[#hello-world-create] == Step 1: Create your CDK project

In this step, you create a new CDK project. A CDK project should be in its own directory, with its own local module dependencies.

*To create a CDK project*:: + . From a starting directory of your choice, create and navigate to a directory named hello-cdk: + [source,bash,subs=“verbatim,attributes”] — $ mkdir hello-cdk && cd hello-cdk — + IMPORTANT: Be sure to name your project directory hello-cdk, *exactly as shown here*. The CDK CLI uses this directory name to name things within your CDK code. If you use a different directory name, you will run into issues during this tutorial.

. From the hello-cdk directory, initialize a new CDK project using the CDK CLIcdk init command. Specify the app template and your preferred programming language with the --language option: + ==== [role=“tablist”] TypeScript:: + [source,bash,subs=“verbatim,attributes”] — $ cdk init app –language typescript —

JavaScript:: + [source,bash,subs=“verbatim,attributes”] — $ cdk init app –language javascript —

Python:: + [source,bash,subs=“verbatim,attributes”] — $ cdk init app –language python — + After the app has been created, also enter the following two commands. These activate the app’s Python virtual environment and installs the {aws} CDK core dependencies. + [source,bash,subs=“verbatim,attributes”] — $ source .venv/bin/activate # On Windows, run .\venv\Scripts\activate instead $ python -m pip install -r requirements.txt —

Java:: + [source,bash,subs=“verbatim,attributes”] — $ cdk init app –language java — + If you are using an IDE, you can now open or import the project. In [.noloc]Eclipse, for example, choose *File* > *Import* > *Maven* > *Existing Maven Projects*. Make sure that the project settings are set to use Java 8 (1.8).

C#:: + [source,bash,subs=“verbatim,attributes”] — $ cdk init app –language csharp — + If you are using Visual Studio, open the solution file in the src directory.

Go:: + [source,bash,subs=“verbatim,attributes”] — $ cdk init app –language go — + After the app has been created, also enter the following command to install the {aws} Construct Library modules that the app requires. + [source,bash,subs=“verbatim,attributes”] — $ go get — ====

The cdk init command creates a structure of files and folders within the hello-cdk directory to help organize the source code for your CDK app. This structure of files and folders is called your CDK *project*. Take a moment to explore your CDK project.

If you have [.noloc]Git installed, each project you create using cdk init is also initialized as a [.noloc]Git repository.

During project initialization, the CDK CLI creates a CDK app containing a single CDK stack. The CDK app instance is created using the link:https://docs.aws.amazon.com/cdk/api/v2/docs/aws-cdk-lib.App.html[App] construct. The following is a portion of this code from your CDK application file:

==== [role=“tablist”] TypeScript:: Located in bin/hello-cdk.ts: + [source,javascript,subs=“verbatim,attributes”] — #!/usr/bin/env node import ‘source-map-support/register’; import \* as cdk from ‘aws-cdk-lib’; import { HelloCdkStack } from ‘../lib/hello-cdk-stack’;

const app = new cdk.App(); new HelloCdkStack(app, ‘HelloCdkStack’, { }); —

JavaScript:: Located in bin/hello-cdk.js: + [source,javascript,subs=“verbatim,attributes”] — #!/usr/bin/env node

const cdk = require(‘aws-cdk-lib’); const { HelloCdkStack } = require(‘../lib/hello-cdk-stack’);

const app = new cdk.App(); new HelloCdkStack(app, ‘HelloCdkStack’, { }); —

Python:: Located in app.py: + [source,python,subs=“verbatim,attributes”] — #!/usr/bin/env python3 import os

import aws\_cdk as cdk

from hello\_cdk.hello\_cdk\_stack import HelloCdkStack

app = cdk.App() HelloCdkStack(app, “HelloCdkStack”,)

== app.synth()

Java:: Located in +src/main/java/.../HelloCdkApp.java+: + [source,java,subs=“verbatim,attributes”] — package com.myorg;

import software.amazon.awscdk.App; import software.amazon.awscdk.Environment; import software.amazon.awscdk.StackProps;

import java.util.Arrays;

public class HelloCdkApp { public static void main(final String[] args) { App app = new App();

…. new HelloCdkStack(app, “HelloCdkStack”, StackProps.builder() .build());

app.synth(); } } —- ….

