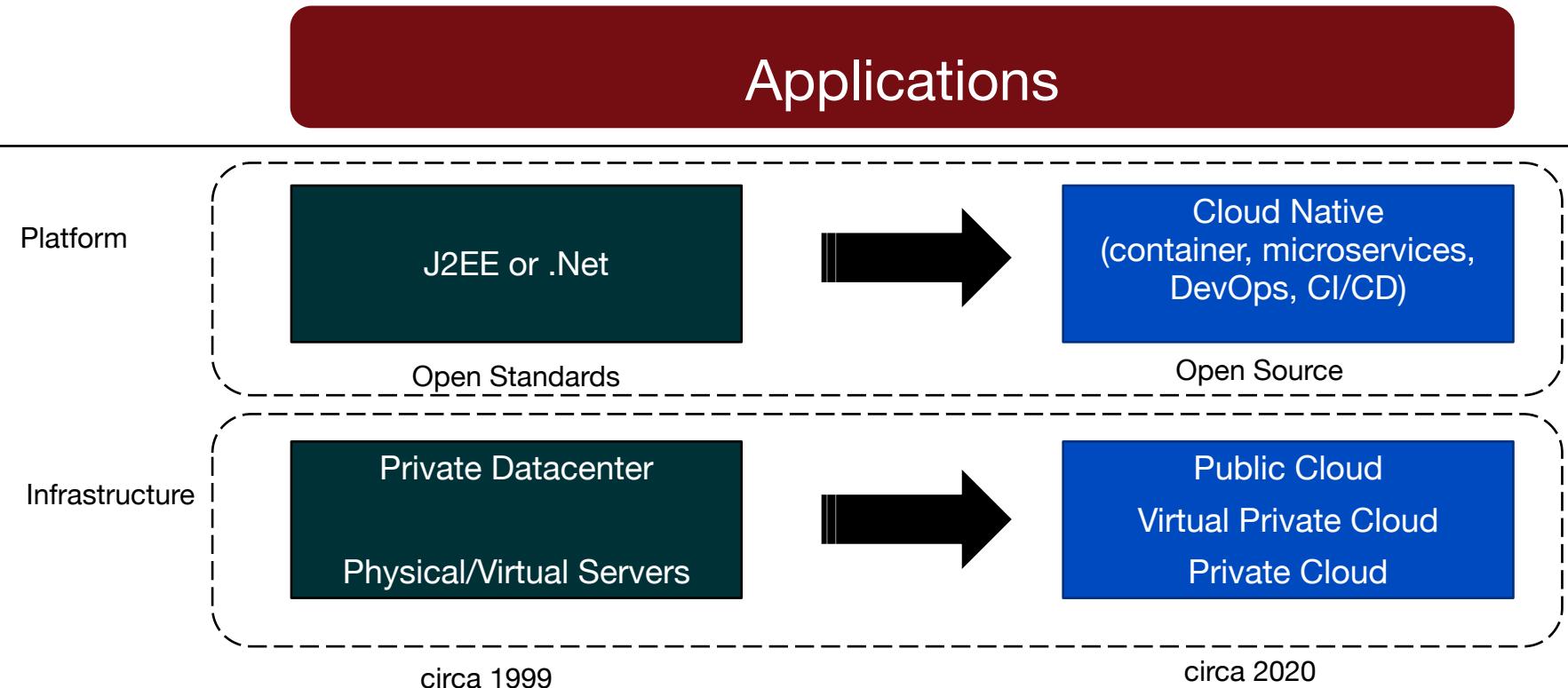


# Cloud Native - Runtimes and Frameworks

---

# Evolution of the App Platform



# Tradeoffs in Application Design

Application Requirements

Stateful & Stateless   Events & Requests

Simple & Flexible

Platform

# Runtime and frameworks



Runtime is a platform for executing commands

For e.g.

JRE, (a Java runtime environment)

Node.js (a JavaScript runtime environment)

Do not enforce any development patterns



A framework focuses more on best practices

Frameworks are typically opinionated,  
designed to produce predictable outcomes.

For e.g. Spring Framework (An opinionated way to create a web server that runs on the java runtime)

Frameworks are often used to fill knowledge gaps

Simply make developers more efficient by providing reliable/predictable outcomes.

# Three stacks for three styles for their runtimes..

12-Factor Apps



Cloud Foundry  
+ others

Functions



OpenWhisk  
+ many others

Containers



Kubernetes

Container Runtime

# Runtimes & Frameworks – Features, Benefits & why



## Simpler development

Integrate the capabilities of programming model of multiple runtimes.



