AWS

Cloud computing is the on-demand delivery of compute power, database, storage, applications, and other IT resources through a cloud services platform through the internet with pay-as-you-go pricing.

Traditionally, a developer looking to build an application had to procure, set up, and maintain physical infrastructure and the application.

A cloud services platform provides rapid access to flexible and low-cost IT resources that you can use to build and maintain software and databases, and create applications to delight customers.

You don’t need to make large upfront investments in hardware and spend a lot of time on the heavy lifting of managing that hardware.

You can access as many resources as you need, almost instantly, and only pay for what you use.

On-demand, pay-as-you-go access to services is fundamental to the cloud computing model.

### Advantages of cloud computing

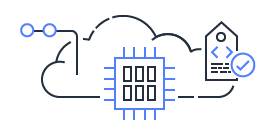
The cloud provides developers with greater flexibility, scalability, and faster time to innovation. With cloud computing, you can

* Pay as you go - Pay only when you use computing resources, and only for how much you use.
* Benefit from massive economies of scale - AWS aggregates usage from hundreds of thousands of customers in the cloud, which leads to higher economies of scale. This translates into lower pay-as-you-go prices.
* Stop guessing capacity - When you make a capacity decision prior to deploying an application, you often end up either sitting on expensive idle resources or dealing with limited capacity. With cloud computing, you can access as much or as little capacity as you need, and scale up and down as required with only a few minutes notice.
* Increase speed and agility - IT resources are only a click away, which means that you reduce the time to make resources available to your developers from weeks to minutes. This dramatically increases agility for the organization, because the cost and time it takes to experiment and develop is significantly lower.
* Realize cost savings - Companies can focus on projects that differentiate their business instead of maintaining data centers. With cloud computing, you can focus on your customers, rather than on the heavy lifting of racking, stacking, and powering physical infrastructure.
* Go global in minutes - Applications can be deployed in multiple Regions around the world with a few clicks. This means that you can provide lower latency and a better experience for your customers at a minimal cost.

Cloud computing provides developers with the ability to focus on what matters most and avoid infrastructure procurement, maintenance, and capacity planning, or undifferentiated heavy lifting.

With the growing popularity of cloud computing, several different service models have emerged to help meet specific needs of different users. Each type of cloud service provides you with different levels of abstraction, control, flexibility, and management.

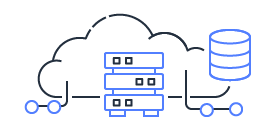
Understanding the differences between Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) can help you decide what service type is right for your needs as a developer.



### Infrastructure as a Service (IaaS)

Infrastructure as a Service (IaaS) contains the basic building blocks for cloud IT, and typically provides access to networking features, computers (virtual or on dedicated hardware), and data storage space. IaaS provides you with the highest level of flexibility and management control over your IT resources and is most like existing IT resources that many developers are familiar with today.

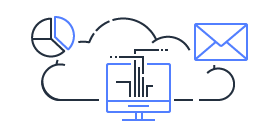
Try it out: [Deploy a LAMP Web App on Amazon Lightsail](https://aws.amazon.com/getting-started/guides/deploy-lamp-lightsail/?pg=cloudessentials)



### Platform as a Service (PaaS)

Platform as a Service (PaaS) removes the need for you to manage the underlying infrastructure (usually hardware and operating systems) and allows you to focus on the deployment and management of your applications. This helps you be more efficient because you don’t need to worry about resource procurement, capacity planning, software maintenance, patching, or any of the other undifferentiated heavy lifting involved in running your application.

Try it out: [Deploy a Web App on AWS Elastic Beanstalk](https://aws.amazon.com/getting-started/guides/deploy-webapp-elb/?pg=cloudessentials)



### Software as a Service (SaaS)

Software as a Service (SaaS) provides you with a completed product that is run and managed by the service provider. In most cases, people referring to SaaS are referring to end-user applications. With a SaaS offering you do not have to think about how the service is maintained or how the underlying infrastructure is managed; you only need to think about how you will use that piece of software. A common example of a SaaS application is web-based email which you can use to send and receive email without having to manage feature additions to the email product or maintain the servers and operating systems that the email program is running on.

Global Infrastructure

With the cloud, you can expand to new geographic regions and deploy globally in minutes. For example, AWS has infrastructure all over the world, so developers can deploy applications in multiple physical locations with just a few clicks. By putting your applications in closer proximity to your end users, you can reduce latency and improve the user experience.

