What is AWS Lambda?

PDF (lambda-dg.pdf#welcome) | RSS (lambda-updates.rss)

Lambda is a compute service that lets you run code without provisioning or managing servers. Lambda runs your code on a high-availability compute infrastructure and performs all of the administration of the compute resources, including server and operating system maintenance, capacity provisioning and automatic scaling, and logging. With Lambda, you can run code for virtually any type of application or backend service. All you need to do is supply your code in one of the languages that Lambda supports (./lambda-runtimes.html) .

> **Note**
>
> In the AWS Lambda Developer Guide, we assume that you have experience with coding, compiling, and deploying programs using one of the supported languages.

You organize your code into Lambda functions (./gettingstarted-concepts.html#gettingstarted-concepts-function) . Lambda runs your function only when needed and scales automatically, from a few requests per day to thousands per second. You pay only for the compute time that you consume—there is no charge when your code is not running.

You can invoke your Lambda functions using the Lambda API, or Lambda can run your functions in response to events from other AWS services. For example, you can use Lambda to:

- Build data-processing triggers for AWS services such as Amazon Simple Storage Service (Amazon S3) and Amazon DynamoDB.
- Process streaming data stored in Amazon Kinesis.
- Create your own backend that operates at AWS scale, performance, and security.

Lambda is a highly available service. For more information, see the AWS Lambda Service Level Agreement (http://aws.amazon.com/lambda/sla/) .

Sections

- When should I use Lambda? (#when-to-use-cloud-functions)
- Lambda features (#features)
- Getting started with Lambda (#welcome-first-time-user)
- Related services (#related-services)
- Accessing Lambda (#accessing)

- Pricing for Lambda (#pricing)

When should I use Lambda?

Lambda is an ideal compute service for many application scenarios, as long as you can run your application code using the Lambda standard runtime environment (./lambda-runtime-environment.html) and within the resources that Lambda provides.

When using Lambda, you are responsible only for your code. Lambda manages the compute fleet that offers a balance of memory, CPU, network, and other resources to run your code. Because Lambda manages these resources, you cannot log in to compute instances or customize the operating system on provided runtimes (./lambda-runtimes.html) . Lambda performs operational and administrative activities on your behalf, including managing capacity, monitoring, and logging your Lambda functions.

If you need to manage your own compute resources, AWS has other compute services to meet your needs. For example:

- Amazon Elastic Compute Cloud (Amazon EC2) offers a wide range of EC2 instance types to choose from. It lets you customize operating systems, network and security settings, and the entire software stack. You are responsible for provisioning capacity, monitoring fleet health and performance, and using Availability Zones for fault tolerance.
- AWS Elastic Beanstalk enables you to deploy and scale applications onto Amazon EC2. You retain ownership and full control over the underlying EC2 instances.

Lambda features

The following key features help you develop Lambda applications that are scalable, secure, and easily extensible:

Concurrency and scaling controls

Concurrency and scaling controls (./invocation-scaling.html) such as concurrency limits and provisioned concurrency give you fine-grained control over the scaling and responsiveness of your production applications.

Functions defined as container images

Use your preferred container image (./images-create.html) tooling, workflows, and dependencies to build, test, and deploy your Lambda functions.

Code signing

Code signing (./configuration-codesigning.html) for Lambda provides trust and integrity controls that let you verify that only unaltered code that approved developers have published is deployed in your Lambda functions.

Lambda extensions

You can use Lambda extensions (./runtimes-extensions-api.html) to augment your Lambda functions. For example, use extensions to more easily integrate Lambda with your favorite tools for monitoring, observability, security, and governance.

Function blueprints

A function blueprint provides sample code that shows how to use Lambda with other AWS services or third-party applications. Blueprints include sample code and function configuration presets for Node.js and Python runtimes.

Database access

A database proxy (./configuration-database.html) manages a pool of database connections and relays queries from a function. This enables a function to reach high concurrency levels without exhausting database connections.

File systems access

You can configure a function to mount an Amazon Elastic File System (Amazon EFS) file system (./configuration-filesystem.html) to a local directory. With Amazon EFS, your function code can access and modify shared resources safely and at high concurrency.

