DOWNLOAD PDF BOOKS ON AWS LAMBA

Chapter 1 : AWS Lambda in Action - Event-Driven Serverless Applications

A short and well written introduction to AWS Lambda. Helping me grasp what it is, where it could be used, where it falls short. The book uses examples written in blog.quintoapp.com, and tells you how to test such functions locally before spending cheap yet valuable Lambda cycles.

What is AWS Lambda? After you upload your code and create what we call a Lambda function, AWS Lambda takes care of provisioning and managing the servers that you use to run the code. You can use AWS Lambda as follows: Lambda is a cool platform, but it only supports a few languages - Java, Node. Amazon has tons of documentation about Lambda, best practices, etc. Normally we are blissfully unaware of platform-specific deployments when writing Node. Much of this convenience is lost, however, when dealing with Amazon Lambda - we truly must pre-build our Node. If we are using a native addon, then this means when building our deployment packages, we must be on the same architecture and platform as Lambda bit, Linux , and we must specifically use the same Node. Therefore, at a minimum, we must build our addon on Linux to deploy. In the future these versions could change - so edit accordingly. I like to use nvm to install and switch between Node. The second will in fact be the Lambda function - it will be a Node. If you want to follow along on your own machine, all the source code is here - this particular example in the lambda-cpp folder. Lets start first with the addon. Again, if you are looking for more details on addon development in general - check out my other posts and my book. Just remember that anything you are linking too must be statically compiled into the resulting executable, and be built for x64 Linux. Now the source codeâ€¦ Lets open up average. We can build this by issuing a node-gyp configure build command. If you have everything setup correctly, you should see gyp info ok at the bottom of the output. You should delete test. As you probably already know, all AWS Lambda functions must expose a handler which gets called whenever an event is triggered. Your directory structure should be as follows: Lets create an event object and put it in sample. You do this through the IAM console. You can download your access key credentials as a csv file using the instructions in the AWS docs here. Once you have your access keys, you need to install the CLI. There are a few ways to do this, and it requires Python to be installed on your machine. Type aws configure and enter your access key and secret key downloaded a moment ago. You can also choose a default region and output format. In other words, if you uzip the zip file you create, index. You need to build the addon and zip it on the right platform see requirements aboveâ€¦ Linux, x64, etc. Inside the directory containing index. Verify the package with less. You should see something along the lines of this: Make sure you update the region accordingly for your setup: If all goes well, you should receive a JSON response with some additional info about the newly deployed function for example, FunctionArn. Invoke the function, specifying the same event as last time which is called payload by the CLI. The LogResult is encoded in base64, so you can do the following: For now, go ahead and add some printouts to your index. Since our Lambda addon function is geared towards simple computation, a next step might be to hook it up as an Gateway API. All of the code from this post, plus a whole bunch of other addon examples are in my nodecpp-demo repository.

Chapter 2 : AWS Lambda â€“ Pricing

AWS Lambda in Action is an example-driven tutorial that teaches you how to build applications that use an event-driven approach on the back-end. Starting with an overview of AWS Lambda, the book moves on to show you common examples and patterns that you can use to call Lambda functions from a web page or a mobile app.

AWS Lambda is a compute service that lets you run code without provisioning or managing servers. AWS Lambda executes your code only when needed and scales automatically, from a few requests per day to thousands per second. You pay only for the compute time you consume - there is no charge when your code is not running. With AWS Lambda, you can run code for virtually any type of application or backend service - all with zero administration. AWS Lambda runs your code on a high-availability compute infrastructure and performs all of the administration of the compute resources, including server and operating system maintenance, capacity provisioning and automatic scaling, code monitoring and logging. All you need to do is supply your code in one of the languages that AWS Lambda supports currently Node. With these capabilities, you can use Lambda to easily build data processing triggers for AWS services like Amazon S3 and Amazon DynamoDB, process streaming data stored in Kinesis, or create your own back end that operates at AWS scale, performance, and security. AWS Lambda is an ideal compute platform for many application scenarios, provided that you can write your application code in languages supported by AWS Lambda that is, Node. When using AWS Lambda, you are responsible only for your code. This is in exchange for flexibility, which means you cannot log in to compute instances, or customize the operating system or language runtime. These constraints enable AWS Lambda to perform operational and administrative activities on your behalf, including provisioning capacity, monitoring fleet health, applying security patches, deploying your code, and monitoring and logging your Lambda functions. If you need to manage your own compute resources, Amazon Web Services also offers other compute services to meet your needs. It gives you the option to customize operating systems, network and security settings, and the entire software stack, but you are responsible for provisioning capacity, monitoring fleet health and performance, and using Availability Zones for fault tolerance. Elastic Beanstalk offers an easy-to-use service for deploying and scaling applications onto Amazon EC2 in which you retain ownership and full control over the underlying EC2 instances. Lambda is a highly available service. If you are a first-time user of AWS Lambda, we recommend that you read the following sections in order: Read the product overview and watch the introductory video to understand sample use cases. These resources are available on the AWS Lambda webpage. Review the Lambda Functions section of this guide. To understand the programming model and deployment options for a Lambda function there are core concepts you should be familiar with. This section explains these concepts and provides details of how they work in different languages that you can use to author your Lambda function code. Try the console-based Getting Started exercise. The exercise provides instructions for you to create and test your first Lambda function using the console. You also learn about the console provided blueprints to quickly create your Lambda functions. For more information, see Getting Started. This section introduces various AWS Lambda components you work with to create an end-to-end experience. Beyond the Getting Started exercise, you can explore the various use cases, each of which is provided with a tutorial that walks you through an example scenario. Depending on your application needs for example, whether you want event driven Lambda function invocation or on-demand invocation , you can follow specific tutorials that meet your specific needs. For more information, see Use Cases. The following topics provide additional information about AWS Lambda:

