codestock

Exploring Serverless Architectures w/ AWS Lambda

Alex Klibisz, Codestock 2016, Knoxville, TN

Intro

- Alex Klibisz
- Computer Science at UTK
- Web development, machine learning
- Used AWS Lambda for StudyLoop
- @alexklibisz, alex.klibisz.com

Goals

1. Understand pros and cons of serverless architectures.
2. Use the “serverless” framework for a productive workflow.
3. Architect a small application (with code).

Agenda

1. Why serverless apps?
2. What is AWS Lambda?
3. AWS Lambda pitfalls
4. Serverless framework
5. Serverless framework demo
6. Use-cases, tools, and libraries

Why Serverless?

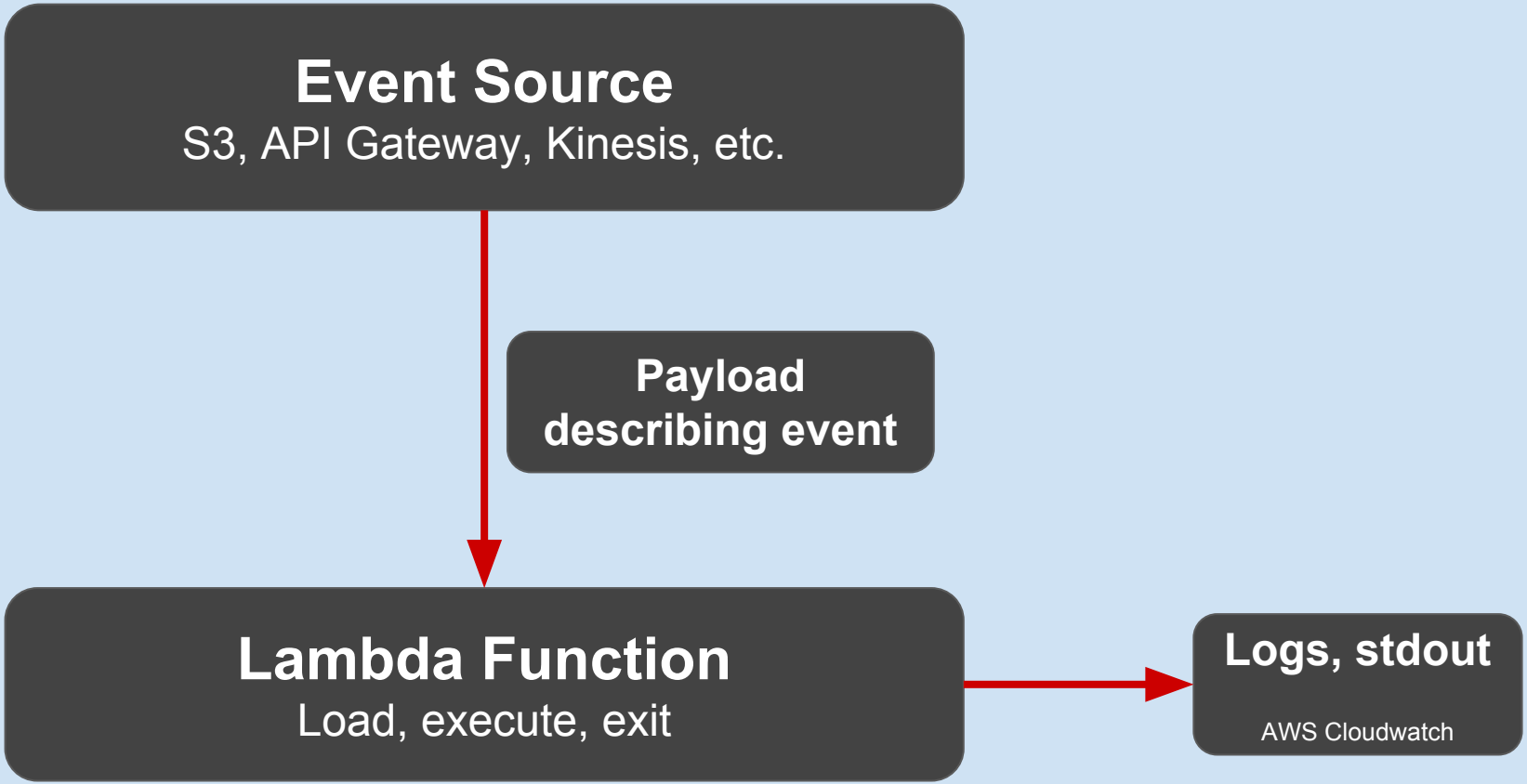
- Serverless: *caring less about the server*