Getting started with Lambda

To work effectively with Lambda, you need coding experience and expertise in the following domains:

- Linux OS and commands, as well as concepts such as processes, threads, and file permissions.
- Cloud concepts and IP networking concepts (for public and private networks).
- Distributed computing concepts such as HTTP as an IPC, queues, messaging, notifications, and concurrency.
- Familiarity with security services and concepts: AWS Identity and Access Management (IAM) and access control principles, and AWS Key Management Service (AWS KMS) and public key infrastructure.
- Familiarity with key services that interact with Lambda: Amazon API Gateway, Amazon S3, Amazon Simple Queue Service (Amazon SQS), and DynamoDB.
- Configuring EC2 instances with Linux.

If you are a first-time user of Lambda, we recommend that you start with the following topics to help you learn the basics:

1. **Read the Lambda product overview (http://aws.amazon.com/lambda/) and explore the Lambda getting started (http://aws.amazon.com/lambda/getting-started/) page.**
2. **To create and test a Lambda function using the Lambda console, try the console-based getting started exercise (./getting-started.html) .** This exercise teaches you about the Lambda programming model and other concepts.
3. **If you are familiar with container image workflows, try the getting started exercise to create a Lambda function defined as a container image (./gettingstarted-images.html) .**

AWS also provides the following resources for learning about serverless applications and Lambda:

- The AWS Compute Blog (http://aws.amazon.com/blogs/compute/) includes useful articles about Lambda.
- AWS Serverless (https://serverlessland.com/) provides blogs, videos, and training related to AWS serverless development.
- The AWS Online Tech Talks (https://www.youtube.com/channel/UCT-nPlVzJI-ccQXlxjSvJmw) YouTube channel includes videos about Lambda-related topics. For an overview of serverless applications and Lambda, see the Introduction to AWS Lambda & Serverless Applications (https://www.youtube.com/watch?v=EBSdyoO3goc) video.

Related services

Lambda integrates with other AWS services (./lambda-services.html) to invoke functions based on events that you specify. For example:

- Use API Gateway (./services-apigateway.html) to provide a secure and scalable gateway for web APIs that route HTTP requests to Lambda functions.
- For services that generate a queue or data stream (such as DynamoDB (./with-ddb.html) and Kinesis (./with-kinesis.html)), Lambda polls the queue or data stream from the service and invokes your function to process the received data.
- Define Amazon S3 (./with-s3.html) events that invoke a Lambda function to process Amazon S3 objects, for example, when an object is created or deleted.
- Use a Lambda function to process Amazon SQS (./with-sqs.html) messages or Amazon Simple Notification Service (Amazon SNS) (./with-sns.html) notifications.
- Use AWS Step Functions (./lambda-stepfunctions.html) to connect Lambda functions together into serverless workflows called state machines.

Accessing Lambda

You can create, invoke, and manage your Lambda functions using any of the following interfaces:

- **AWS Management Console** – Provides a web interface for you to access your functions. For more information, see Lambda console (./foundation-console.html) .
- **AWS Command Line Interface (AWS CLI)** – Provides commands for a broad set of AWS services, including Lambda, and is supported on Windows, macOS, and Linux. For more information, see Using Lambda with the AWS CLI (./gettingstarted-awscli.html) .
- **AWS SDKs** – Provide language-specific APIs and manage many of the connection details, such as signature calculation, request retry handling, and error handling. For more information, see AWS SDKs (http://aws.amazon.com/tools/#SDKs) .
- **AWS CloudFormation** – Enables you to create templates that define your Lambda applications. For more information, see AWS Lambda applications (./deploying-lambda-apps.html) . AWS CloudFormation also supports the AWS Cloud Development Kit (AWS CDK) (http://aws.amazon.com/cdk) .
- **AWS Serverless Application Model (AWS SAM)** – Provides templates and a CLI to configure and manage AWS serverless applications. For more information, see SAM CLI (./lambda-settingup.html#lambda-settingup-samcli) .

Pricing for Lambda

There is no additional charge for creating Lambda functions. There are charges for running a function and for data transfer between Lambda and other AWS services. Some optional Lambda features (such as provisioned concurrency (./configuration-concurrency.html)) also incur charges. For more information, see AWS Lambda Pricing (http://aws.amazon.com/lambda/pricing/) .