Amazon Lambda

Sterling
Talent Solutions

Layman Definition for AWS Lambda

Lambda is method/function which can be written in programming language and have to be deployed on AWS environment like website we are hosting on web server.

What is AWS Lambda?

- AWS Lambda is a serverless compute service that runs your code in response to events and automatically manages the underlying compute resources for you.
- AWS Lambda lets you run code without provisioning or managing servers. You pay only for the compute time you consume - there is no charge when your code is not running.

Supporting Programming Languages

- C#
- Java 8
- Node.js 4.3
- Node.js 6.10
- Python 2.7
- Python 2.6

Key Features

- Support Server Less Framework
- Extend Other AWS Services with Custom Logic
- Build Custom Back-end Services
- Completely Automated Administration
- Built-in Fault Tolerance
- Automatic Scaling
- Bring Your Own Code
- Pay Per Use
- Flexible Resource Model

Key Features - Support Server Less Framework

Serverless computing allows you to build and run applications and services without thinking about servers. With serverless computing, your application still runs on servers, but all the server management is done by AWS. At the core of serverless computing is AWS Lambda, which lets you run your code without provisioning or managing servers

Key Features-Extend Other AWS Services with Custom Logic

AWS Lambda allows you to add custom logic to AWS resources such as Amazon S3 buckets and Amazon DynamoDB tables, making it easy to apply compute to data as it is enters or moves through the cloud.

Sterling
Talent Solutions

Key Features-Build Custom Back-end Services

You can use AWS Lambda to create new back-end services for your applications that are triggered on-demand using the Lambda API or custom API endpoints built using Amazon API Gateway.

Key Features-Completely Automated Administration

AWS Lambda manages all the infrastructure to run your code on highly available, fault-tolerant infrastructure, freeing you to focus on building differentiated back-end services. With Lambda, you never have to update the underlying OS when a patch is released, or worry about resizing or adding new servers as your usage grows. AWS Lambda seamlessly deploys your code, does all the administration, maintenance, and security patches, and provides built-in logging and monitoring through Amazon CloudWatch.

Key Features-Built-in Fault Tolerance

Lambda has built-in fault tolerance. AWS Lambda maintains compute capacity across multiple Availability Zones in each region to help protect your code against individual machine or data center facility failures. Both AWS Lambda and the functions running on the service provide predictable and reliable operational performance. AWS Lambda is designed to provide high availability for both the service itself and for the functions it operates. There are no maintenance windows or scheduled downtimes.

Key Features-Automatic Scaling

AWS Lambda invokes your code only when needed and automatically scales to support the rate of incoming requests without requiring you to configure anything. There is no limit to the number of requests your code can handle. AWS Lambda typically starts running your code within milliseconds of an event, and since Lambda scales automatically, the performance remains consistently high as the frequency of events increases. Since your code is stateless, Lambda can start as many instances of it as needed without lengthy deployment and configuration delays.

Key Features-Bring Your Own Code

With AWS Lambda, there are no new languages, tools, or frameworks to learn. You can use any third party library, even native ones. AWS Lambda supports Java, Node.js, C#, and Python code, with support for other languages coming in the future.