## More runtimes for development

Offer the flexibility developers need to pick the right runtimes for their development choices.



## Strategic flexibility

Get the flexibility to pick the right blend of [containerized](#) runtimes and middleware services for your application.



## DevOps integration and automation

Provide developers with a self-service platform for provisioning, building, and deploying both apps and their components.

- Cloud-native development - simpler and more flexible
- Needs portability across multiple cloud infrastructures -> allowing the developer to use *microservices*, containers, and DevOps automation.
- Build new apps—faster than ever

# Infrastructure as Code (IaC)

*Infrastructure as code (IaC) - means to provision & manage your IT infrastructure using configuration / scripts files written in code.*

## **Scripting / Imperative**

*Creating infrastructure by writing scripts*

- End state to be reached line by line, i.e recipe

*ex: shell / bash*

*Example: (step by step to reach end state)*

```
ibmcloud cli create K8s  
ibmcloud cli create VM  
ibmcloud cli create VPC
```

## **Declarative**

*Creating infrastructure by writing declarative code, not controlled by the flow of code*

- End state is important & now how to reach end-state

*ex: CloudFormation, Terraform*

*Example: (end state)*

*K8s resource*

- > host: ---.---.---
- > port: --

*VM resources*

- > domain: --.--

*VPN resources*

- > subnets: --.---.---
- > IP: --.---.---

# Spring Boot – What ?

Firstly, Spring boot is not a framework & Spring framework has been around.

It's an approach to develop spring application with less configuration.

It is easy to learn when you know spring framework, it developed on top of existing spring framework.

It's an innovation of spring team.

It is open source.

# History of Spring Framework

- 1999 J2EE 1.2
- 2001 xDoclet 1.0
- **2004 Spring Framework 1.0**
- 2006 Java EE 5
- **2013 Spring Boot**

# Spring Framework

- Spring is one of the most widely used Java EE Frameworks for building applications.
- One of the major features of the Spring framework is the dependency injection. It helps make things simpler by allowing us to develop loosely coupled applications.
- The Spring framework can be used for all layers of implementation in the development of an application. It supports declarative programming.
- Considering its POJO model, it is a very lightweight framework.
- It allows loose coupling and easy testability.
- It is capable of eliminating the formation of singleton and factory classes.
- It supports both XML and annotation configurations.
- It provides middleware services.

# Why Spring Boot?

- Easy to developed spring-based application with java - [Spring Boot](#) doesn't require you to deploy WAR files.
- It aims to reduce the IOC (DI – Dependency Injection). It reduce lots of development time and increase productivity.
- Avoid writing boilerplate code, annotation and xml configuration
- Easy to integrate with spring ecosystem like **jdbc, security, orm, data** etc.
- It follows opinionated default configuration to spring development
- It provide CLI tool to developed and test spring boot. It simplifies Spring dependencies and runs applications straight from a command line.
- Use Maven and Gradle to build application easily
- It offers production ready features.

# MicroProfile

Optimizing Enterprise Java  
for a MicroService Architecture

- <http://microprofile.io/>
- Project @ Eclipse Foundation  
Open Source (Eclipse)
- Enterprise Java for Microservices
- Builds on Java EE standards



The image shows the official MicroProfile website. At the top is the MicroProfile logo, which consists of three yellow interconnected circles. To the right of the logo is the word "MICROPROFILE" in large blue letters, with "TM" in small letters at the top right corner. Below "MICROPROFILE" is the tagline "OPTIMIZING ENTERPRISE JAVA" in orange. The main content area features the text "Eclipse MicroProfile" and "Optimizing Enterprise Java for a Microservices Architecture". It also mentions "Now Available Eclipse MicroProfile 3.3" and a "More Information" button. On the right side of the main content area is a white rectangular badge for the "Duke's Choice Award 2018 Winner" from Oracle, featuring a cartoon character holding a coffee cup.



