include::attributes.txt[]

// Attributes

[.topic] [#toolkit-library-gs] = Getting started with the CDK Toolkit Library :info\_titleabbrev: Getting started :keywords: CDK Toolkit Library, Programmatic access, {aws} CDK, {aws} Cloud Development Kit ({aws} CDK), deploy, {aws} CloudFormation, Infrastructure as Code, synthesize

== [abstract]

## Get started with using the {aws} CDK Toolkit Library to programmatically perform CDK actions, such as synthesis and deployment, in your code.

// Content start

Get started with using the {aws} CDK Toolkit Library to programmatically perform CDK actions, such as synthesis and deployment, in your code.

[#toolkit-library-gs-prerequisites] == Prerequisites

. Supported version of Node.js installed. . {aws} credentials configured. . Basic familiarity with the {aws} CDK.

For more information, see xref:prerequisites[{aws} CDK prerequisites].

[#toolkit-library-gs-install] == Step 1: Installing the CDK Toolkit Library

Install the CDK Toolkit Library package in your project’s development environment by running the following:

== [source,none,subs=“verbatim,attributes”]

## npm install –save @aws-cdk/toolkit-lib

[#toolkit-library-gs-initialize] == Step 2: Initializing the CDK Toolkit Library

Create a CDK Toolkit instance to perform programmatic actions on your CDK app.

== [source,typescript,subs=“verbatim,attributes”]

import { Toolkit } from ‘@aws-cdk/toolkit-lib’;

const toolkit = new Toolkit({ // Optional configuration options go here }); —

You can customize the CDK Toolkit instance during creation. For instructions, see xref:toolkit-library-configure[Configure your CDK Toolkit instance].

[#toolkit-library-gs-ca] == Step 3: Creating a cloud assembly source for your CDK app

A cloud assembly source provides instructions for generating CloudFormation templates from your CDK app. You can create one in multiple ways. The following are a few examples:

| . *An inline assembly builder function*: + [source,typescript,subs=“verbatim,attributes”] |
| --- |
| import \* as cdk from ‘aws-cdk-lib’; |
| const cloudAssemblySource = await toolkit.fromAssemblyBuilder(async () => { const app = new cdk.App(); new MyStack(app, ‘MyStack’); return app.synth(); }); |

. *An existing CDK app file*: + [source,typescript,subs=“verbatim,attributes”] — const cloudAssemblySource = await toolkit.fromCdkApp(“ts-node app.ts”); — –

For more information, see xref:toolkit-library-configure-ca[Configure cloud assembly sources].

[#toolkit-library-gs-define] == Step 4: Defining programmatic actions for your CDK app

Now that you’ve created a CDK Toolkit instance and cloud assembly source, you can start to define programmatic actions. The following is a basic example that creates a deployment of the MyStack stack:

== [source,typescript,subs=“verbatim,attributes”]

import { StackSelectionStrategy } from ‘@aws-cdk/toolkit-lib’;

await toolkit.deploy(cloudAssemblySource, { stacks: { strategy: StackSelectionStrategy.PATTERN\_MUST\_MATCH, // Deploy only stacks that exactly match the provided patterns patterns: [“MyStack”], }, }); —

[#toolkit-library-gs-customize] == Step 5: Customizing the CDK Toolkit further

You can configure and customize the CDK Toolkit further for your needs:

* *Messages and interactions* - Configure how the CDK Toolkit communicates with users and applications. See xref:toolkit-library-configure-messages[Configure messages & interactions].
* *Error handling* - Implement structured error handling for CDK operations. See xref:toolkit-library-configure-errors[Configure error handling].

[#toolkit-library-gs-resources] == Additional resources

For more information on the CDK Toolkit Library npm package, see the link:https://www.npmjs.com/package/@aws-cdk/toolkit-lib [ReadMe] in the \_@aws-cdk/toolkit-lib\_ npm package.

For API reference information, see the link:https://docs.aws.amazon.com/cdk/api/toolkit-lib/[CDK Toolkit Library API reference].