Key Features-Pay Per Use

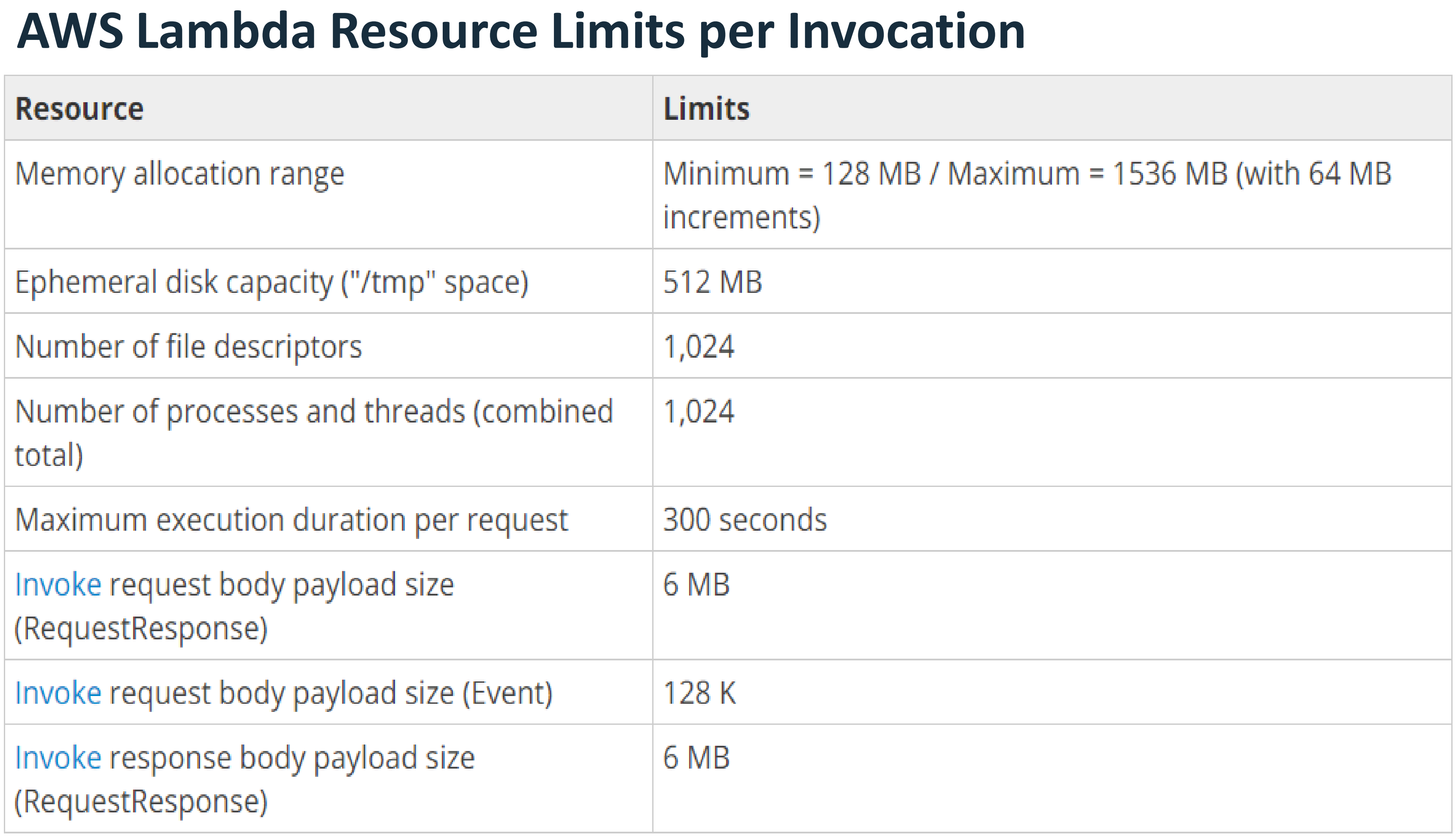
With AWS Lambda you pay only for the requests served and the compute time required to run your code. Billing is metered in increments of 100 milliseconds, making it cost-effective and easy to scale automatically from a few requests per day to thousands per second.

AWS - Interact with other AWS Service

1. API-Gateway
2. IOT
3. Cloudwatch-Events
4. Cloudwatch-Logs
5. CodeCommit
6. Cognito
7. DynamoDB
8. Kinesis
9. S3
10. SNS

Sterling
Talent Solutions

Limitation



AWS Lambda Resource Limits per Invocation

Resource	Limits
Memory allocation range	Minimum = 128 MB / Maximum = 1536 MB (with 64 MB increments)
Ephemeral disk capacity ("/tmp" space)	512 MB
Number of file descriptors	1,024
Number of processes and threads (combined total)	1,024
Maximum execution duration per request	300 seconds
Invoke request body payload size (RequestResponse)	6 MB
Invoke request body payload size (Event)	128 K
Invoke response body payload size (RequestResponse)	6 MB