Chapter 3 : blog.quintoapp.com: AWS Lambda: A Guide to Serverless Microservices eBook: Matthew Fulle

Amazon Web Services is Hiring. Amazon Web Services (AWS) is a dynamic, growing business unit within blog.quintoapp.com We are currently hiring Software Development Engineers, Product Managers, Account Managers, Solutions Architects, Support Engineers, System Engineers, Designers and more.

Instead of loading your application code into a virtual machine or a container , you upload it into Lambda. There it sits, dormant, until some external event triggers it, whereupon the Lambda service brings your app out of quiescence and executes it. Once the application completes its task, the code is automatically removed from the Lambda service. All previous forms of computing have required that the application operations team manage the execution environment that runs the application code. Not with Lambda, which has ushered in the era of serverless computing. Even though Lambda has been available as a fully released product for only just over a year, it has generated a lot of experimentation and even production use. And, in a sign of how promising the technology is, all of the other big cloud providersâ€"including Microsoft, Google, and IBMâ€"have rushed their own versions of Lambda to market. But what are serverless applications? After all, the code has to run somewhere. What serverless does is shift the responsibility for running that environment from the user to the cloud providerâ€"and in that simple sentence a revolution resides. Moving the responsibility upward Serverless computing moves the responsibility interface upward, by a lot. With Lambda, AWS takes on the responsibility of making sure that the application code gets loaded and executed, and it ensures that sufficient computing resources are available to run your Lambda code, no matter how much processing it requires. Lambda is itself based on containers. The limits AWS imposes on the Lambda user include: The code must be less than MB uncompressed, or 75 MB compressed It must run for no more than five minutes It can access no more than MB of ephemeral storage This last limit points out something quite salient about Lambda: Its fundamental design assumption is that your code can run in a bounded fashion, with no need for significant resources and no need to run for extended timeframes. To my mind, there are three significant benefits. The cloud made it possible to deploy an application without ever worrying about hardware. Lambda extends that trend and further lightens the load on application developers. In turn, this lets IT organizations increase their focus on the most important part of IT: If you believe that software is eating the world, then sloughing off low-value activities to allow greater investment in high-value application functionality is a huge win for IT, and for the organization it serves. It increases the efficiency of IT spend Even though cloud computing got rid of the infrastructure investment, everybody knows someone who lets cloud virtual machines run endlessly while doing no productive work. Managing capacity, even virtual capacity, is difficult. Lambda eliminates the need to do that. A good fit for event-driven applications Finally, the Lambda paradigm of executing code only when needed is a good fit for an emerging category of event-driven applications that will represent an ever larger proportion of corporate application portfolios in the future. Event-driven applications may support the IoT or user-driven content submission think Snapchat. Such applications are event-driven, episodic, and subject to unpredictable traffic patterns. With serverless, all that wasted money comes back to you. You pay only for what you use and avoid the headache of virtual machine instances and EBS altogether. So how much can you save on your cloud computing bills? At Serverlessconf, two presentations on Lambda-based applications indicated that they had not even surpassed the daily free limit on Lambda calls. In other words, they were operating inside the Lambda free tier. This, despite the fact that both presenters were discussing real-world applications with significant levels of use, as opposed to prototype or proof-of-concept efforts. Perhaps the most astonishing thing about Lambda and its serverless counterparts at other cloud providers is how willing these organizations are to cannibalize their own offerings. Once people comprehend the financial benefits associated with serverless computing, there will be strong growth in interest, experimentation, and adoption. Welcome to the post-virtual machine, post-container world Cloud computing has brought enormous change to the world of applications. It makes long-standing constraints on application development and deployment disappear. It promises to change application paradigms even more than cloud computing has done and holds out the tantalizing possibility of moving to a post-virtual machine,

post-container world. Likewise, serverless computing is going to break a lot of existing practices and processes.

