

# Introduction to Serverless PHP

Rob Allen

June 2018

# Deployment options

1. Physical servers
2. Virtual machines
3. Containers

# Container deployments

1. Platform (e.g. Kubernetes)
2. Application (e.g. Cloud Foundry)
3. Serverless (e.g. OpenWhisk)

# Serverless?

*The first thing to know about serverless computing is that "serverless" is a pretty bad name to call it.*

– Brandon Butler, Network World

# *AKA: Functions as a Service*

- A runtime to execute your functions
- No capacity planning or load balancing; just tasks being executed.
- Pay for execution, not when idle

# Use-cases

## Synchronous

Service is invoked and provides immediate response  
(HTTP requests: APIs, chat bots)

## Asynchronous

Push a message which drives an action later  
(web hooks, timed events, database changes)

## Streaming

Continuous data flow to be processed

# Benefits

- No need to think about servers
- Concentrate on application code
- Pay only for what you use, when you use it
- Language agnostic: NodeJS, Swift, Python, Java, C#, etc

# Challenges

- Start up latency
- Time limit
- State is external
- DevOps is still a thing



# It's about value



**Beau** @BeauVrolyk · 30m

Replying to @akrabat @kelseyhightower

1) "Serverless" is a point on the path to true app isolation. Apps want to just run, their authors don't care about infrastructure at all.



1



2



**Beau** @BeauVrolyk · 29m

Replying to @akrabat @kelseyhightower

2) The App author should not need to know, anymore than a Journalist knows about printing presses or what the voltage of the power used.



1



1



2



**Beau** @BeauVrolyk · 25m

Replying to @akrabat @kelseyhightower

3) We are relearning what was known in the time-share days. Pricing needs to be based on something customers value, not infa. items like VMs

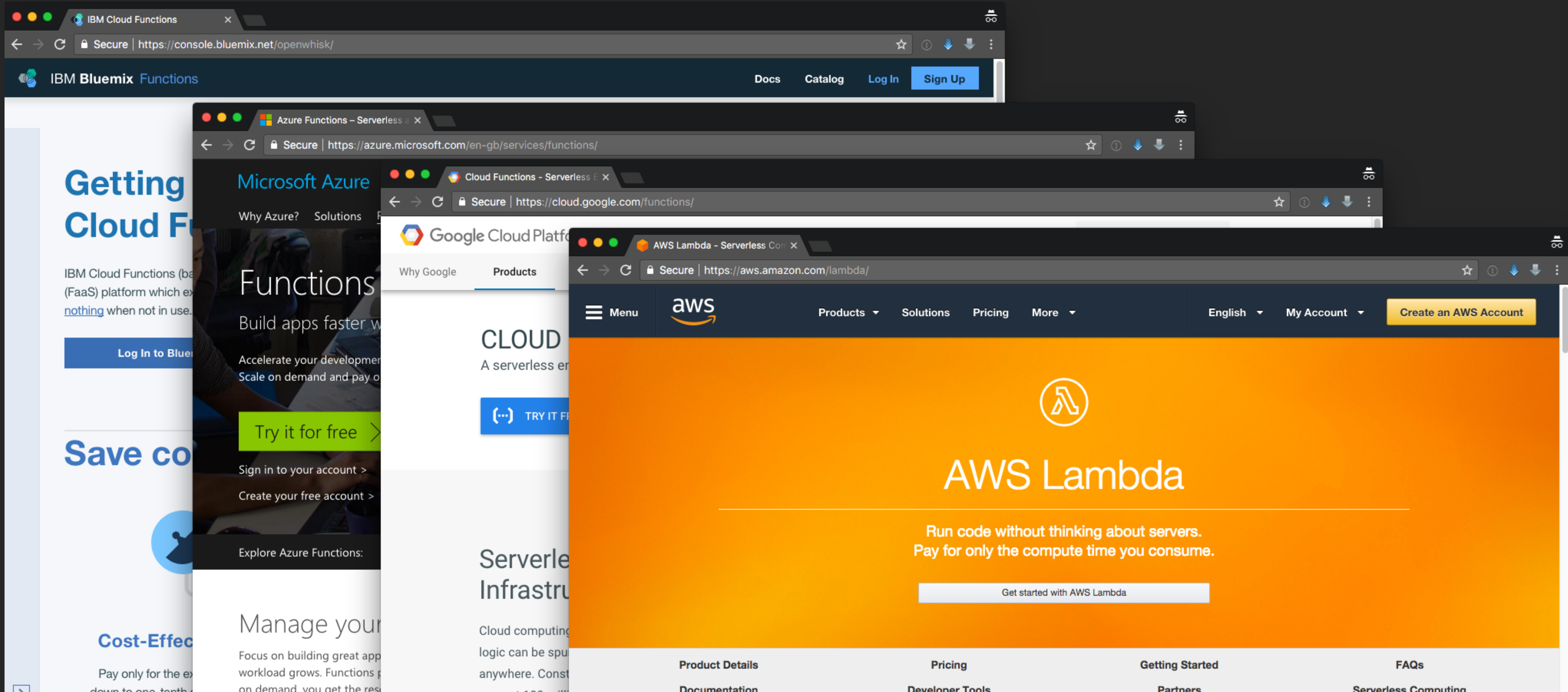


# When should you use serverless?

- Occasional server needs on a static site
- Variable traffic levels
- Additional compute without extending current platform
- Responding to web hooks