- Accommodate inconsistent loads
- Save money (pay by execution time)
- Save time (reduced server configuration, monitoring)

What is AWS Lambda?

1. Platform offered by AWS.
2. Functions in Java, Python, Node.js.
3. Respond to events in AWS.
4. Pay for execution time.

See Docs: *Lambda Pricing Details*


Event Source
S3, API Gateway, Kinesis, etc.
Payload describing event
Lambda Function
Load, execute, exit
Logs, stdout
AWS Cloudwatch

Event Sources

- S3
- DynamoDB
- Kinesis
- Simple Notification Service
- Simple Email Service
- Cognito
- CloudFormation
- Cloudwatch
- AWS Config
- Echo
- API Gateway

See Docs: *Event Examples*

Amazon S3 Put Sample Event

```
"Records": [
  {
    "eventVersion": "2.0",
    "eventTime": "1970-01-01T00:00:00.000Z",
    "requestParameters": {
      "sourceIPAddress": "127.0.0.1"
    },
    "s3": {
      "configurationId": "testConfigRule",
      "object": {
        "eTag": "0123456789abcdef0123456789abcdef",
        "sequencer": "0A1B2C3D4E5F678901",
        "key": "HappyFace.jpg",
        "size": 1024
      },
      "bucket": {
        "arn": bucketarn,
        "name": "sourcebucket",
        "ownerIdentity": {
          "principalId": "EXAMPLE"
        }
      },
      "s3SchemaVersion": "1.0"
    },
```

AWS Lambda Pitfalls

- Tedious deployment process
- Unfamiliar testing methodology
- Runtime environment restrictions
- No persistent state or file system
- Latency while functions are loaded to execute

... Not a silver bullet

So we need tooling.

Serverless Framework

- Released as *JAWS* ~ May 2015, later *Serverless*
- CLI application, runs locally
- Open-source, written in Node.js
- Automated configuration (cloud formation, regions, environments)
- Create, deploy Node.js, Python Lambda functions
- Run simple tests locally
- Map functions to endpoints and events
- Plugins ecosystem