AWS is steadily expanding global infrastructure to help our customers achieve lower latency and higher throughput, and to ensure that their data resides only in the AWS Region they specify. As our customers grow their businesses, AWS will continue to provide infrastructure that meets their global requirements.

AWS Cloud infrastructure is built around AWS Regions and Availability Zones. A Region is a physical location in the world where we have multiple Availability Zones. Availability Zones consist of one or more discrete data centers, each with redundant power, networking, and connectivity, housed in separate facilities. These Availability Zones offer you the ability to operate production applications and databases that are more highly available, fault tolerant, and scalable than would be possible from a single data center.

AWS Cloud infrastructure is extensive, offering 200 fully featured services from data centers globally. With the [largest global infrastructure footprint](https://aws.amazon.com/about-aws/global-infrastructure/?pg=cloudessentials) of any cloud provider, AWS provides you the cloud infrastructure where and when you need it.

### AWS Global Infrastructure Map

[](https://aws.amazon.com/what-is-aws)

### How to interact with AWS

When infrastructure becomes virtual, as with cloud computing, the way developers work with infrastructure changes slightly. Instead of physically managing infrastructure, you logically manage it, through the AWS Application Programming Interface (AWS API). When you create, delete, or change any AWS resource, you will use API calls to AWS to do that.

You can make these API calls in several ways, but we will focus on these to introduce this topic:

* The AWS Management Console
* The AWS Command Line Interface (AWS CLI)
* IDE and IDE toolkits
* AWS Software Development Kits (SDKs)

### The AWS Management Console

When first getting started with AWS, people often begin with the AWS Management Console, a web-based console that you log in to through a browser. The console comprises a broad collection of service consoles for managing AWS resources. By working in the console, you do not need to worry about scripting or syntax. You can also select the specific Region you want an AWS service to be in.

After working in the console, you may want to move away from manual deployment of AWS service, perhaps because you have become more familiar with AWS or are working in a production environment that requires a degree of risk management. This is where the AWS Command Line Interface (CLI) comes in.

### AWS CLI

The [AWS CLI](https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-welcome.html?pg=cloudessentials) is an open source tool that enables you to create and configure AWS services using commands in your command-line shell. You can run commands in Linux or macOS using common shell programs such as bash, zsh, and tcsh, or on Windows, at the Windows command prompt or in PowerShell. One option for getting up and running quickly with the AWS CLI is [AWS CloudShell](https://aws.amazon.com/cloudshell/?pg=cloudessentials), a browser-based shell that provides command-line access to AWS resources. CloudShell is pre-authenticated with your console credentials. Common development and operations tools are pre-installed, so no local installation or configuration is required.

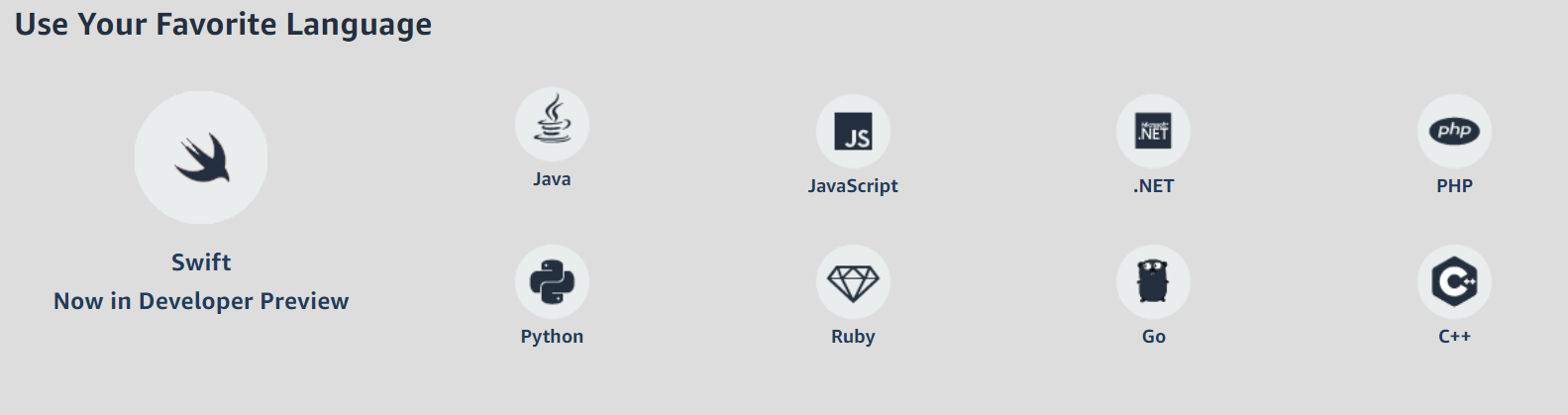
By moving to the AWS CLI, you can script or program the API calls. Instead of using a GUI, you create commands using a defined AWS syntax. One benefit of the CLI is that you can create single commands to create multiple AWS resources, which could help reduce the chance of human error when selecting and configuring resources. With the CLI, you need to learn the proper syntax for forming commands, but as you script these commands, you make them repeatable. This should save you time in the long run.