Sterling Talent Solutions

Limitation

AWS Lambda Account Limits Per Region

Resource	Default Limit
Concurrent executions	1000

Limitation

AWS Lambda Deployment Limits

tem	Default Limit
Lambda function deployment package size (compressed .zip/.jar file)	50 MB
Total size of all the deployment packages that can be uploaded per region	75 GB
Size of code/dependencies that you can zip into a deployment package (uncompressed .zip/.jar size)	250 MB
Total size of environment variables set	4 KB

Pricing

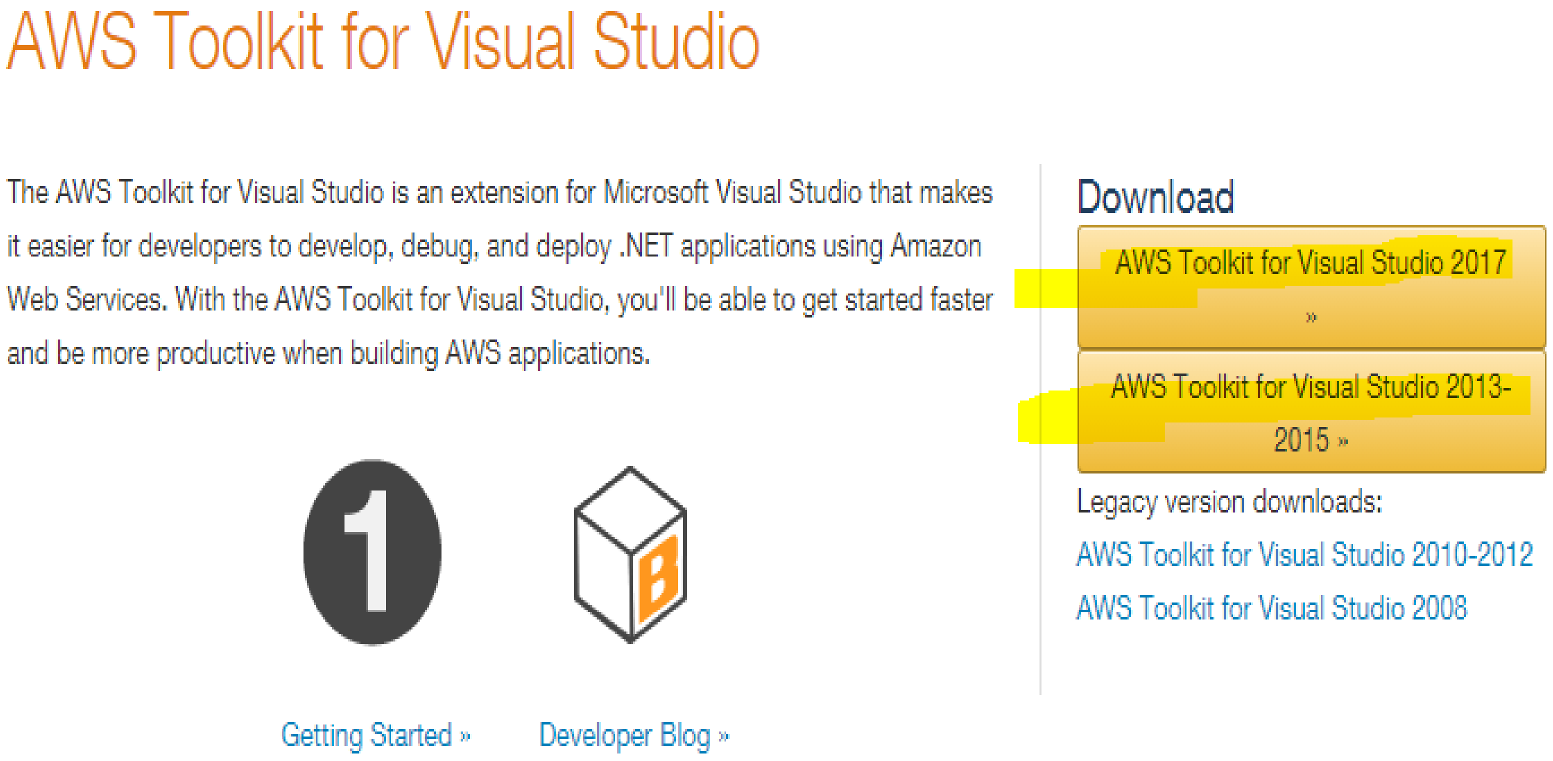
- First 1 million requests per month are free
- $0.20 per 1 million requests thereafter ($0.0000002 per request)