Chapter 4 : blog.quintoapp.com - Recursive invocations / fork invocations in AWS Lambda - Stack Overflow

If you are a Cloud administrator and/or developer who wishes to explore, learn, and leverage AWS Lambda to design, build, and deploy Serverless applications in the cloud, then this is the book for you!

Create Serverless Microservices with Node. AWS is a collection of developer tools that Amazon develops and publicly offers. To follow along with this article you will need an AWS account of your own. You can create a free AWS account at https: Where or how does the code actually run then? It is used for describing solutions for on-demand code execution. You just upload your code to a FaaS provider AWS Lambda, in this case and the FaaS provider executes it and manages any infrastructure for you behind the scenes. Pros On-demand usage pricing. Traditional server hosting uses a recurring billing cycle. Your server is always up and running using resources and waiting for input. You pay a monthly or yearly fee to keep it running for the duration of your billing cycle. Lambda uses on-demand pricing, where billing is calculated per function use. This means that if your project utilizes Lambda functions that are not being used in high demand, you can save a significant amount of money over traditional hosting solutions. Lambda pricing is as follows: As the traffic and usage of your application increase, you may need to add more hosted servers to your infrastructure to keep up with demand. This can cause failures and bottlenecks for your users. Lambda takes care of scaling silently when needed, removing additional cognitive overhead. Cons Inconsistent local development work flow. Lambda Key Concepts Function code and triggers Lambda has two main concepts: Once uploaded, code will not execute on its own. Some example triggers are: For more info see: Click the blue button Create a Lambda function to get started. Select a blueprint The next screen should prompt you to Select a blueprint and present a list of filter-able blueprints. Click the Blank Function option, it should be first in the list of blueprints. This page can used for future reference to check other utilities of Lambda. Configure trigger The following screen is Configure Triggers and it should look like: Click Next to bypass this screen. Optionally, you can also give a description to the function. You should be presented with an inline editor that has a sample function that looks like this: You can follow the steps below, or feel free grab the code from the example repository. Defining a function handler that matches the Lambda signature. With the response form Github, a map is created that includes url and star count for each repo. I used the command line zip utility on OSX, like this: Configure a function handler and role Under this section we want to set a few values. Handler is the reference to your Lambda function in the uploaded JavaScript file. By default this is set to index. This maps to our uploaded code by looking for a file index which in our case is index. Create new role from template s Role name: If everything looks OK, click the Create function button to continue. Assigning a trigger to our new function Now that our function has been created and initialized, we need a way to invoke it. For our trigger we will be using API Gateway. After the trigger is successfully added, you should see it attached to your function under the Triggers tab. Although we used a 3rd party client GitHub integration within our function code, this can be replaced by any other client API or a database client connection. Serverless frameworks The process of setting up Lambda that has been demonstrated in this article is very manual and somewhat ephemeral, but there are other ways to configure and initialize Lambda driven by the AWS API. However, at the writing of this article, the framework is undergoing a major version upgrade from 0. Unfortunately, this upgrade is not backwards compatible. The majority of popular plugins from 0. Serverless, with its working plugins, promises to deliver a very comprehensive Lambda experience. It provides local development environment setup for rapid iteration, automated Lambda code deployment, multiple deployment staging environments, and much more. OpenLambda attempts to simulate a Lambda environment by providing a local development experience. It also offers tools to make it easier to deploy Lambda code, allowing for rapid iteration. This remedies one of the cons of Lambda listed above. What do you think? Will you consider FaaS for future projects? Please share your thoughts in the comments below! Front end web developer.

Chapter 5 : AWS Lambda | PDF Free Download

AWS Lambda lets you quickly and easily build and run applications on the AWS cloud that automatically run code in response to events. AWS Lambda makes back-end tasks like producing a thumbnail from a new image or processing requests from a mobile app simple to implement, ready for web-scale traffic.

