Building and testing Xamarin applications

In this article

Introduction
Prerequisites
Building Xamarin.iOS apps
Building Xamarin.Android apps
Specifying a .NET version

You can create a continuous integration (CI) workflow in GitHub Actions to build and test your Xamarin application.

Note: GitHub-hosted runners are not currently supported on GitHub Enterprise Server. You can see more information about planned future support on the GitHub public roadmap.

Introduction

This guide shows you how to create a workflow that performs continuous integration (CI) for your Xamarin project. The workflow you create will allow you to see when commits to a pull request cause build or test failures against your default branch; this approach can help ensure that your code is always healthy.

For a full list of available Xamarin SDK versions on the GitHub Actions-hosted macOS runners, see the README file for the version of macOS you want to use in the GitHub Actions Runner Images repository.

Prerequisites

We recommend that you have a basic understanding of Xamarin, .NET Core SDK, YAML, workflow configuration options, and how to create a workflow file. For more information, see:

- "Workflow syntax for GitHub Actions"
- "Getting started with .NET"
- "Learn Xamarin"

Building Xamarin.iOS apps

The example below demonstrates how to change the default Xamarin SDK versions and build a Xamarin.iOS application.

```yaml
name: Build Xamarin.iOS app

on: [push]
```

```yaml
jobs:
  build:

    runs-on: macos-latest

    steps:
    - uses: actions/checkout@v4
    - name: Set default Xamarin SDK versions
      run: |
        $VM_ASSETS/select-xamarin-sdk-v2.sh --mono=6.12 --ios=14.10

    - name: Set default Xcode 12.3
      run: |
        XCODE_ROOT=/Applications/Xcode_12.3.0.app
        echo "MD_APPLE_SDK_ROOT=$XCODE_ROOT" >> $GITHUB_ENV
        sudo xcode-select -s $XCODE_ROOT

    - name: Setup .NET Core SDK 5.0.x
      uses: actions/setup-dotnet@v3
      with:
        dotnet-version: '5.0.x'

    - name: Install dependencies
      run: nuget restore <sln_file_path>

    - name: Build
      run: msbuild <csproj_file_path> /p:Configuration=Debug
/p:Platform=iPhoneSimulator /t:Rebuild
```

Building Xamarin.Android apps

The example below demonstrates how to change default Xamarin SDK versions and build a Xamarin.Android application.

```yaml
name: Build Xamarin.Android app

on: [push]

jobs:
  build:

    runs-on: macos-latest

    steps:
    - uses: actions/checkout@v4
    - name: Set default Xamarin SDK versions
      run: |
        $VM_ASSETS/select-xamarin-sdk-v2.sh --mono=6.10 --android=10.2

    - name: Setup .NET Core SDK 5.0.x
      uses: actions/setup-dotnet@v3
      with:
        dotnet-version: '5.0.x'

    - name: Install dependencies
      run: nuget restore <sln_file_path>

    - name: Build
      run: msbuild <csproj_file_path> /t:PackageForAndroid /p:Configuration=Debug
```

Specifying a .NET version

To use a preinstalled version of the .NET Core SDK on a GitHub-hosted runner, use the `setup-dotnet` action. This action finds a specific version of .NET from the tools cache on

each runner, and adds the necessary binaries to `PATH`. These changes will persist for the remainder of the job.

The `setup-dotnet` action is the recommended way of using .NET with GitHub Actions, because it ensures consistent behavior across different runners and different versions of .NET. If you are using a self-hosted runner, you must install .NET and add it to `PATH`. For more information, see the `setup-dotnet` action.