# Need for Eclipse MicroProfile

- With the Java EE full profile becoming so large, over 30 APIs at the last count, it doesn't seem right to include all those specifications when you are building something that you call 'micro',
- Not going to use the vast majority of those Java EE APIs
- The space was becoming fragmented, with each vendor implementing their own opinion about the right way to do microservices



### MicroProfile 1.2 (Sept 2017)

MicroProfile-1.1  
Config 1.1  
Fault Tolerance 1.0  
Health 1.0  
Metrics 1.0  
JWT 1.0

### MicroProfile 1.4 (June 2018)

MicroProfile 1.3  
Config 1.3  
Fault Tolerance 1.1  
JWT 1.1  
Open Tracing-1.1  
Rest Client-1.1

### MicroProfile 1.1 (August 2017)

microProfile-1.0  
Config 1.0

### MicroProfile 1.3 (Dec 2017)

MicroProfile 1.2  
Config 1.2  
Metrics 1.1  
OpenApi 1.0  
OpenTracing 1.0  
RestClient 1.0

### MicroProfile 3.0 (June 2019)

MicroProfile 2.1

Metrics 2.0  
Health Check  
2.0  
Fault Tolerance 2.0  
OpenAPI 1.1  
OpenTracing 1.3  
Rest Client 1.2

2019

### MicroProfile 2.1 (Oct 2018)

MicroProfile 2.0  
OpenTracing 1.2

### MicroProfile 2.0.1 (July 2018)

MicroProfile 1.4  
JAX-RS 2.1 // Java EE 8  
CDI 2.0 // Java EE 8  
JSON-P 1.1 // Java EE 8  
JSON-B 1.0 // Java EE 8

2020

### MicroProfile 3.2 (Nov 2019)

MicroProfile 3.0  
Metrics 2.2  
Health Check 2.1

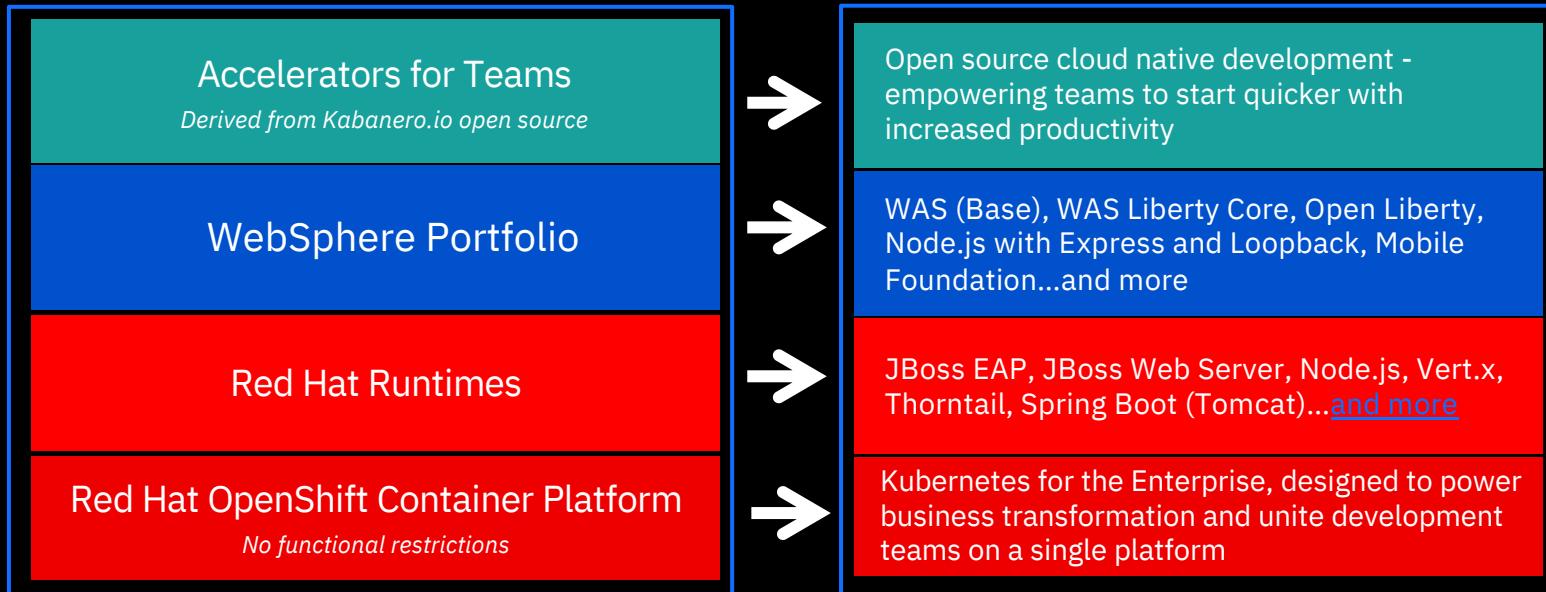
### MicroProfile 3.3 (Feb 2020)