Serverless Caveats

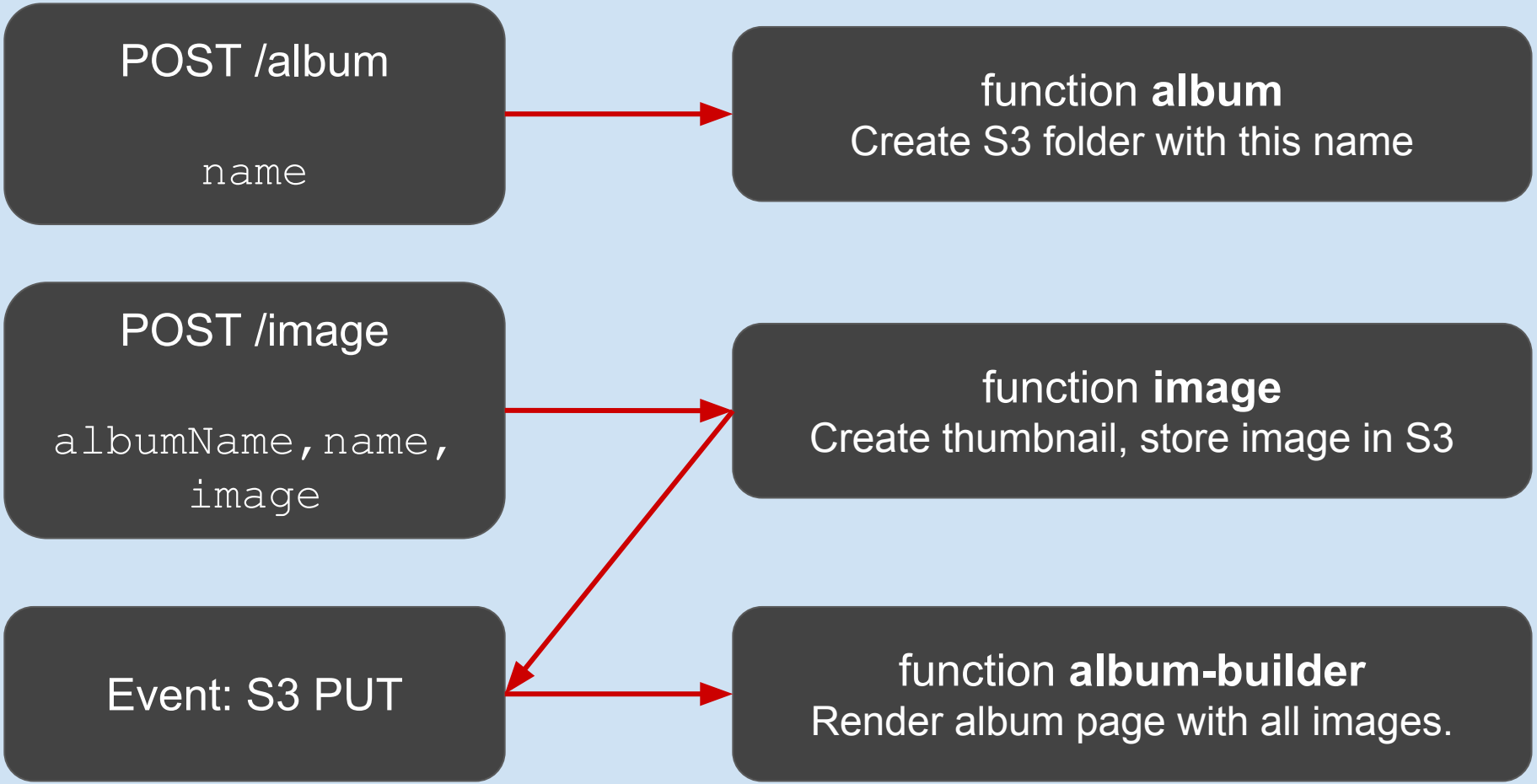
- Prioritizes HTTP API use-cases
- Counterintuitive response statuses (error returns a 200 status)
- Complexity for multiple developers
- Quickly evolving = breaking changes, documentation falls behind

Serverless Demo

“Lambda Albums”

- Static site hosted on S3
- Upload images via Lambda endpoints
- Store images, thumbnails on S3
- Render SEO-friendly static galleries to S3
- Inspired by article *Making static websites less static: S3 cloud, AWS Lambda, and a rough one-day hack* - *airpair.com*
- www.lambda-albums.xyz

Let’s see it in action.


POST /album
name
function **album**
Create S3 folder with this name
POST /image
albumName,name,
image
function **image**
Create thumbnail, store image in S3
Event: S3 PUT
function **album-builder**
Render album page with all images.

Let’s see the code.

Use-cases (educated opinions)

- Good: fire-and-forget interaction with AWS services
  - Log, stream, media processing
  - Document generation
  - Sending notifications

- Bad: realtime or low-latency constraints

Other Tools, Libraries

- Apex: run Go and other non-supported languages
- Zappa: serverless python web services
- Shep: node.js on AWS lambda
- Claudia Bot Builder: bots for FB messenger, skype, etc.
- LambCI: continuous integration on AWS Lambda

codestock

- *Exploring Serverless Architectures*, CodeStock 2016
- Alex Klibisz
- @alexklibisz, alex.klibisz.com

- Slides and code: alex.klibisz.com/pages/talks