Building serverless APIs

Julien Simon, Principal Technical Evangelist, AWS
julsimon@amazon.fr
@julsimon

amazon web services

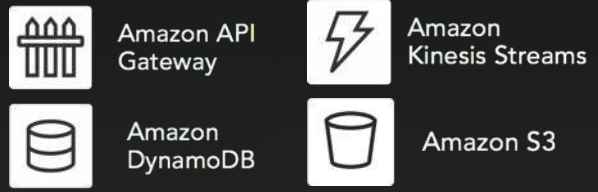
Serverless architecture

=

Managed services

+

AWS Lambda



Reference architectures: http://www.allthingsdistributed.com/2016/06/aws-lambda-serverless-reference-architectures.html

amazon web services

AWS Lambda

- Announced at re:Invent 2014
- Deploy pure functions in Java, Python, Node.js and C#
- Just code, without the infrastructure drama
- Built-in scalability and high availability
- Integrated with many AWS services
- Pay as you go
  - Combination of execution time (100ms slots) & memory used
  - Starts at $0.000000208 per 100ms
  - Free tier available: first 1 million requests per month are free

http://aws.amazon.com/lambda
AWS re:Invent 2014 | (MBL202) NEW LAUNCH: Getting Started with AWS Lambda https://www.youtube.com/watch?v=UFj27laTWQA