Azure Cloud Functions vs. Before you make any major decisions, keep in mind that these services are evolving. Supported Languages This will change over time for every cloud function service. Azure Functions supports Node. AWS Lambda supports Node. I think PHP support is a very strategic choice given its popularity, but I wish one of these services would better support Go. Because it runs fast and its code is easy to read. It is a language of brevity and therefore fits well for the cloud and micro services. Add Rust and Swift to the list too. This is the method in which Go works in Lambda. Both Apex and also Sparta are serverless frameworks that let you use Go in Lambda as well. It would be great for a cloud function service to support Go natively. Web Dashboard No amount of fancy devops tricks or tooling will save you from having to use the web dashboard at least every now and then. Once you understand the organizational structure of Azure Function it becomes a little easier. The same can be said for AWS Lambda too. One important difference is the editor. However, Azure has the more robust Visual Studio Online which can also be used. Most of your development time should be outside of dashboards anyway. This is much different than Lambda. This is the biggest similarity between Lambda and Azure Functions. The pricing model for this is then like EC2. For example, you can set environment variables on App Services which are then available for your Azure Functions. Lambda now can as pointed out by Jeremy Axmacher , thank you. The entire container architecture is different. Lambdas provision a brand new one and deploys your code from a zip file on a cold request. Subsequent requests can be subject to container re-use and be handled much faster. However, you need to understand there is no persistence and with Node. Azure Functions of course is a Windows system 32bit for dynamic services while Lambda is a Linux system. Though I do hope that changes since Ubuntu can now run on Windows. In fact, that would put Azure in a good position with cloud hosting. Continuous Deployment Speaking of working with the files, you can deploy your Azure Functions in a variety of ways: Git you can connect GitHub, Bitbucket, etc. Lambda has some great 3rd party tools for deploying Lambdas and both cloud services have SDKs to make just about anything possible. I do have to hand it to Microsoft here though. You can hook up GitHub, even with your favorite CI tool, and easily deploy your cloud functions to Azure. People have created tools to test your Lambda locally so you can write code, run, and then even deploy if all looks good. Regardless, Lambda is closed source. I would expect some clever CI integrations in the future. Every function automatically maps to an HTTP endpoint if enabled. API Gateway is fine, but complex and time consuming. Again, some serverless frameworks ease this pain point by automatically setting up an API for Lambdas. Microsoft gets points for UX because you have a lot less to configure. I would say that AWS Lambda has a slight edge in flexibility here, but both services are constantly changing. The Azure app and function names are reflected in the URLs to trigger the cloud functions. This helps avoid confusion, but might also be limiting. With API Gateway, you need to configure each method manually. It seems that everything must be passed as a querystring or in the request body. You can configure CORS and you can also manage your own domain name for the endpoints. These are two serverless frameworks that address this specific organizational concern. They help a developer logically group functions. Another pain point with Lambda is the maximum execution time limit. It also requires EC2 instances which are billed in a different fashion. You can write and execute the same exact code regardless of the environment and billing model. There are apparently no time limits either so long as your function is not idle. On the other hand, I imagine there could be some surprises in billing too. Will Microsoft keep billing you for some accidental loop in a cloud function? Assuming the same amount of resources were used, will the cost work out to be about the same regardless of billing models? There is some chatter about this concern though if you follow the GitHub issue. First, since there is persistence and no execution time limits, working with large files and ETL is easier. Just watch out for how cost effective this ends up being or not being. With Azure Functions, the content-type is set by the code.

This can be very useful, but could also lead to some bloated functions. Remember, your goal is not to build a complete application within a cloud function. So be careful not to fall into any bad practices. Again, Azure Functions run on a server with a persistent filesystem and webroot. However, it is accessible to you while logged in. All of your code and assets can be seen via a URL like this: Can you somehow open this to the public? It definitely might raise an eye-brow or two though. You have more insight, access to and control over the environment running your cloud function than you do with AWS Lambdaâ€¦For better or worse. Speaking of insight, when listing the directory of D: The concurrent execution limit per instance is a bit unknown. Or does each Azure account gets its own, very large, storage volume? I would encourage everyone to explore and try each of these services. Understand why and when they they can help you. The end result can be a game changer for you too. The cloud has powerful magic, but not all clouds are the same. Get to know the environment your code lives in as well as the capabilities of the services available to you. Read the fine print. Understand the limitations and tradeoffs. Only with this information can you make an informed decision. Remember, your choice will be the path less taken here. The jury is still out. I use a variety of services because I believe there is strength and opportunity in diversity. I love AWS and will continue to use it happily, but I definitely think these companies can learn a thing or two from each other. Less than ideal monitoring and debugging. There will be trial and error. You are subject to the possibility of having to re-build code and move your operations. If it gives you any solace, you will be in good company. Some major companies and smart developers are on board with cloud services and the micro service movement. This is the new frontier on the web. Buy the ticket, take the ride.