MicroProfile 3.2  
Config 1.4  
Metrics 2.3  
Fault Tolerance 2.1  
Health 2.2  
Rest Client 1.4

# IBM Cloud Pak for Applications

*What is “OpenShift plus enterprise runtimes plus enterprise runtimes and accelerators for teams”?*

**OpenShift plus WebSphere portfolio and RH Runtimes and accelerators for teams**



*Best combination of platform, runtimes, and tools for Cloud Native development*

# *Cloud Pak for Applications: what you need today, what you need tomorrow*



## Build new cloud-native apps

### ACCELERATORS FOR TEAMS & ENTERPRISE GOVERNANCE

Equipping teams with content, tools, architectures and methods so they can focus on business problems from day 1

#### Red Hat CodeReady Workspaces

Collaborative OpenShift-native IDE

#### Red Hat Runtimes

**Java/Jakarta EE**  
Open Liberty, JBoss EAP

**Java SE**  
OpenJDK

**Java Web**  
JBoss WS

**Javascript**  
Node.js

**Spring**  
Spring Boot

**Serverless**  
Cloud Functions

**MicroProfile**  
Open Liberty, Thorntail

**Reactive**  
Vert.x

**Java.next**  
Quarkus

#### WebSphere

**Traditional WebSphere**

**Liberty**

**Distributed Data**  
Data Grid

**Messaging**  
AMQ Broker

**Mobile Foundation**

#### Red Hat OpenShift Container Platform

# Cloud Pak for Apps Components:

## Red Hat Runtimes

Red Hat Runtimes provides a set of open runtimes, tools, and components for developing and maintaining cloud-native applications. It offers lightweight runtimes and frameworks for highly distributed cloud architectures, such as microservices.

<https://access.redhat.com/articles/4394291>

Product or Service	Product Information & Documentation
Red Hat JBoss® Enterprise Application Platform	 Red Hat JBoss® Enterprise Application Platform
Red Hat JBoss® Web Server	<a href="#">Red Hat JBoss® Web Server</a>
Node.js	
Eclipse Vert.x	
Thorntail	 THORNTAIL
Spring Boot	<a href="#">Spring Boot</a>
Open Liberty	 Open Liberty
OpenJDK	<a href="#">OpenJDK</a>
Red Hat Data Grid	
Red Hat AMQ Broker	
Red Hat Single Sign-On	<a href="#">Red Hat Single Sign-On</a>
Red Hat Core Services	<a href="#">Red Hat Core Services</a>
Launcher Service	<a href="#">Launcher on developers.redhat.com</a>

# IBM WebSphere Liberty

## Open Liberty

Open Liberty is the OS foundation for the WebSphere Liberty product portfolio that can be used for developing new cloud-native applications with zero start-up cost.

## WebSphere Application Server

Use this version for Java EE Full Profile applications that need to integrate with existing Java EE applications such as those using JMS.

## WebSphere Liberty Core

The ideal version for Web applications or microservices that rely only on the Java EE Web Profile.

## WebSphere Application Server ND

Use this version for managing Java EE Full Profile applications that run outside of a containerized environment.

## WebSphere Application Server Family edition

The Family edition offers increased license flexibility across your enterprise for three WebSphere products: WebSphere Application Server, WebSphere Liberty Core and WebSphere Application Server ND.



# CLOUD FOUNDRY



# CLOUD FOUNDRY

## Discover

Use

### Ecosystem

## Community

## Events

About

# Open Source Cloud Application Platform

**Backed by Cisco, Google, IBM, Microsoft, Pivotal, SAP, SUSE and more**

Cloud Foundry makes it faster and easier to build, test, deploy and scale applications, providing a choice of clouds, developer frameworks, and application services. It is an open source project and is available through a variety of private cloud distributions and public cloud instances.

## WHY CLOUD FOUNDRY

GET STARTED

*”Here is my source code  
Run it on the cloud for me  
I do not care how”*

# Create Apps, Not the Platform

Remove the cost and complexity of configuring infrastructure for your apps.

## ANY APP

Cloud Foundry has a container-based architecture that runs apps in any programming language. Deploy apps to CF using your existing tools and with zero modification to the code. Instantiate, deploy, and manage high-availability Kubernetes clusters with CF BOSH on any cloud.