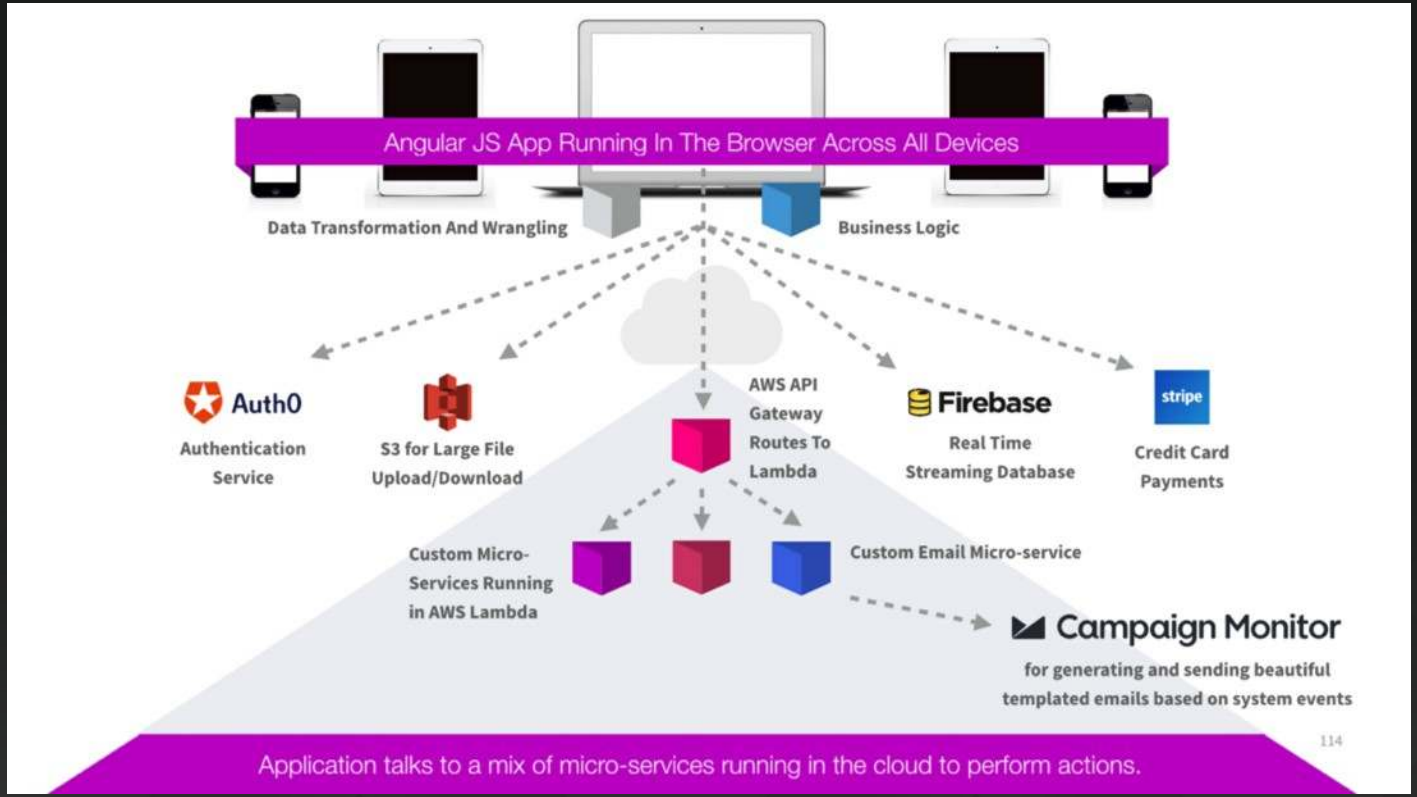
amazon web services

What can you do with AWS Lambda?

- Grow ‘connective tissue’ in your AWS infrastructure
  - Example: http://www.slideshare.net/JulienSIMON5/building-a-serverless-pipeline
- Build event-driven applications
- Build APIs together with Amazon API Gateway
  - RESTful APIs
  - Resources, methods
  - Stages

http://aws.amazon.com/apigateway

A Cloud Guru: 100% Serverless



https://read.acloud.guru/serverless-the-future-of-software-architecture-d4473ffed864

Typical development workflow

1. Write and deploy a Lambda function
2. Create and deploy a REST API with API Gateway
3. Connect the API to the Lambda function
4. Invoke the API
5. Test, debug and repeat ;)

Simplifying Development

Code samples available at https://github.com/juliensimon/aws/tree/master/lambda_frameworks

The Serverless framework

formerly known as JAWS: Just AWS Without Servers

- Announced at re:Invent 2015 by Austen Collins and Ryan Pendergast
- Supports Node.js, as well as Python and Java (with restrictions)
- Auto-deploys and runs Lambda functions, locally or remotely
- Auto-deploys your Lambda event sources: API Gateway, S3, DynamoDB, etc.
- Creates all required infrastructure with CloudFormation
- Simple configuration in YML

http://github.com/serverless/serverless
https://serverless.com
AWS re:Invent 2015 | (DVO209) https://www.youtube.com/watch?v=D_U6luQ6l90 & https://vimeo.com/141132756

amazon web services

Serverless: “Hello World” API

```
$ serverless create

Edit handler.js, serverless.yml and event.json

$ serverless deploy [--stage stage_name]

$ serverless invoke [--local] --function function_name

$ serverless info

$ http $URL
```

Gordon

- Released in Oct'15 by Jorge Batista
- Supports Python, Javascript, Golang, Java, Scala, Kotlin (including in the same project)
- Auto-deploys and runs Lambda functions, locally or remotely
- Auto-deploys your Lambda event sources: API Gateway, CloudWatch Events, DynamoDB Streams, Kinesis Streams, S3
- Creates all required infrastructure with CloudFormation
- Simple configuration in YML

https://github.com/jorgebastida/gordon
https://news.ycombinator.com/item?id=11821295

Gordon: “Hello World” API

```
$ gordon startproject helloworld

$ gordon startapp helloapp

Write hellofunc() function

$ gordon build

$ echo '{"name":"Julien"}' | gordon run helloapp.hellofunc

$ gordon apply [--stage stage_name]

$ http post $URL name=Julien
```

AWS Chalice

Think of it as a serverless framework for Flask apps

- Released in Jul'16, still in beta
- Just add your Python code
  - Deploy with a single call and zero config
  - The API is created automatically, the IAM policy is auto-generated
- Run APIs locally on port 8000 (similar to Flask)
- Fast & lightweight framework
  - 100% *boto3* calls (AWS SDK for Python) → fast
  - No integration with CloudFormation → no creation of event sources

https://github.com/awslabs/chalice
https://aws.amazon.com/blogs/developer/preview-the-python-serverless-microframework-for-aws/

AWS Chalice: “Hello World” API

```
$ chalice new-project helloworld

Write your function in app.py

$ chalice local

$ chalice deploy

$ export URL=`chalice url`

$ http $URL

$ http put $URL/hello/julien

$ chalice logs [ --include-lambda-messages ]
```

amazon web services

AWS Chalice: PUT/GET in S3 bucket

```
$ chalice new-project s3test

Write your function in app.py

$ chalice local

$ http put http://localhost:8000/objects/doc.json value1=5 value2=8

$ http get http://localhost:8000/objects/doc.json

$ chalice deploy [stage_name]

$ export URL=`chalice url`

$ http put $URL/objects/doc.json value1=5 value2=8

$ http get $URL/objects/doc.json
```

amazon web services

Summing things up

Serverless

The most popular serverless framework

Built with and for Node.js. Python and Java: YMMV

Rich features, many event sources

Not a web framework

Gordon

Great challenger!

