



ADC²⁴
Bristol

ROCK-SOLID RELEASES

*BEGINNER-FRIENDLY INTRODUCTION TO CI/CD
FOR AUDIO DEVELOPERS*

FERNANDO GARCIA DE LA CRUZ

Rock-Solid Releases

CI/CD for Audio Developers

Fernando Garcia de la Cruz



Universitat
Pompeu Fabra
Barcelona



Ear Candy
Technologies

MTG

Music Technology
Group

Audio Developer Conference 2024 - Bristol, UK
November 2024



A little about me

- Software & Data Engineer from México 🇲🇽 🌮
- Guitar Player & Technology Enthusiast
- Audio Developer (and more) at [Ear Candy Technologies](#)
- Master's Student in Sound & Music Computing at Universitat Pompeu Fabra, Barcelona

[Website](#) | [LinkedIn](#) | [GitHub](#)



Fernando Garcia
de la Cruz

**Rock-Solid
Releases:**
CI/CD for Audio
Developers

audio.dev

**ADC²⁴
Bristol**

AUDIO DEVELOPER
CONFERENCE
NOVEMBER 11-13
BRISTOL UK & ONLINE

Agenda

- CI/CD for audio plugins
- Github Actions (Tools for CI/CD)
- Requirements
- Practical examples
- Tips and tricks
- Resources

Before we start

- **Beginner-Friendly** Focus, good place to start!
- **Not a Deep Dive:** Sharing insights from experience, not expert knowledge
- **Basic Plugin Example:** JUCE plugin, and GitHub Actions example pipeline
- **Out of Scope:** AAX, OS-specific details, signing, notarization.
- **Examples on Mac:** Concepts are adaptable across platforms



Code

**adc24-rsr** Star
Created by fergarciadlc
Repository for "Rock-Solid Releases" ADC 2024 talk, beginner-friendly CI/CD workflows, CMake builds, and setups for audio plugin development.

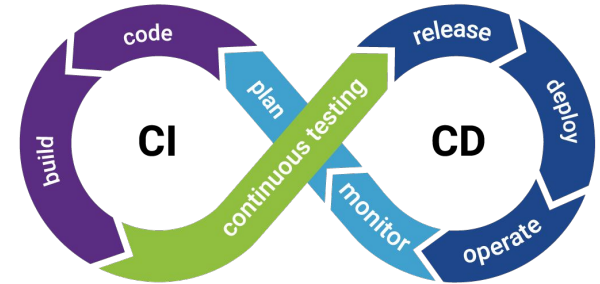
0 FORKS **0** STARS

<https://github.com/fergarciadlc/adc24-rsr>

CI/CD in a Nutshell

Basic Intuition

- **CI (Continuous Integration)**: Merges and tests code frequently to catch errors early.
- **CD (Continuous Delivery)**: Automates the deployment process to release reliable updates smoothly.



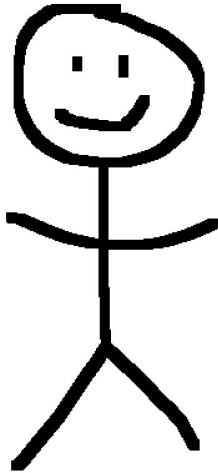
CI/CD for Audio Development?

Consistency Across Platforms

- Ensures code is always deployable.
- Improved **Code Quality** (validation and testing)
- **catch bugs early!**
- Reduced Manual Effort
- Automate deploy process

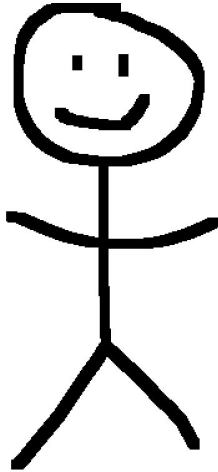


Meet Juan



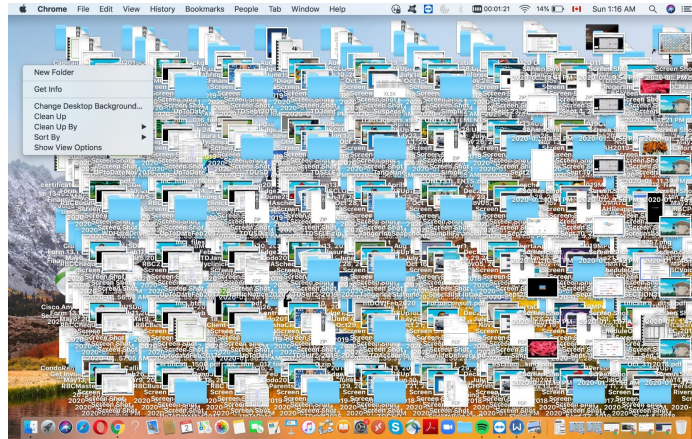
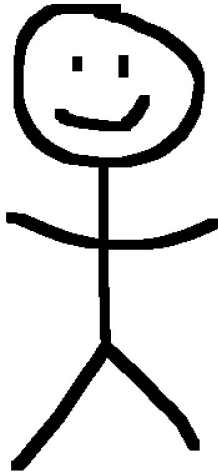
Meet Juan