## ACCESS TO SERVICES

Applications deployed to Cloud Foundry access external resources via the Open Service Broker API. See available services and integrations in The Foundry.

INTEROPERABILITY

Cloud Foundry is highly adaptable and will withstand shifts in technology so you can adopt new tools, languages or platforms down the road.

ANY CLOUD

By decoupling applications from infrastructure, you can make individual decisions about where to host workloads – on-premise, in [public clouds](#), or in managed infrastructures – and move those workloads as necessary in minutes, with no changes to the app.

# INTEGRATION

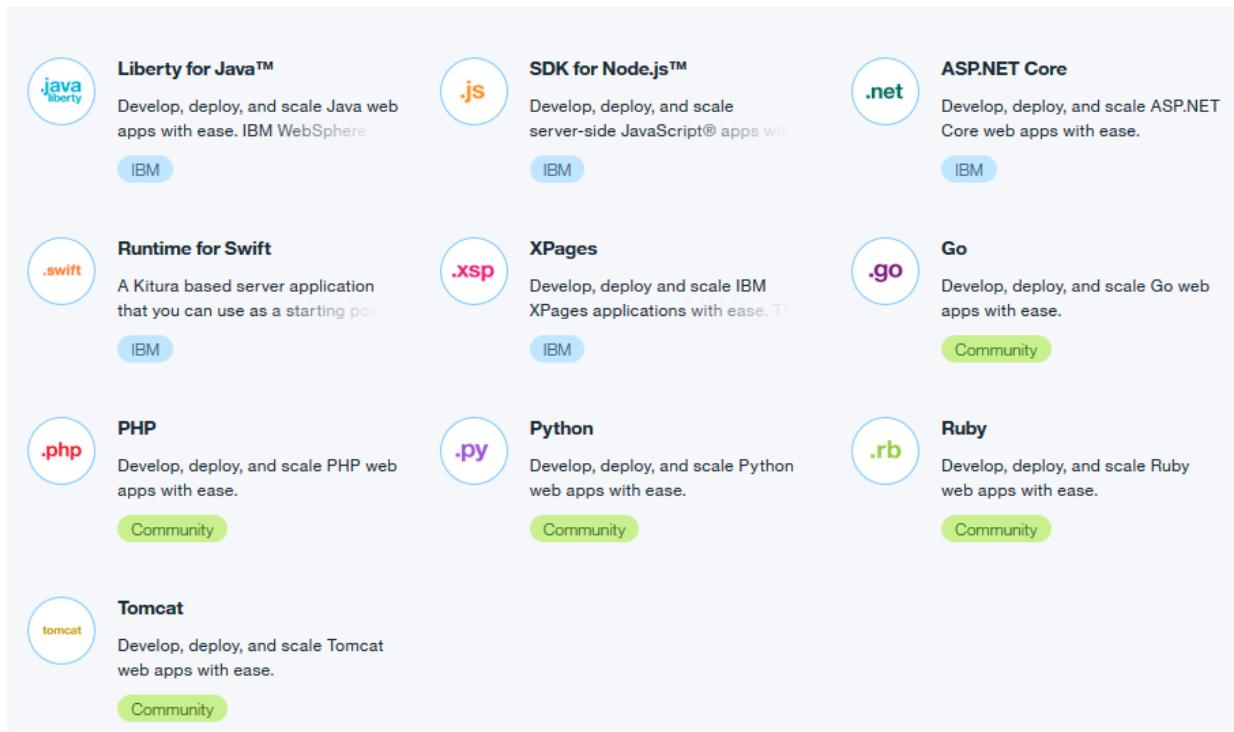
Cloud Foundry won't disrupt your current workflow. It is compatible with the tech and tools you use today – whether that's AWS or Docker or Kubernetes or Java or .NET – and just about anything in your current environment.

OPEN SOURCE

# Application runtimes and buildpacks

## Cloud Foundry Buildpacks

- specific runtime
- build framework
- types on IBM Cloud
  - IBM
  - Cloud Foundry Community
  - External (bring your own!)



