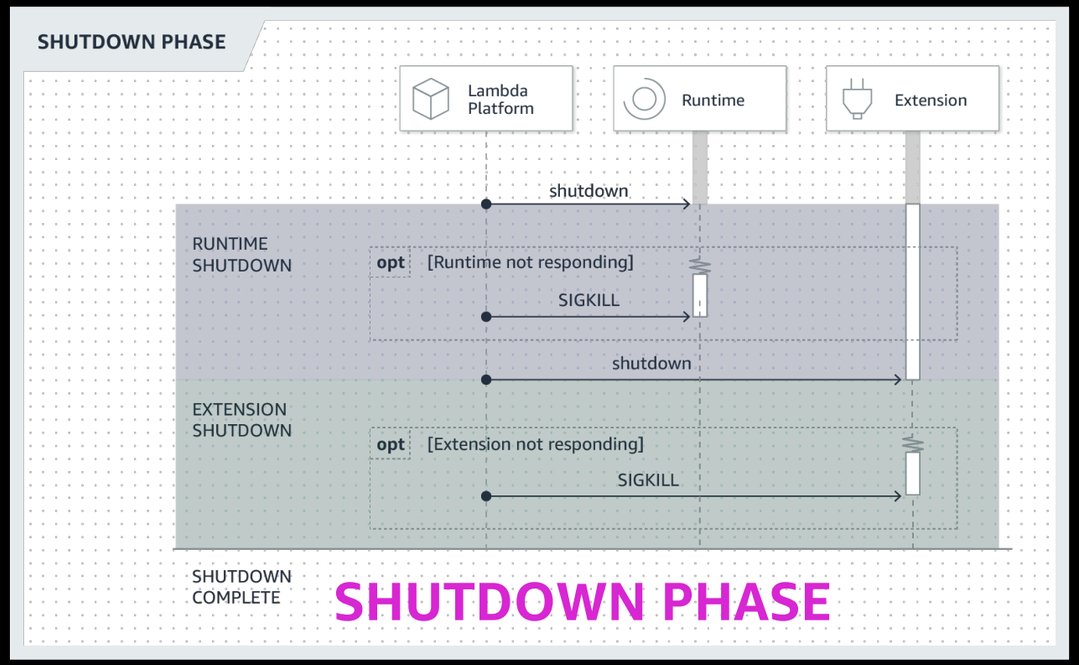
Lambda lifecycle with extensions



Lambda execution environment lifecycle







https://s12d.com/extensions-api-docs

Building Lambda extensions example

	Timestamp	Message
	2020-09-18T12:34:22.732+01:00	EXTENSION Name: extension1.sh State: Ready Events: [INVOKE,SHUTDOWN]
1	2020-09-18T12:34:23.768+01:00	START RequestId: 50d2e816-1479-4915-b409-4b9679ea385b Version: $LATEST
	2020-09-18T12:34:23.769+01:00	[extension1.sh] Registering...
	2020-09-18T12:34:23.769+01:00	[extension1.sh] Register response: {"functionName":"byol-runtime-demo-function","functionVersion":"
2	2020-09-18T12:34:24.772+01:00	[runtime] Initializing...
3	2020-09-18T12:34:24.772+01:00	[runtime] Waiting for invocation...
	2020-09-18T12:34:24.772+01:00	[extension1.sh] Waiting for event...
4	2020-09-18T12:34:24.809+01:00	[extension1.sh] Received event: {"eventType":"INVOKE","deadlineMs":958471,"requestId":"50d2e816-147
	2020-09-18T12:34:24.809+01:00	[runtime] Received invocation: {"payload": "hello"}
5	2020-09-18T12:34:24.869+01:00	[runtime] Executing function: function.handler
	2020-09-18T12:34:24.909+01:00	[function] Receiving invocation: '{"payload": "hello"}'
6	2020-09-18T12:34:24.909+01:00	[runtime] Sending invocation response: Echoing request: '{"payload": "hello"}'
	2020-09-18T12:34:25.029+01:00	[runtime] Runtime API response: {"status":"OK"}
7	2020-09-18T12:34:25.029+01:00	[runtime] Waiting for invocation...
	2020-09-18T12:34:25.829+01:00	[extension1.sh] PROCESSING/SLEEPING
8	2020-09-18T12:34:40.831+01:00	[extension1.sh] DONE PROCESSING/SLEEPING
	2020-09-18T12:34:41.833+01:00	[extension1.sh] Waiting for event...
9	2020-09-18T12:34:41.893+01:00	END RequestId: 50d2e816-1479-4915-b409-4b9679ea385b
	2020-09-18T12:34:41.893+01:00	REPORT RequestId: 50d2e816-1479-4915-b409-4b9679ea385b Duration: 17111.39 ms Billed Duration: 19200
10	2020-09-18T12:45:22.808+01:00...	[runtime] Received SIGTERM
	2020-09-18T12:45:22.808+01:00...	[runtime] Exiting
11	2020-09-18T12:45:22.819+01:00...	[extension: extension1.sh] Received SHUTDOWN event. Exiting.