Node.js, Python, Java, Scala, Golang

Comparable to Serverless feature-wise

Not a web framework

Chalice

AWS project, in beta

Python only

Does only one thing, but does it great

Dead simple, zero config

Flask web framework

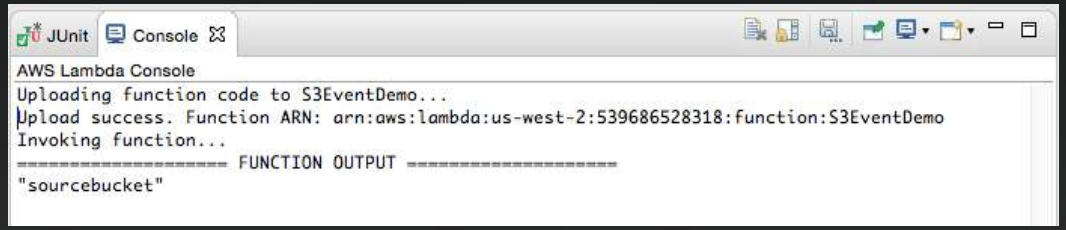
More Lambda frameworks

- Kappa https://github.com/garnaat/kappa
  - Released Dec'14 by Mitch Garnaat, author of boto and the AWS CLI (still maintained?)
  - Python only, multiple event sources
- Apex https://github.com/**apex**/**apex**
  - Released in Dec'15 by TJ Holowaychuk
  - Python, Javascript, Java, Golang
  - Terraform integration to manage infrastructure for event sources
- Zappa https://github.com/Miserlou/Zappa
  - Released in Feb'16 by Rich Jones
  - Python web applications on AWS Lambda + API Gateway
- Docker-lambda https://github.com/lambci/docker-lambda
  - Released in May'16 by Michael Hart
  - Run functions in Docker images that "replicate" the live Lambda environment

2 Java tools for AWS Lambda

Eclipse plug-in

- Code, test and deploy Lambdas from Eclipse
- Run your functions locally and remotely
- Test with local events and Junit4
- Deploy standalone functions, or with the AWS Serverless Application Model (Dec'16)



Serverless Java Container

- Run Java RESTful APIs as-is
- Default implementation of the Java servlet `HttpServletRequest` `HttpServletResponse`
- Support for Java frameworks such as Jersey or Spark

https://java.awsblog.com/post/TxWZES6J1RSQ2Z/Testing-Lambda-functions-using-the-AWS-Toolkit-for-Eclipse
https://aws.amazon.com/blogs/developer/aws-toolkit-for-eclipse-serverless-application
https://github.com/awslabs/aws-serverless-java-container

Simplifying Deployment

AWS Serverless Application Model (SAM)

- CloudFormation extension released in Nov'16 to bundle Lambda functions, APIs & events
- 3 new CloudFormation resource types
  - *AWS::Serverless::Function*
  - *AWS::Serverless::Api*
  - *AWS::Serverless::SimpleTable* (DynamoDB)
- 2 new CloudFormation CLI commands
  - '*aws cloudformation package*'
  - '*aws cloudformation deploy*'
- Integration with CodeBuild and CodePipeline for CI/CD
- Expect SAM to be integrated in most / all frameworks

https://aws.amazon.com/fr/blogs/compute/introducing-simplified-serverless-application-deplyoment-and-management
https://github.com/awslabs/serverless-application-model/