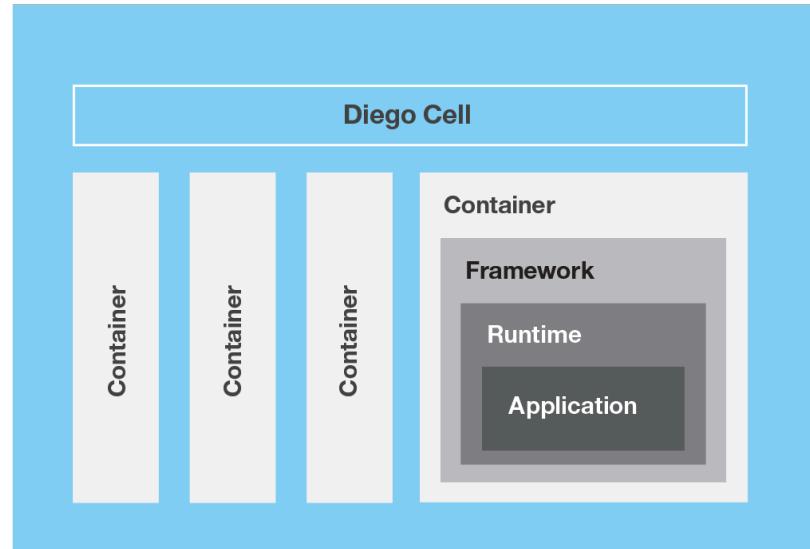
<b>Liberty for Java™</b> Develop, deploy, and scale Java web apps with ease. IBM WebSphere <small>IBM</small>	<b>SDK for Node.js™</b> Develop, deploy, and scale server-side JavaScript® apps with ease. <small>IBM</small>	<b>ASP.NET Core</b> Develop, deploy, and scale ASP.NET Core web apps with ease. <small>IBM</small>
<b>Runtime for Swift</b> A Kitura based server application that you can use as a starting point. <small>IBM</small>	<b>XPages</b> Develop, deploy and scale IBM XPages applications with ease. <small>IBM</small>	<b>Go</b> Develop, deploy, and scale Go web apps with ease. <small>Community</small>
<b>.php</b> Develop, deploy, and scale PHP web apps with ease. <small>Community</small>	<b>.py</b> Develop, deploy, and scale Python web apps with ease. <small>Community</small>	<b>.rb</b> Develop, deploy, and scale Ruby web apps with ease. <small>Community</small>
<b>Tomcat</b> Develop, deploy, and scale Tomcat web apps with ease. <small>Community</small>		

# Buildpacks provide support for a language runtime

## Buildpack

- collection of scripts
- app framework dependencies
- prepares runtime
- packages application into a droplet for deployment

Automatically selected



Simplifies Cloud deployment – just provide your application code



# What is Serverless?

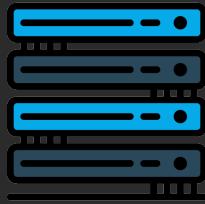
“Focus on your application & code, not the infrastructure”

“backend as a service”





Traditionally, we built our applications and controlled how they can communicate with a server. A server that we are **managing** and **provisioning** for. There are some issues that arose though.



1. Requires no management and operation of infrastructure, enabling developers to focus more narrowly on code/custom business logic.
2. It's a computing service **RUNS** code only on-demand on a per-request basis, **scaling** transparently with the number of requests being served (event-based)
3. **\$\$**  
Enables end users to pay only for resources being used, never paying for idle capacity.

# Serverless Computing

SERVERLESS COMPUTING



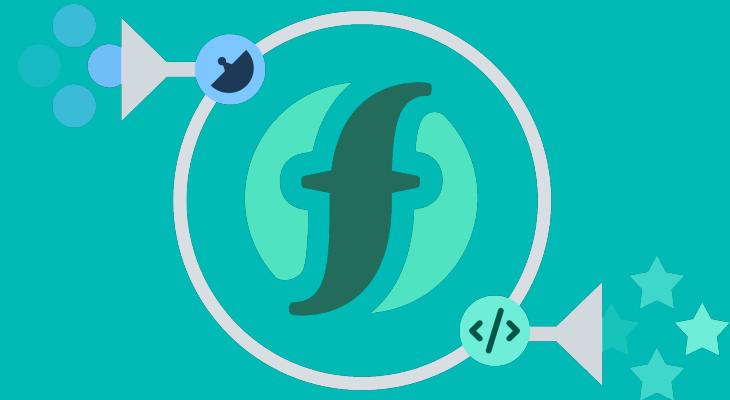
Does not refer to a specific technology.