Chapter 6 : What Is AWS Lambda? - AWS Lambda

The book assumes you have some prior knowledge and hands-on experience with AWS core services such as EC2, IAM, S3, along with the knowledge to work with any popular programming language such as blog.quintoapp.com, Java, C#, and so on.

If you buy something we get a small commission at no extra charge to you. AWS is a cloud computing system used for various web and database platforms. AWS may seem like just another cloud system but it has a variety of tools and features. Thankfully there are plenty of books to help you get started creating your own cloud apps. Plus this book covers so much that it can be useful for both beginners and intermediate users. It is very detailed and much easier to read than the official AWS documentation. However this book does not teach you detailed specifics like how to spin up a server environment or how to scale your app on AWS. This book really gets into the details about cloud computing and the Amazon hosting platform. However the later chapters get into much more detail including databases and server infrastructure. Later chapters teach you how to scale webapps on AWS and how to handle fault tolerance on your cloud system. All-in-all I think this book is meant for existing developers who have little knowledge of AWS. This is the absolute best title for a complete beginner who needs a walkthrough of the AWS platform. The writing style is a bit terse and very technical. Anyone should be able to pick up this book and get going with AWS. The early chapters help you sign up for a free developer account and spin up an EC2 instance from scratch. All chapters include relevant source code and most of the writing follows a step-by-step tutorial process. The authors also share tips for using the terminal and the GUI, two different AWS interfaces that support a wide readerbase. No matter where you are with cloud hosting or AWS this book offers the best guide for beginners and most intermediate users too. Lambda is more of a managed system and while it does have similar features, it also has differences in the interface and setup. Every chapter is incredibly detailed and the writing style caters to complete beginners. You should have no problem picking up this book and reading it cover to cover. I recommend this title for beginners or intermediate developers who love working on a serverless platform. If you read this book all the way through you will gain tremendous confidence in managing your own cloud projects. Each service has about pages devoted to explaining the features, how they work, and how to get started. Danilo is an experienced user and worked hard to organize this book in a clear-cut fashion. Unfortunately the first couple chapters are very complex and full of information. But the directions are crystal clear so once you understand the basics of Lambda you should have no problem working through these examples. However there are plenty of chapters covering deployment with all the supported languages Java, Python, and Node. The book has several authors and all of them are AWS experts some even working for Amazon. While you may not need a study guide for the exam, this is absolutely the best one you can get. It does cover some of the basics of a serverless environment, but it delves more into the key features of AWS administration. This page guide covers the basics of configuring and maintaining an AWS system. Any systems administrator or DevOps team would greatly benefit from reading this book. AWS is quickly becoming the norm for cloud computing and this book will get you up to speed without any pointless drivel. AWS Scripted Scripting and automation is the way of the future. This applies to all areas including hosting which is where AWS Scripted comes into play. This is the 1st book in a series of two books covering the basics of scripting and automation on AWS. Early chapters cover automation for spinning up an instance and installing certain programs. All source code is completely free and easy to save for future use. But while the code is great this book also teaches a lot about AWS as a framework for secure cloud computing. The solution is a dev-targeted book like Programming Amazon EC2. Every chapter offers both a CLI solution and a web interface solution. Anyone with some basic dev knowledge should be able to complete this book and learn a lot about EC2 in the process. My biggest complaint is how little this book covers security. It works great as an intro guide for web developers but this is not the tome of EC2 development. More experienced users should try to find books that cater to your area of expertise. But in truth all of these books are fantastic, they just approach different topics from different angles. Jaime Morrison Jaime is a jr. He covers general news and useful resources in the web design space.

DOWNLOAD PDF BOOKS ON AWS LAMBA

Chapter 7 : Deploy a Sample Custom Skill to AWS Lambda | Custom Skills

AWS Lambda is an ideal compute platform for many application scenarios, provided that you can write your application code in languages supported by AWS Lambda (that is, blog.quintoapp.com, Java, Go and C# and Python), and run within the AWS Lambda standard runtime environment and resources provided by Lambda.

Chapter 8 : Putting Amazon Lambda to work with Kinesis blog.quintoapp.com

With AWS Lambda, you can run code without provisioning or managing servers. You pay only for the compute time that you consumeâ€”there's no charge when your code isn't running.

Chapter 9 : Amazon AWS Lamba Python Script | Amazon Web Services | Python | Software Architecture

Amazon Web Services (AWS) is a dynamic, growing business unit within blog.quintoapp.com We are currently hiring Software Development Engineers, Product Managers, Account Managers, Solutions Architects, Support Engineers, System Engineers, Designers and more.