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Description: Get items from a DynamoDB table.
Resources:
  GetFunction:
    Type: AWS::Serverless::Function
    Properties:
      Handler: index.get
      Runtime: nodejs4.3
      Policies: AmazonDynamoDBReadOnlyAccess
      Environment:
        Variables:
          TABLE_NAME: !Ref Table
      Events:
        GetResource:
          Type: Api
          Properties:
            Path: /resource/{resourceId}
            Method: get
  Table:
    Type: AWS::Serverless::SimpleTable
```

Sample SAM template for:

- Lambda function
- HTTP GET API
- DynamoDB table

Lambda@Edge

preview

The AWS edge

68 POPs

9 Regional Edge Caches

43 Cities

21 Countries

5 Continents

North America

South America

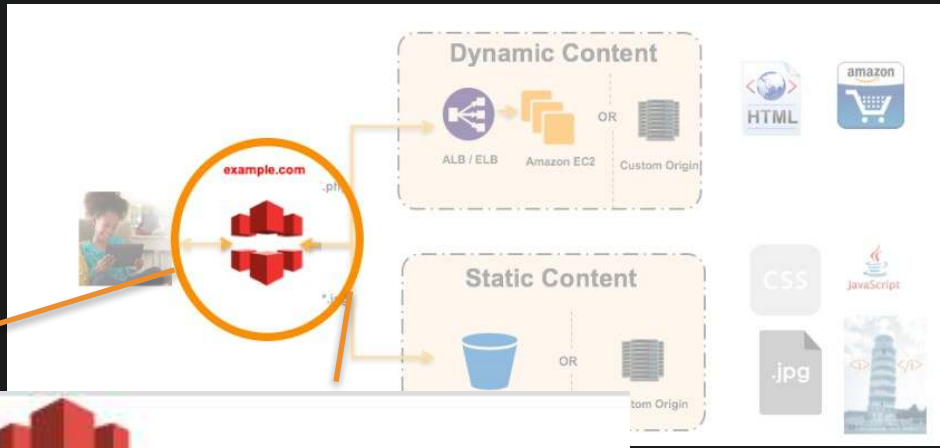
EMEA

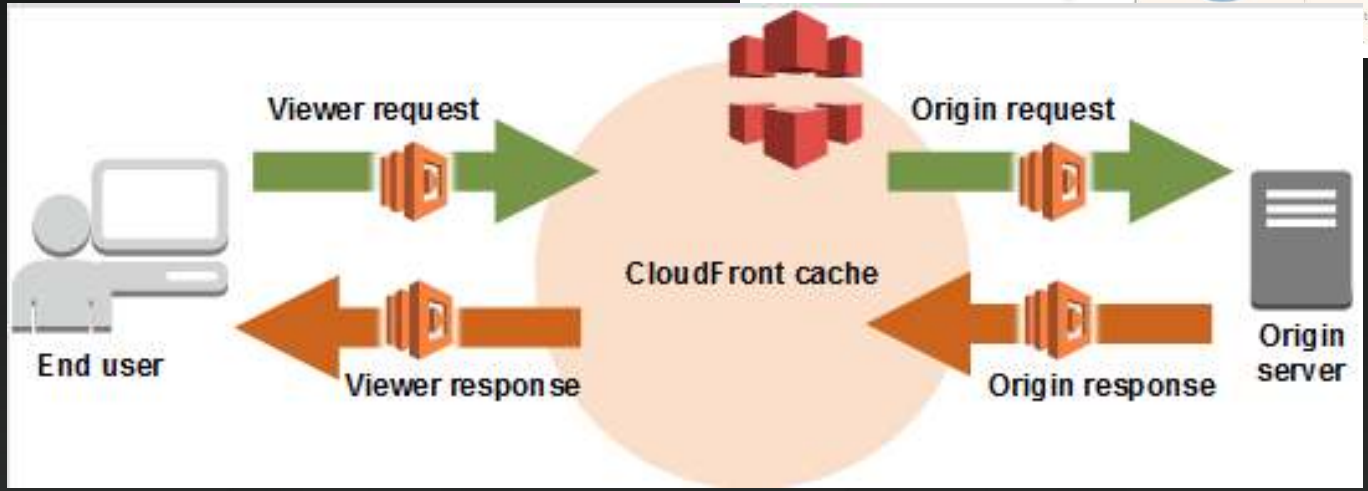
APAC

Lambda@Edge

- Lambda@Edge is an extension of AWS Lambda that allows you to run Node.js code at AWS global edge locations.
- Bring your own code to the edge and customize your content very close to your users, improving the end-user experience.
- Typical use cases
  - Customize content, based on user/device properties
  - Validate visitors: check tokens, filter bot traffic
  - Rewrite URLs: inject ads, hide file structure
  - Run A/B testing

CloudFront Triggers for Lambda@Edge Functions





Lambda@Edge Service Limits

- Runtime: Node.js 4.3
- Triggered by CloudFront events
- Access: No network connections, AWS region access, disk access or Amazon VPC

Items	Lambda@Edge	Lambda
Timeout	50 ms	300 seconds
Function “Power Level”	128 MB	128 MB – 1.5 GB
Function Deployment Package Size	1MB	50MB

Lambda@Edge demo: the Metal A/B test \m/

- Static website in S3, distributed to CloudFront