Memory (MB)	Free tier seconds per month	Price per 100ms ($)
128	3,200,000	0.000000208
192	2,133,333	0.000000313
256	1,600,000	0.000000417
320	1,280,000	0.000000521
384	1,066,667	0.000000625
448	914,286	0.000000729
512	800,000	0.000000834
576	711,111	0.000000938
640	640,000	0.000001042
704	581,818	0.000001146
768	533,333	0.000001250
832	492,308	0.000001354
896	457,143	0.000001459
960	426,667	0.000001563
1024	400,000	0.000001667
1088	376,471	0.000001771
1152	355,556	0.000001875
1216	336,842	0.000001980
1280	320,000	0.000002084
1344	304,762	0.000002188
1408	290,909	0.000002292
1472	278,261	0.000002396
1536	266,667	0.000002501

Sterling Talent Solutions

AWS Lambda with Visual Studio

- AWS Toolkit for Visual Studio
  - https://aws.amazon.com/visualstudio/



AWS Toolkit for Visual Studio

The AWS Toolkit for Visual Studio is an extension for Microsoft Visual Studio that makes it easier for developers to develop, debug, and deploy .NET applications using Amazon Web Services. With the AWS Toolkit for Visual Studio, you'll be able to get started faster and be more productive when building AWS applications.

Download

AWS Toolkit for Visual Studio 2017 »

AWS Toolkit for Visual Studio 2013-2015 »

Legacy version downloads:

AWS Toolkit for Visual Studio 2010-2012

AWS Toolkit for Visual Studio 2008

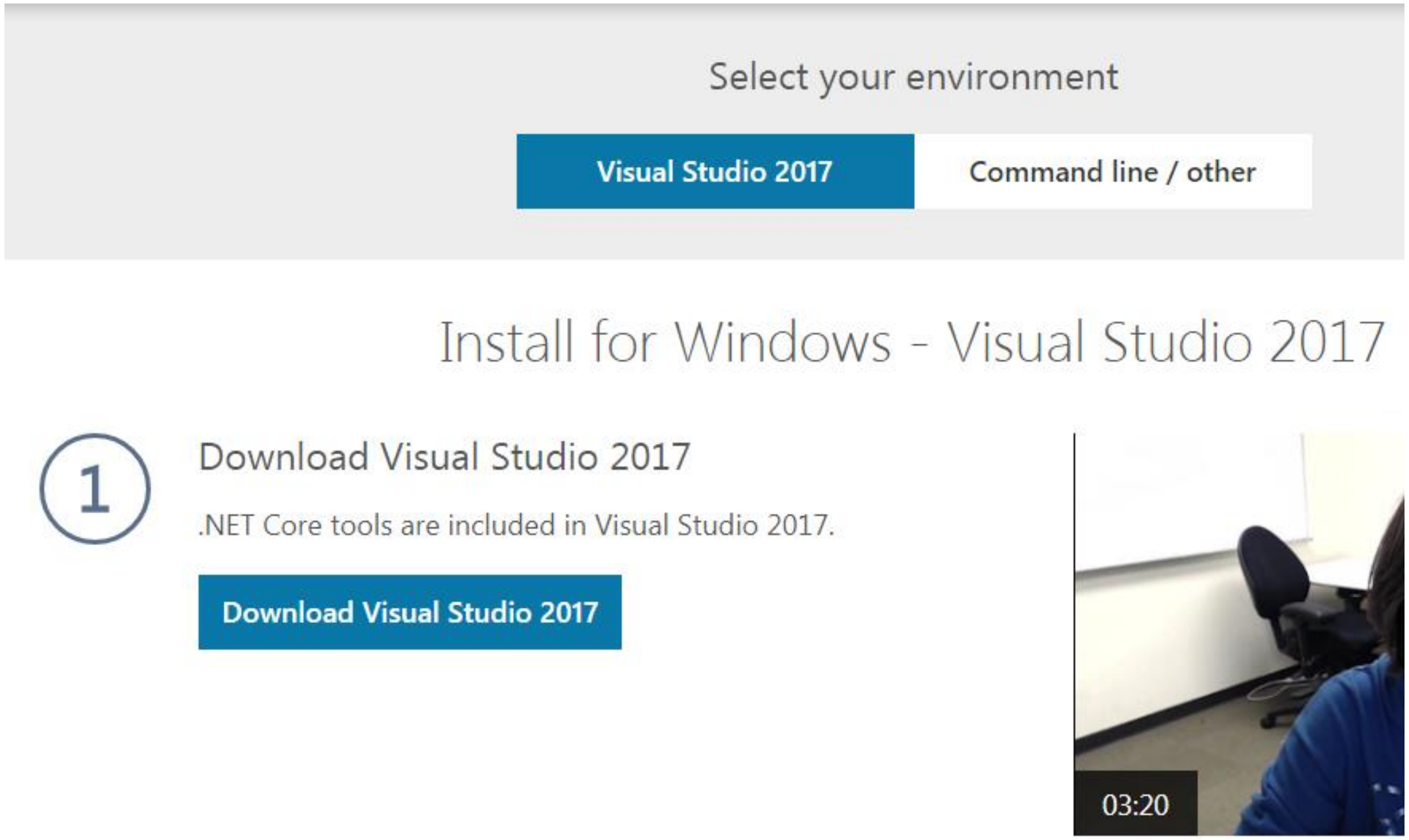
Getting Started »