C#:: Located in src/HelloCdk/Program.cs: + [source,csharp,subs=“verbatim,attributes”] — using Amazon.CDK; using System; using System.Collections.Generic; using System.Linq;

namespace HelloCdk { sealed class Program { public static void Main(string[] args) { var app = new App(); new HelloCdkStack(app, “HelloCdkStack”, new StackProps {}); app.Synth(); } } } —

Go:: Located in hello-cdk.go: + [source,go,subs=“verbatim,attributes”] — package main

import ( “github.com/aws/aws-cdk-go/awscdk/v2” “github.com/aws/constructs-go/constructs/v10” “github.com/aws/jsii-runtime-go” )

// …

func main() { defer jsii.Close()

app := awscdk.NewApp(nil)

NewHelloCdkStack(app, “HelloCdkStack”, &HelloCdkStackProps{ awscdk.StackProps{ Env: env(), }, })

app.Synth(nil) }

== // …

====

The CDK stack is created using the link:https://docs.aws.amazon.com/cdk/api/v2/docs/aws-cdk-lib.Stack.html[Stack] construct. The following is a portion of this code from your CDK stack file:

==== [role=“tablist”] TypeScript:: Located in lib/hello-cdk-stack.ts: + [source,javascript,subs=“verbatim,attributes”] — import \* as cdk from ‘aws-cdk-lib’; import { Construct } from ‘constructs’;

export class HelloCdkStack extends cdk.Stack { constructor(scope: Construct, id: string, props?: cdk.StackProps) { super(scope, id, props);

// Define your constructs here

} } —

JavaScript:: Located in lib/hello-cdk-stack.js: + [source,javascript,subs=“verbatim,attributes”] — const { Stack } = require(‘aws-cdk-lib’);

class HelloCdkStack extends Stack { constructor(scope, id, props) { super(scope, id, props);

// Define your constructs here

} }

== module.exports = { HelloCdkStack }

Python:: Located in hello\_cdk/hello\_cdk\_stack.py: + [source,python,subs=“verbatim,attributes”] — from aws\_cdk import ( Stack, ) from constructs import Construct

class HelloCdkStack(Stack):

def *init*(self, scope: Construct, construct\_id: str, **kwargs) -> None: super().*init*(scope, construct\_id,** kwargs)

# Define your constructs here —-

Java:: Located in +src/main/java/.../HelloCdkStack.java+: + [source,java,subs=“verbatim,attributes”] — package com.myorg;

import software.constructs.Construct; import software.amazon.awscdk.Stack; import software.amazon.awscdk.StackProps;

public class HelloCdkStack extends Stack { public HelloCdkStack(final Construct scope, final String id) { this(scope, id, null); }

public HelloCdkStack(final Construct scope, final String id, final StackProps props) { super(scope, id, props);

// Define your constructs here } } —-

C#:: Located in src/HelloCdk/HelloCdkStack.cs: + [source,csharp,subs=“verbatim,attributes”] — using Amazon.CDK; using Constructs;

namespace HelloCdk { public class HelloCdkStack : Stack { internal HelloCdkStack(Construct scope, string id, IStackProps props = null) : base(scope, id, props) { // Define your constructs here } } } —

Go:: Located in hello-cdk.go: + [source,go,subs=“verbatim,attributes”] — package main

import ( “github.com/aws/aws-cdk-go/awscdk/v2” “github.com/aws/constructs-go/constructs/v10” “github.com/aws/jsii-runtime-go” )

type HelloCdkStackProps struct { awscdk.StackProps }

func NewHelloCdkStack(scope constructs.Construct, id string, props \*HelloCdkStackProps) awscdk.Stack { var sprops awscdk.StackProps if props != nil { sprops = props.StackProps } stack := awscdk.NewStack(scope, &id, &sprops)

return stack }

== // …

====