Logging Incoming Events

In this lesson, you'll learn how to keep a continuous check on logs using the `--tail` option of the `sam logs` command.

An amazingly useful option of `sam logs` is `--tail`. It will continuously check for updates and show new log outputs as they appear. Remember that CloudWatch isn't pulling information from a single source, so the tail operation isn't synchronous, but it is generally quick enough for troubleshooting.

Go ahead and let's try it out. You will run the `sam logs` command in tail mode and keep it open. Then, you'll add some more logging to the function so you can see the outputs.

```
sam logs -n HelloWorldFunction --stack-name sam-test-1 --tail
```

Next, open the function source code from `hello-world/app.js` and add a line to log the event details just below the function start (add line 3 from the following example into your function):

```
let response;
exports.lambdaHandler = async (event, context) => {
  console.log(JSON.stringify(event, null, 2));
```

You will build, package, and deploy the function again when you press the **Run** button. Check out Chapter 3 if you need a refresher on how it is done.

When the deployment completes, you have to run the `sam logs` command provided below. Then, reload the web page that triggers the Lambda function (you can find it in your stack outputs).

You should see a JSON object for the request appear in the console window where you are tailing the logs.

Here is the `sam logs` command that you will run after you click on the **Run**

button and a terminal appears:

```
sam logs -n HelloWorldFunction --stack-name sam-test-1 --tail
```

Don't forget to make the specified changes to `app/helloworld/app.js`.

If you make changes to any of the files below, press the Run button to register the changes.

Environment Variables

Key:	Value:
AWS_ACCESS_KEY_ID	Not Specified...
AWS_SECRET_ACCE...	Not Specified...
BUCKET_NAME	Not Specified...
AWS_REGION	Not Specified...


```
{
  "body": "{\"message\": \"hello world\"}",
  "resource": "/{proxy+}",
  "path": "/path/to/resource",
  "httpMethod": "POST",
  "isBase64Encoded": false,
  "queryStringParameters": {
    "foo": "bar"
  },
  "pathParameters": {
    "proxy": "/path/to/resource"
  },
  "stageVariables": {
    "baz": "qux"
  },
  "headers": {
    "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8",
    "Accept-Encoding": "gzip, deflate, sdch",
    "Accept-Language": "en-US,en;q=0.8",
    "Cache-Control": "max-age=0",
    "CloudFront-Forwarded-Proto": "https",
    "CloudFront-Is-Desktop-Viewer": "true",
    "CloudFront-Is-Mobile-Viewer": "false",
    "CloudFront-Is-SmartTV-Viewer": "false",
    "CloudFront-Is-Tablet-Viewer": "false",
    "CloudFront-Viewer-Country": "US",
    "Host": "1234567890.execute-api.us-east-1.amazonaws.com",
    "Upgrade-Insecure-Requests": "1",
    "User-Agent": "Custom User Agent String",
    "Via": "1.1 08f323deadbeefa7af34d5feb414ce27.cloudfront.net (CloudFront)",
    "X-Amz-Cf-Id": "cDehVQoZnx43VYQb9j2-nvCh-9z396Uhbp027Y2JvkCPNLmGJHqlaA==",
    "X-Forwarded-For": "127.0.0.1, 127.0.0.2",
    "X-Forwarded-Port": "443",
    "X-Forwarded-Proto": "https"
  },
```

```
  "requestContext": {
    "accountId": "123456789012",
    "resourceId": "123456",

    "stage": "prod",
    "requestId": "c6af9ac6-7b61-11e6-9a41-93e8deadbeef",
    "requestTime": "09/Apr/2015:12:34:56 +0000",
    "requestTimeEpoch": 1428582896000,
    "identity": {
      "cognitoIdentityPoolId": null,
      "accountId": null,
      "cognitoIdentityId": null,
      "caller": null,
      "accessKey": null,
      "sourceIp": "127.0.0.1",
      "cognitoAuthenticationType": null,
      "cognitoAuthenticationProvider": null,
      "userArn": null,
      "userAgent": "Custom User Agent String",
      "user": null
    },
    "path": "/prod/path/to/resource",
    "resourcePath": "/{proxy+}",
    "httpMethod": "POST",
    "apiId": "1234567890",
    "protocol": "HTTP/1.1"
  }
}
```


Reloading the web page triggers the Lambda function and you will be able to see the corresponding logs in the terminal.

> You'll notice that there are other folders for each chapter in the widget above. These tutorial files evolve through chapters in this course, and the final version at the end of each chapter is in a separate directory of the source code package at https://runningserverless.com. For this lesson, you are using the code from scratch. Therefore, you can ignore chapter prefixes and keep editing the `app/hello-world/app.js` file. If you want to skip steps from the tutorial, chapter versions make it easy to start from a specific point in the tutorial or inspect solutions.

This is a very useful pattern for inspecting the format of incoming events. Various AWS services all have their own event formats when invoking Lambda functions, and usually, it helps to first just log the event to the console to see how to extract important information.

Hopefully, you were easily able to see the logs. Get ready to move on to the next lesson!