Developer Blog »

Sterling Talent Solutions

AWS Lambda with Visual Studio

- AWS Lambda require minimum Core Framework 1.0
  - https://www.microsoft.com/net/core#windowscmd

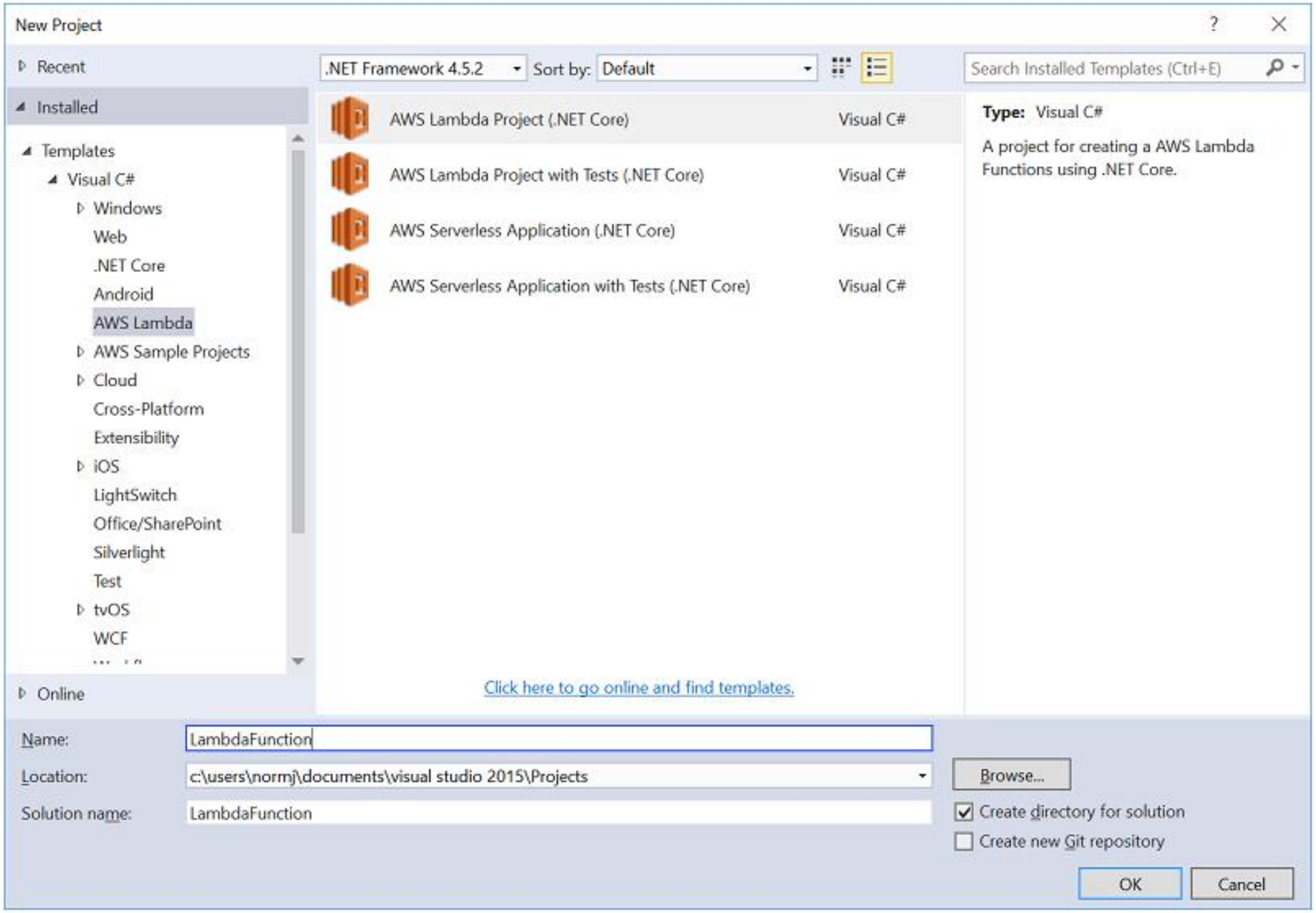


Install for Windows - Visual Studio 2017

1 Download Visual Studio 2017

.NET Core tools are included in Visual Studio 2017.

AWS Lambda with Visual Studio



New Project

.NET Framework 4.5.2 Sort by: Default

AWS Lambda Project (.NET Core)	Visual C#
AWS Lambda Project with Tests (.NET Core)	Visual C#
AWS Serverless Application (.NET Core)	Visual C#
AWS Serverless Application with Tests (.NET Core)	Visual C#

Type: Visual C#

A project for creating a AWS Lambda Functions using .NET Core.

Click here to go online and find templates.

