



# Continuous Integration with CodeBuild



Joseph Cole-Showers

```
! index.jsp      ! settings.xml U      ! buildspec.yml U X
1  ! buildspec.yml
2
3  version: 0.2
4
5  phases:
6    install:
7      runtime-versions:
8        java: corretto8
9    pre_build:
10      commands:
11        - echo Initializing environment
12        - export CODEARTIFACT_AUTH_TOKEN=`aws codeartifact get-authorization-token --region us-east-1 --repository nextwork-web-project --profile default | jq -r .token`
13    build:
14      commands:
15        - echo Build started on `date`
16        - mvn -s settings.xml compile
17    post_build:
18      commands:
19        - echo Build completed on `date`
20        - mvn -s settings.xml package
21  artifacts:
22    files:
23      - target/nextwork-web-project.war
discard-paths: no
```



# Introducing Today's Project!

In this project, I will demonstrate how to use AWS CodeBuild to automate the build process in the CI/CD pipeline. i.e Compressing web app files into a single compress file like a zip file that I can later deploy.

## Key tools and concepts

Services I used include: CodeBuild, CodeConnection, CodeArtifact, S3, EC2, Github and VS Code. Key concepts I learnt include the build process, buildspec.yml, using Codeconnections to connect AWS with Github.

## Project reflection

This project took me approximately 2hrs to do including troubleshooting time. The most challenging part was resolving connections to Github from AWS.

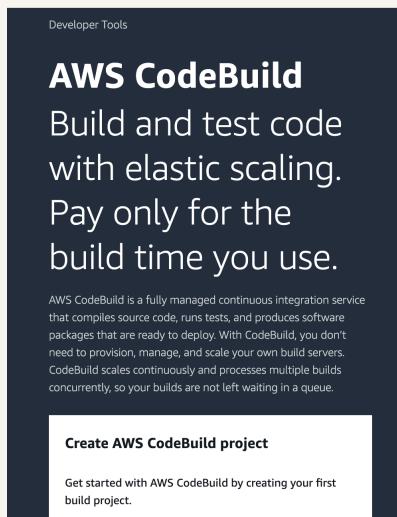
This project is part four of a series of DevOps projects where I'm building a CI/CD pipeline.



# Setting up a CodeBuild Project

CodeBuild is a continuous integration service, which means it helps to compile and package code. It turns the web app code into something computers can run. Engineering teams use it because CI services automates this process.

My CodeBuild project's source configuration means where the code lives and I selected GitHub. That is where the web app source code repo is stored.



The screenshot shows the AWS CodeBuild landing page. At the top, it says "Developer Tools" and "AWS CodeBuild". Below that, it features the tagline "Build and test code with elastic scaling. Pay only for the build time you use." A detailed description follows: "AWS CodeBuild is a fully managed continuous integration service that compiles source code, runs tests, and produces software packages that are ready to deploy. With CodeBuild, you don't need to provision, manage, and scale your own build servers. CodeBuild scales continuously and processes multiple builds concurrently, so your builds are not left waiting in a queue." At the bottom, there is a call-to-action button labeled "Create AWS CodeBuild project" and a smaller text box that says "Get started with AWS CodeBuild by creating your first build project." A horizontal orange bar is located just below the "Create AWS CodeBuild project" button.



# Connecting CodeBuild with GitHub

There are multiple credential types for GitHub, like GitHub App, Personal Access Tokens and OAuth app. I used GitHub App because it is the most simple, yet the most secure way between GitHub and AWS. AWS handles all of the credentials e.g the tokens

The service that helped connect is AWS CodeConnections. CodeConnections helps to connect environment with other third-party platforms.

The screenshot shows the 'Source' configuration page for AWS CodeConnections. At the top, there is a dropdown menu labeled 'Source' and a button labeled 'Add source'. Below this, under 'Source 1 - Primary', the 'Source provider' is set to 'GitHub'. A success message indicates that the account is connected via an AWS managed GitHub App. There is an option to 'Manage account credentials'. Below this, there is a checkbox for 'Use override credentials for this project only'. Under 'Repository', two options are shown: 'Repository in my GitHub account' (selected) and 'Public repository'. The entire configuration is contained within a light gray rounded rectangle.