# Serverless providers



# OpenWhisk

# OpenWhisk

OpenSource; multiple providers:

IBM

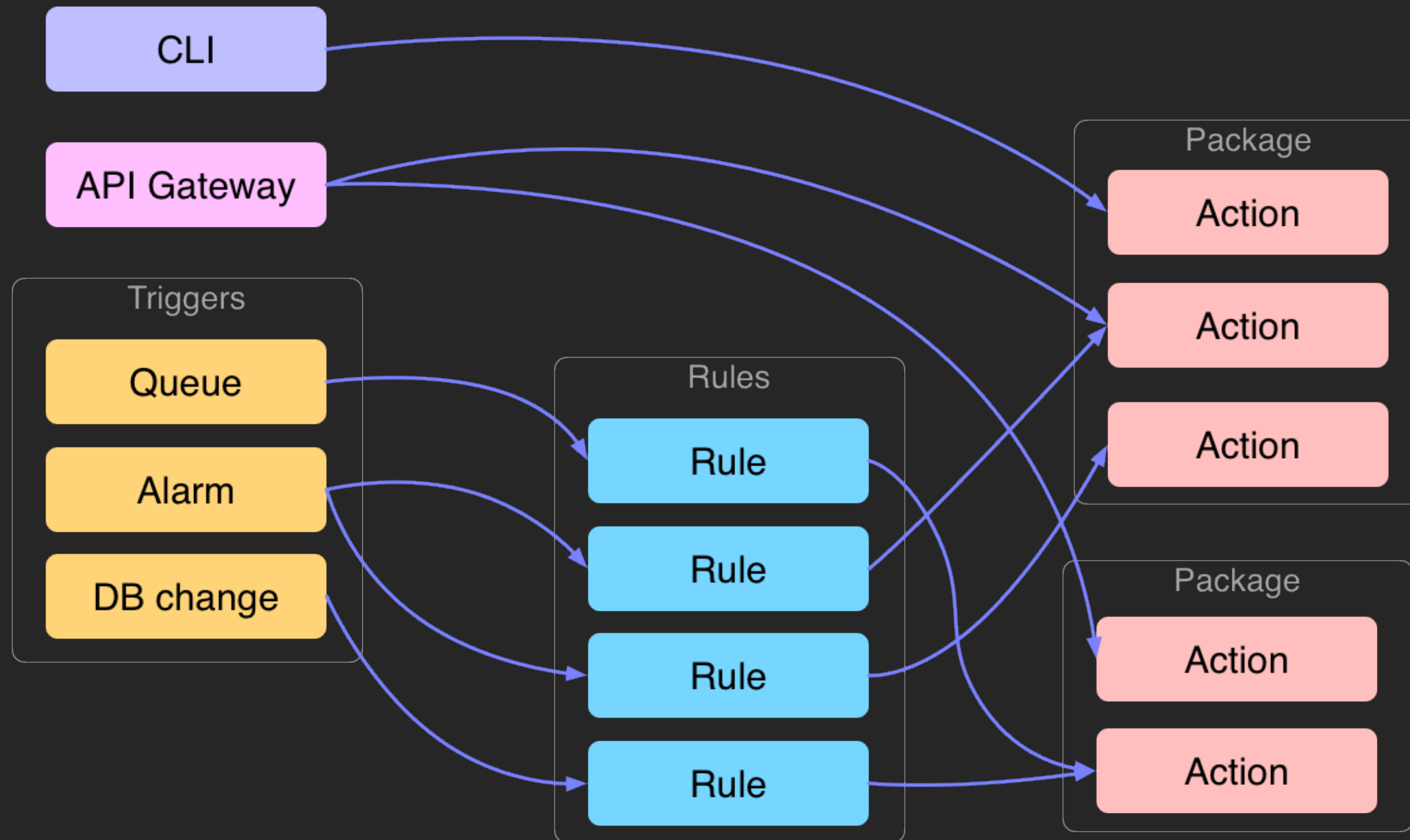
RedHat

Adobe (for Adobe Cloud Platform APIs)

&, of course, self-hosted



# Invoking an action



# Serverless PHP

# Hello world in PHP

```
1 <?php
2 function main(array $args) : array
3 {
4     $name = $args["name"] ?? "World";
5
6     return [ "msg" => "Hello $name" ];
7 }
```