Apache OpenWhisk



IBM's Cloud Functions started as the [OpenWhisk](#) research project at the IBM Watson Research center

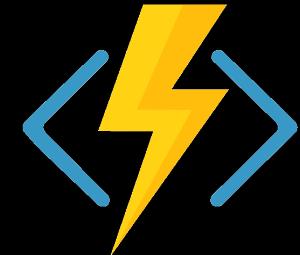
IBM Cloud Functions



AWS Lambda



Azure Functions



# Backend as a Service progression

IaaS	CaaS	PaaS	FaaS/Serverless
<b>Functions</b>	<b>Functions</b>	<b>Functions</b>	<b>Functions</b>
<b>Applications</b>	<b>Application</b>	<b>Applications</b>	<b>Applications</b>
<b>Runtimes</b>	<b>Runtimes</b>	<b>Runtimes</b>	<b>Runtimes</b>
<b>Operating Systems</b>	<b>Operating Systems</b>	<b>Operating Systems</b>	<b>Operating Systems</b>
<b>Infrastructure</b>	<b>Infrastructure</b>	<b>Infrastructure</b>	<b>Infrastructure</b>

Provided by  
Developer

Virtualization  
Provided by Cloud

Functions-as-a-Service (FaaS) platform based on Apache OpenWhisk

Run your application code without servers, scale it automatically, and pay nothing when it's not in use.

[Download CLI](#)[Start Creating](#)

## Runtimes

Work with what you already know and love. Develop your functions directly in your most favorite language, or provide us with a Docker container to develop using any compiled language such as Go, C, etc.



</> JS/NodeJS

</> Go

</> Swift

</> PHP

</> Java

</> .NET

</> Python

</> Any language (Docker)

</> Ruby



# Responsibilities with traditional servers

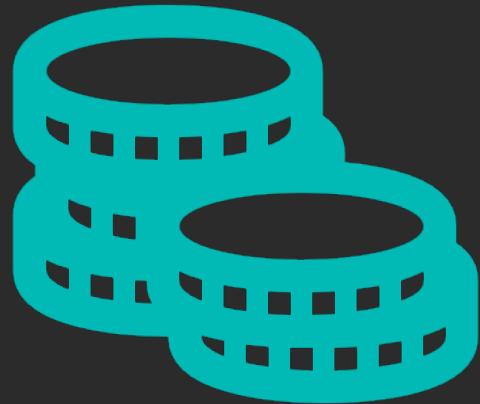
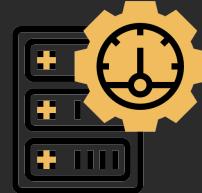
- Costs