# CodeBuild Configurations

## Environment

My CodeBuild project's Environment configuration means the environment setup for the compute power i.e the EC2 instance that will be compiling and compressing the files when I run the CodeBuild project.

## Artifacts

Build artifacts are files that get created as part of the build process. They're important because they represent artifacts that will be used later in the CI/CD pipeline. My build process will create a compressed file for the webapp. Stored it in S3.

## Packaging

When setting up CodeBuild, I also chose to package artifacts into a zip file. This helps with package management as files are all organized neatly in a single executable file.



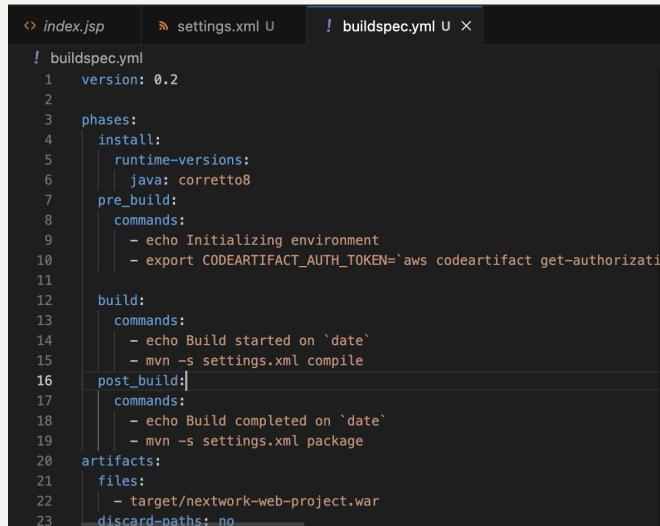
## Monitoring

For monitoring, I enabled CloudWatch Logs, which is a service that will log commands that run and any errors that happen. This is helpful for troubleshooting.

# buildspec.yml

My first build failed because CodeBuild could not find the buildspec.yml file in the source code root directory. A buildspec.yml file is needed because it tells CodeBuild how to run the build process.

The first 2 phases in my buildspec.yml file installs Java and gets access to CodeArtifact. The third phase in my buildspec.yml file compiles the web app. The fourth phase in my buildspec.yml file packages the web app into a single compressed artifact



```
! buildspec.yml
1   version: 0.2
2
3   phases:
4     install:
5       runtime-versions:
6         | java: corretto8
7     pre_build:
8       commands:
9         | - echo Initializing environment
10        | - export CODEARTIFACT_AUTH_TOKEN=`aws codeartifact get-authorization-token --profile default --region us-east-1 --domain nextwork --domain-owner 123456789012 --repository nextwork-web-project`
11
12     build:
13       commands:
14         | - echo Build started on `date`
15         | - mvn -s settings.xml compile
16     post_build:
17       commands:
18         | - echo Build completed on `date`
19         | - mvn -s settings.xml package
20   artifacts:
21     files:
22       | - target/nextwork-web-project.war
23   discard-paths: no
```

# Success!

My second build also failed, but with a different error that said I failed to run an error while executing the command 'mvn -s settings.xml' to compile the web app.

To resolve the second error, I attached a policy for access CodeArtifact to CodeBuild role. When I built my project again, I saw a success. With the privileges to CodeArtifact granted by the role, Codebuild now has access to the repo.

To verify the build, I checked S3 bucket for the artifact. Seeing the artifact tells me that Codebuild was a success. The build process compiled the code from the source repo, compressed it to a file and stored the file in S3 bucket.

## Build status

Status

 **Succeeded**



NextWork.org

# **Everyone should be in a job they love.**

Check out nextwork.org for  
more projects