### IDE and IDE Toolkits

AWS offers support for popular Integrated Development Environments (IDEs) and IDE toolkits so you can author, debug, and deploy your code on AWS from within your preferred environment. Supported IDEs and toolkits include [AWS Cloud9](https://aws.amazon.com/cloud9/?pg=cloudessentials), IntelliJ, PyCharm, Visual Studio, Visual Studio Code, Azure DevOps, Rider, and WebStorm.

### SDKs

Software Development Kits (SDKs) are tools that allow you to interact with the AWS API programmatically. AWS creates and maintains SDKs for most popular programming languages, including those shown in the following diagram.

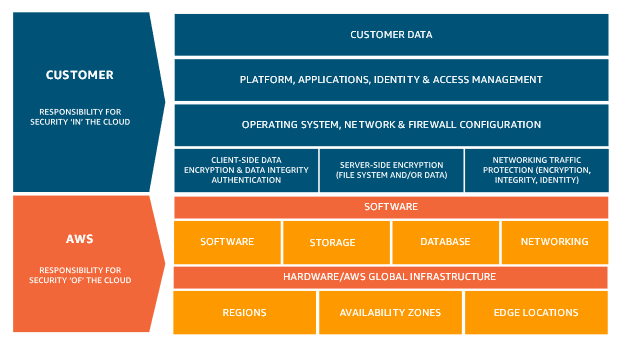


SDKs come in handy when you want to integrate your application source code with AWS services. For example, you might use the [Python SDK](https://boto3.amazonaws.com/v1/documentation/api/latest/guide/index.html?pg=cloudessentials) to write code to store files in Amazon Simple Storage Service (Amazon S3) instead of on your local hard drive. The ability to manage AWS services from a place where you can run source code, with conditions, loops, arrays, lists, and other programming elements, provides a lot of power and creativity.

These are just some of the tools available to developers on AWS. For a full list of AWS tools for developing applications faster and easier, see [Tools to Build on AWS](https://aws.amazon.com/tools/?pg=cloudessentials).

Security

When you build applications on AWS, managing security and compliance is a shared responsibility between AWS and you. To depict this shared responsibility, AWS created the shared responsibility model (see the following diagram). The distinction of responsibility is commonly referred to as security “of” the cloud compared to security “in” the cloud.



### AWS responsibility

Being responsible for security of the cloud means that AWS protects and secures the infrastructure that runs the services offered in the AWS Cloud. AWS is responsible for:

* Protecting and securing AWS Regions, Availability Zones, and data centers, down to the physical security of the buildings
* Managing the hardware, software, and networking components that run AWS services, such as the physical servers, host operating systems, virtualization layers, and AWS networking components.

### Customer responsibility

Customers, or anyone building on the cloud, are responsible for security in the cloud. When using any AWS service, you’re responsible for properly configuring the service and your applications, in addition to ensuring that your data is secure.

Your level of responsibility depends on the AWS service. Some services require you to perform all the necessary security configuration and management tasks, while other more abstracted services require you to only manage the data and control access to your resources.

Due to the varying levels of effort, customers must consider which AWS services they use and review the level of responsibility required to secure each service. They must also review how the shared security model aligns with the security standards in their IT environment, in addition to any applicable laws and regulations.

A key concept is that customers maintain complete control of their data and are responsible for managing the security related to their content.