- Maintenance

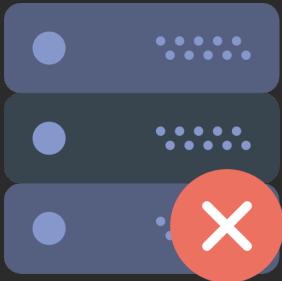


- Scaling up and down according to the application usage

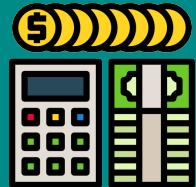




**Serverless computing refers to a model where the existence of servers is entirely abstracted away.**



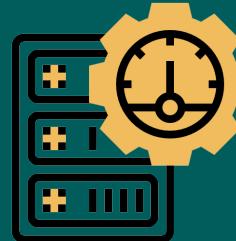
### Cost Effective Computing



### Low maintenance

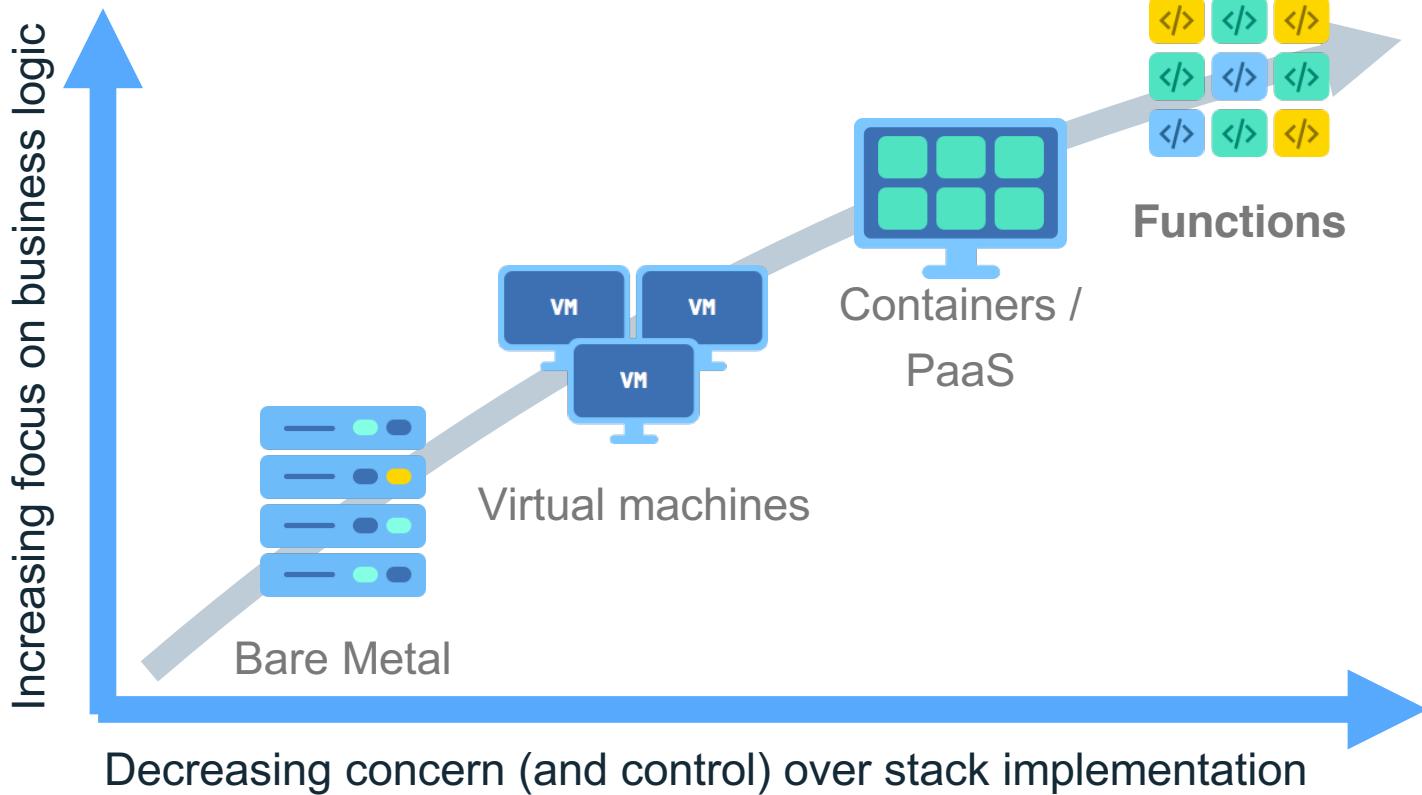


### Easy to scale



# What is Serverless computing (Functions-as-a-Service)?

SERVERLESS COMPUTING



## Event Providers..

Benefit from an ecosystem of event consumers and emitters from different areas like analytics, cognitive, data, IoT, mobile, and more. Using our open event-provider interface, [you can enable](#) any service you would like to use.



Github



Event Streams



Periodic



Cloud Object storage



Cloudant



Mobile Push

## Runtimes..

Work with what you already know and love. Develop your functions directly in your most favorite language, or provide us with a Docker container to develop using any compiled language such as Go, C, etc.



JS/NodeJS



Go



Swift



PHP



Java



.NET



Python



Any language (Docker)



Ruby

# What can I use Cloud functions for ?

## Cognitive Data Processing

Analyze data as soon as it becomes available. Let your function make use of powerful cognitive services like IBM Watson to detect objects or people appearing in images or videos.

[Case-study: Skylink \(drone image analyzing\)](#)



## Data Processing

Execute code whenever data is updated in your datastore. Easily automate processes like audio normalization, image rotation, sharpening, noise reduction, thumbnail generation, or video transcoding.



## IoT Ready

React to and process IoT sensor data. Let any IoT device send data to our IBM Watson IoT platform and define cloud rules to call your functions and execute custom application logic.



## Serverless Backends

Expose application logic by implementing serverless microservices. Simply map your functions to well-defined API endpoints any client can call by making use of Web Actions or our latest API Gateway integration.

[Go to tutorial](#)



## Mobile Backend

Allow mobile developers to easily access server-side logic and to outsource compute-intensive tasks to a scalable cloud platform. Let them implement functions in languages like Swift and easily consume server-side functions using our iOS SDK.



## Scheduled Tasks

Execute your functions periodically. Define schedules following a cron-like syntax to specify when actions are supposed to be executed.