https://s12d.com/extensions-building

Using extensions in Lambda container images

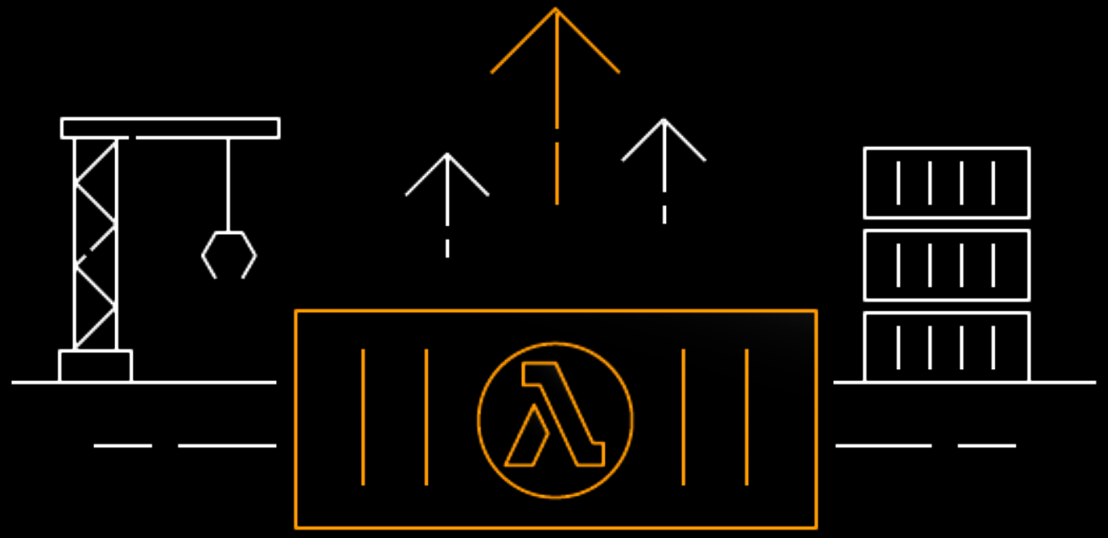


AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS Lambda container image support

PACKAGE AND DEPLOY FUNCTIONS AS CONTAINER IMAGES



- Use a consistent set of tools for containers and Lambda-based application
- Deploy large applications with AWS provided or third-party images of up to 10 GB
  - ML, data analytics, large dependencies
- Benefit from sub-second automatic scaling, high availability, 140 native service integrations, pay for use

https://s12d.com/lambda-containers

Using Lambda layers in container images

Lambda layers are used in .zip archive functions

Lambda layers are not directly supported in container images

Layers and extensions are files copied into specific file paths under `/opt` in the execution environment

The files are read-only in the execution environment

You can use a Dockerfile to copy layer functionality to container images

Using Lambda layers in container images

1. Use a container image version of a Lambda layer
2. Copy the contents of a Lambda layer into a container image
3. Build extension image layer Dockerfile
4. Build function image layer Dockerfile including extension image

```
FROM 123456789012.dkr.ecr.us-east-1.amazonaws.com/log-extension-image:latest AS layer
FROM public.ecr.aws/lambda/python:3.8
# Layer code
WORKDIR /opt
COPY --from=layer /opt/ .
```

https://s12d.com/image-layer-ex

Re:Cap

Use Lambda extensions to augment your Lambda functions

Easily plug in your existing tools to integrate deeply with Lambda
Use cases: monitoring, observability, security, governance, and more

Much more here:
https://s12d.com/SVS317

No complex installation, configuration, or operational overhead
Use extensions by configuring the options in your Lambda functions

Internal vs. external extensions
Use Extensions API to build your own extension
Use Logs API to consume platform and function logs directly from Lambda

How to use layers and extensions in Lambda container images

Serverlessland.com

Home Blogs Videos Learn Events About

Welcome to **Serverless** Land

This site brings together all the latest blogs, videos, and training for AWS Serverless. Learn to use and build apps that scale automatically on low-cost, fully-managed serverless architecture.



Thank you!

Julian Wood

@julian_wood

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Please complete the session survey

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.