Assumptions:

1. You have installed and configured AWS CLI
2. You have a third party mail service setup already (we use mailgun for this example).

The entire process for this simple effort requires a handful of steps:

1. Install SAM Local
2. Define your Lambda in AWS Console (can also do so using CLI)
3. Write your function!

SAM Local

First, let's install **SAM Local**.

SAM Local is an amazing tool written by *Paul Maddox* from the AWS team that allows you to build and test Lambda functions locally.

```
npm install -g aws-sam-local
```

Later, we'll use `event.json` as a sample file we test our function with, so build that quickly; for example:

```
{
    Name: "Jake"
}
```

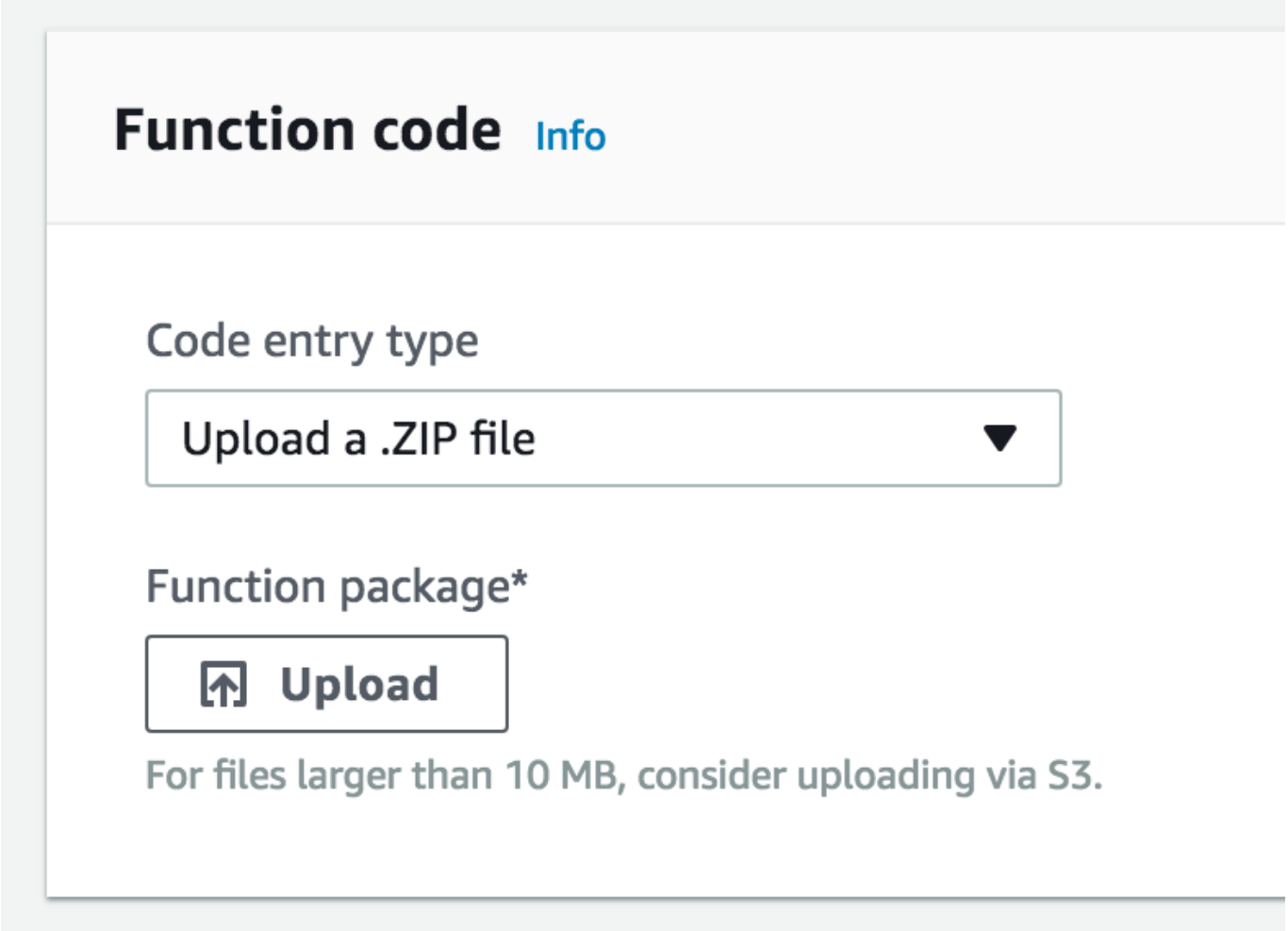
Define your Lambda in AWS Console

You can easily simplify this process using the `aws cli` but for this we'll simply share a simple step-by-step that outlines the function for a first time developer.

1. Create a New Function
    1. Author from scratch
    2. Name it something you'll recognize later
    3. Runtime *Node.js 6.1.0*
    4. Role *Choose an Existing Role*
    5. Select a Role that has policies to execute Lambda functions, and access SSM Parameter Store (and anything else your function will be doing).
2. Function Designer
    1. Select S3 from the left column
    2. Scroll down to *Configure Triggers*
    3. Select an appropriate bucket, and then make sure you have the trigger **enabled**.

Write your Function

Once complete, we'll be zipping our `node_modules` and `index.js` file and uploading them to the Function via the console.




```
'use strict';

const aws = require('aws-sdk');
const promise = require('bluebird');
const mailgun = require('mailgun.js');

const ssm = promise.promisifyAll(new aws.SSM({ region: 'us-west-2'
}));
const s3 = new aws.S3({ apiVersion: '2006-03-01' });

exports.handler = (event, context, callback) => {
```

```
    //('Received event:', JSON.stringify(event, null, 2));

    let options = {
        Names: [
            'mailgun_api_key',
        ],
        WithDecryption: true
    };

    ssm.getParameters(options, function (err, data) {
        if (err) {
            console.log(err, err.stack);
        } else {

            let params = {};

            data.Parameters.forEach(function (param) {