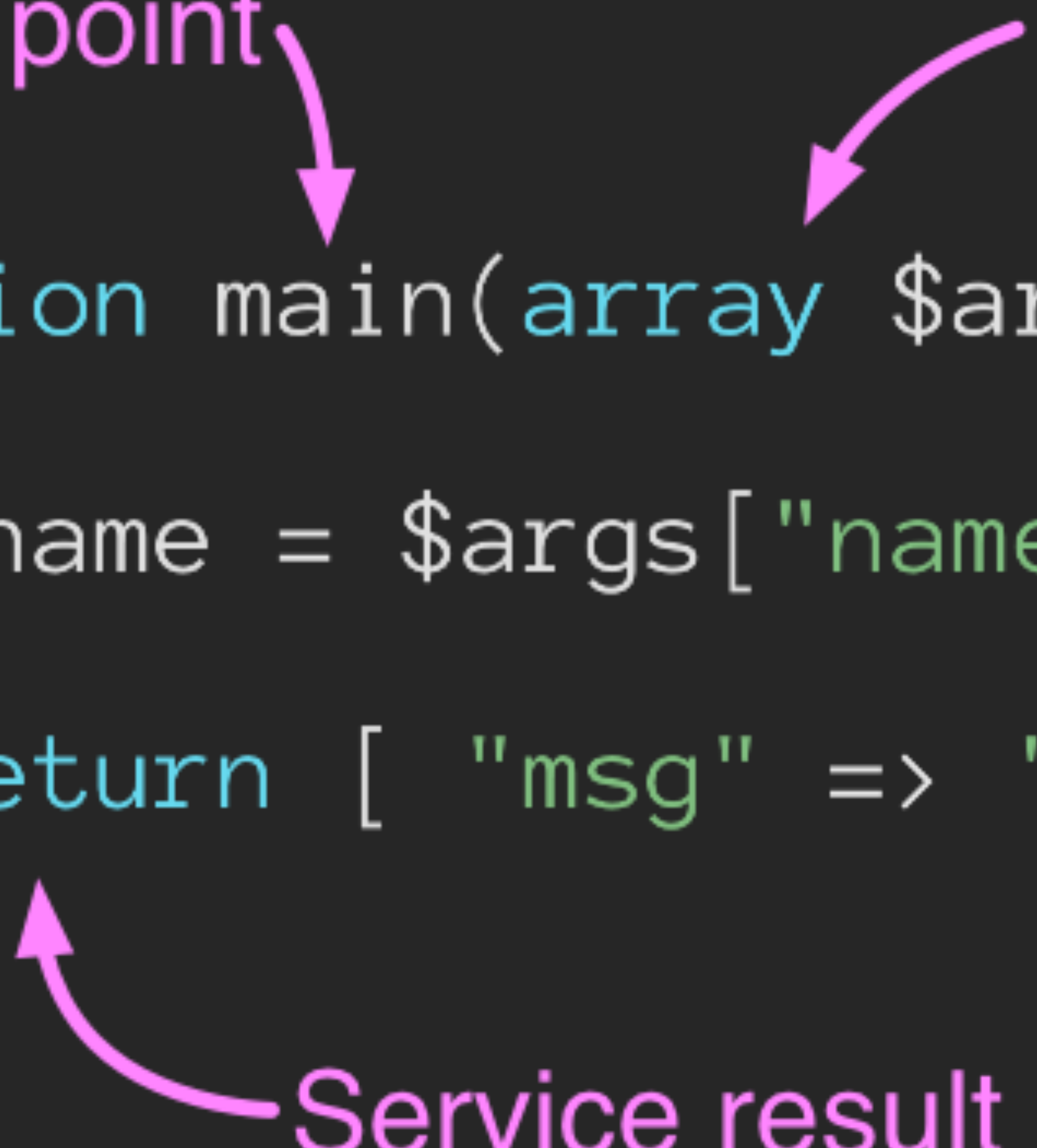
# Hello world in PHP

Entry point

Event parameters

```
1 <?php
2 function main(array $args) : array
3 {
4     $name = $args["name"] ?? "World";
5
6     return [ "msg" => "Hello $name" ];
7 }
```

Service result



# Running your action

```
$ wsk action update hello hello.php  
ok: updated action hello
```

```
$ wsk action invoke hello --result  
{  
  "msg": "Hello World"  
}
```

# Dependencies

Zip them up

```
$ zip -r hello.zip hello.php vendor
```

```
$ wsk action update hello hello.zip --kind php:7.1
```

# Web access

Add the `--web` flag:

```
$ wsk action update hello hello.php --web true  
$ curl https://openwhisk.ng.bluemix.net/api/v1/web/ \ 19FT_demo/default/hello.json
```

# What to do in your action

- Compute!
- Store to database
- Make API calls to other services
- Store to cloud storage (S3)
- Trigger other actions

# Demo time!

To sum up

# Resources

- <http://www.openwhisk.org>
- <https://medium.com/openwhisk>
- <https://github.com/akrabat/ow-php-ftime>
- <https://www.martinfowler.com/articles/serverless.html>



# Questions?

Thank you!