```
http://d2eb9d90qk3238.cloudfront.net
```

- The index page references an image file that doesn't exist in the S3 bucket

```
<img src="http://d2eb9d90qk3238.cloudfront.net/album.jpg">
```

- A Lambda function is triggered on the "Origin request" event and replaces `album.jpg` with a random file

Going further

New Lambda videos from re:Invent 2016

AWS re:Invent 2016: What's New with AWS Lambda (SVR202)https://www.youtube.com/watch?v=CwxWhyGteNc

AWS re:Invent 2016: Serverless Apps with AWS Step Functions (SVR201)
https://www.youtube.com/watch?v=75MRve4nv8s

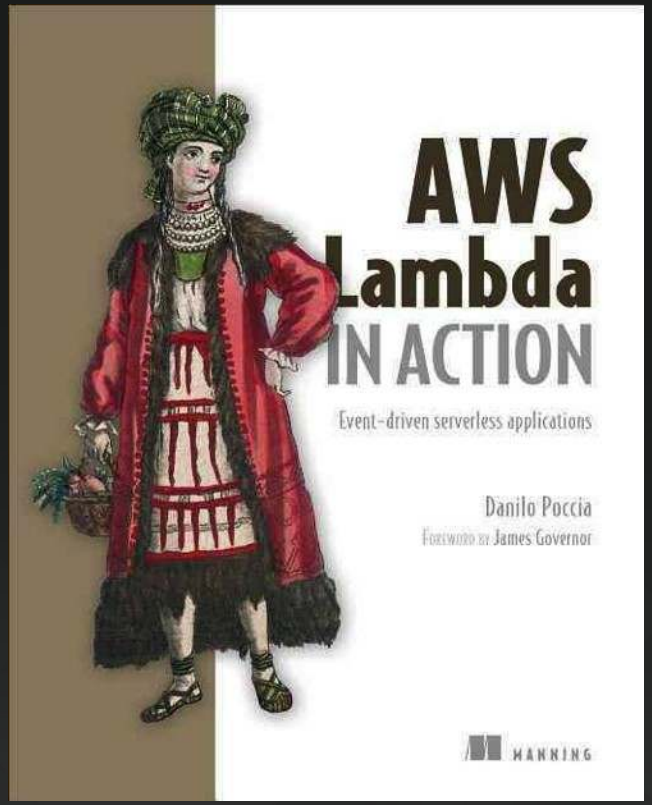
AWS re:Invent 2016: Real-time Data Processing Using AWS Lambda (SVR301)
https://www.youtube.com/watch?v=VFLKOy4GKXQ

AWS re:Invent 2016: Serverless Architectural Patterns and Best Practices (ARC402)
https://www.youtube.com/watch?v=b7UMoc1iUYw

AWS re:Invent 2016: Bringing AWS Lambda to the Edge (CTD206)
https://www.youtube.com/watch?v=j26novaqF6M

AWS re:Invent 2016: Ubiquitous Computing with Greengrass (IOT201)
https://www.youtube.com/watch?v=XQQjX8GTEko

The only Lambda book you need to read



Written by AWS Technical Evangelist Danilo Poccia

Released in December 2016

https://www.amazon.com/Aws-Lambda-Action-Event-driven-Applications/dp/1617293717/

amazon web services

Thank you!

Julien Simon, Principal Technical Evangelist, AWS
julsimon@amazon.fr
@julsimon

amazon web services