Name: LambdaFunction

Location: c:\users\normj\documents\visual studio 2015\Projects

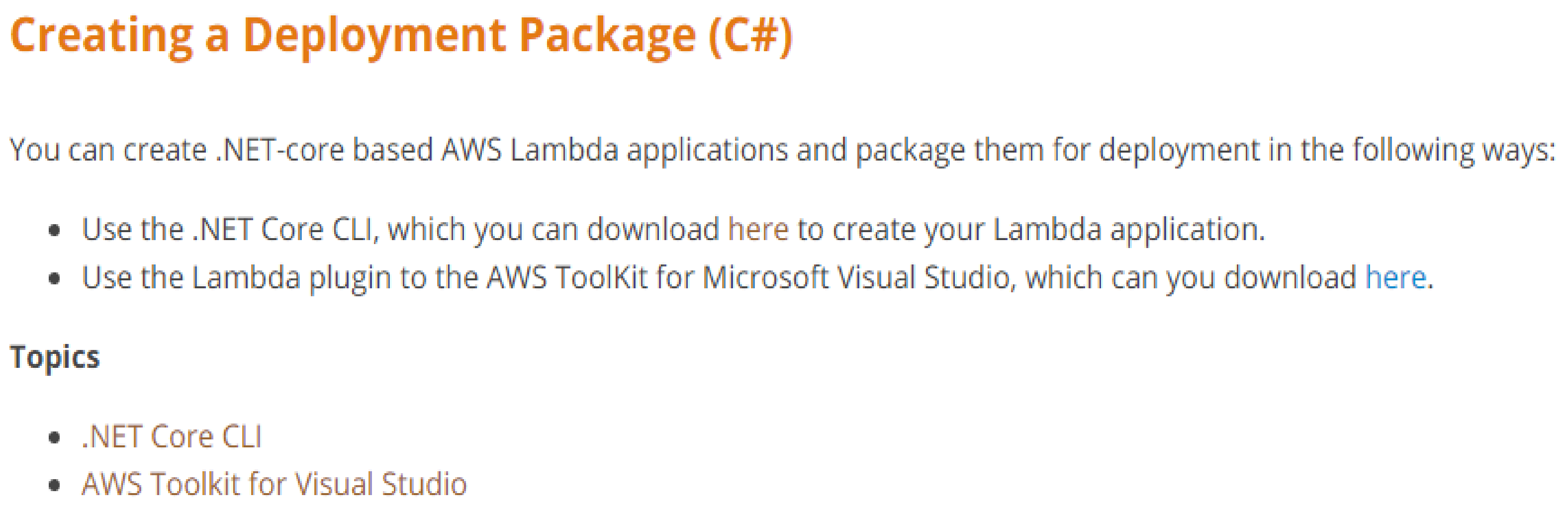
Solution name: LambdaFunction

☑ Create directory for solution

☐ Create new Git repository

Sterling Talent Solutions

AWS Lambda with Visual Studio



Creating a Deployment Package (C#)

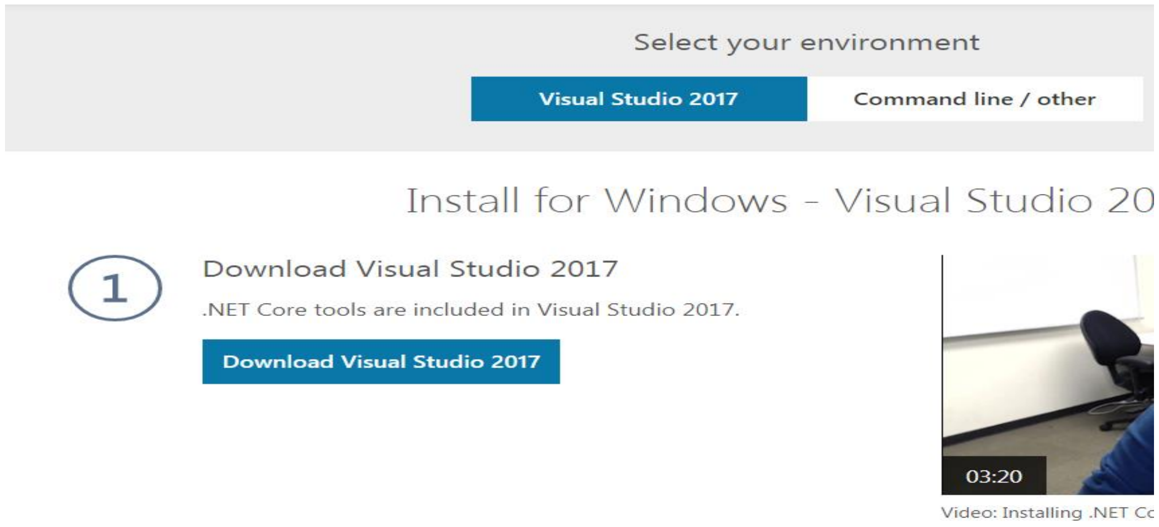
You can create .NET-core based AWS Lambda applications and package them for deployment in the following ways:

- Use the .NET Core CLI, which you can download here to create your Lambda application.
- Use the Lambda plugin to the AWS ToolKit for Microsoft Visual Studio, which can you download here.

Topics

- .NET Core CLI
- AWS Toolkit for Visual Studio

https://www.microsoft.com/net/core#windowsvs2017



Select your environment

Visual Studio 2017 | Command line / other

Install for Windows - Visual Studio 20

1 Download Visual Studio 2017

.NET Core tools are included in Visual Studio 2017.

Download Visual Studio 2017

03:20

Video: Installing .NET Cc

AWS Lambda with Visual Studio

- **Create the Deployment Package**
- To create the deployment package, open a command prompt and navigate to the folder that contains your project.json file and run the following commands:
- **dotnet restore** which will restore any references to dependencies of the project that may have changed during the development process.
- **dotnet publish** which compiles the application and packages the source code and any dependencies into a folder. The output of the command window will instruct you where the folder was created. For example:
- **Copy**
- publish: Published to C:\Users\*yourname**project-folder*\bin\debug\netcoreapp1.1\publishThe contents of this folder represent your application and at a minimum would look something like this:
- *application-name*.deps.json
- *application-name*.dll
- *application-name*.pdb
- *application-name*.runtimeconfig.json
- Zip the contents of the folder (not the folder itself). This is your deployment package.