                params[param.Name] = param.Value;
            });

            var mailer = mailgun.client({username: 'api', key:
params['mailgun_api_key']});

            // Get the object from the event and show its content type
            const bucket = event.Records[0].s3.bucket.name;
            const key =
decodeURIComponent(event.Records[0].s3.object.key.replace(/+/g, ' '));

            var url = "http://" + bucket + ".s3.amazonaws.com/" + key;

            s3.getObject({
```

```
                Bucket: bucket,
                Key: key,
            }, (err, data) => {
                if (err) {
                    //console.log(err);
                    const message = "Error getting object " + key + "
from bucket " + bucket + ". Make sure they exist and your bucket is in
the same region as this function.";
                    console.log(message);
                    callback(message);
                } else {
                    mailer.messages.create('example.com', {
                        from: "Bot <bot@example.com>",
                        to: ["sales@example.com"],
                        subject: "New App Build: " + key,
                        text: "A new App build is complete: " + url,
                        html: "<h1>New App Build</h1><p>A new App
build is complete: " + url + "</p>"
                    })
                    .then(msg => console.log(msg)) // logs response
data
                    .catch(err => console.log(err)); // logs any error
                }
            });
        }
    });

};
```

Testing Your Function

First, SAM will need a `template.yml` file to work correctly.

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Description: When a file is added to `mybucket` email the dev team.
Resources:
  MyFunction:
    Type: AWS::Serverless::Function
    Properties:
      Handler: index.handler
      Runtime: nodejs6.10
      CodeUri: s3://mybucket/myapp.zip
      Policies: AmazonS3ReadOnlyAccess
      Events:
        PhotoUpload:
          Type: S3
          Properties:
            Bucket: !Ref Bucket
            Events: s3:ObjectCreated:*

  Bucket:
    Type: AWS::S3::Bucket
```

From here, you can test your function and iterate as you develop locally with the command below:

```
sam local invoke "MyFunction" -e event.json
```