CEO: Super cool tensor neural gpt binaural plugin company TM



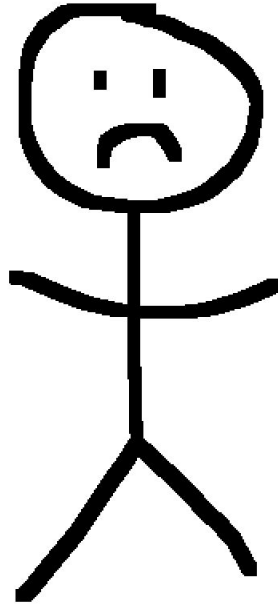
The Problem

Juan's Desktop



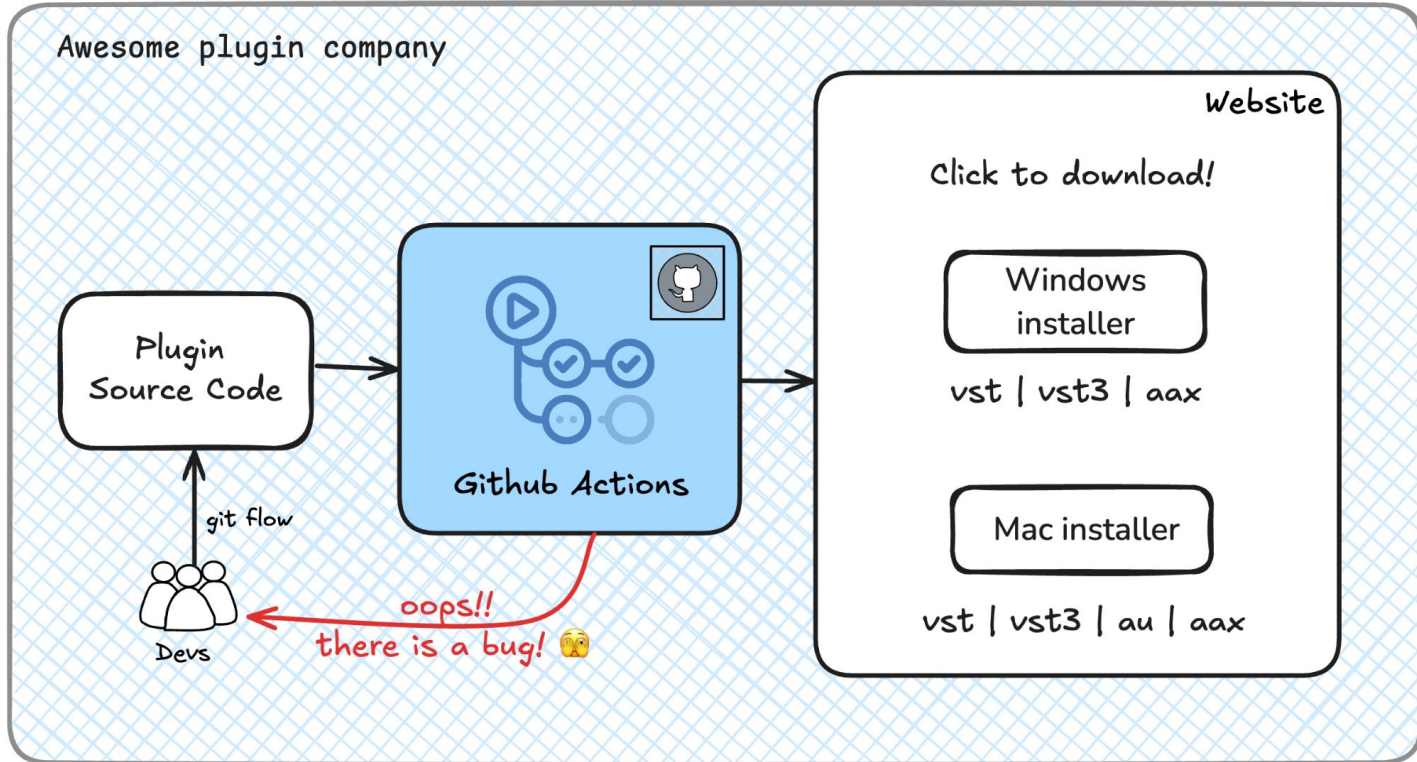
Wrong version of the plugin

Juan is sad

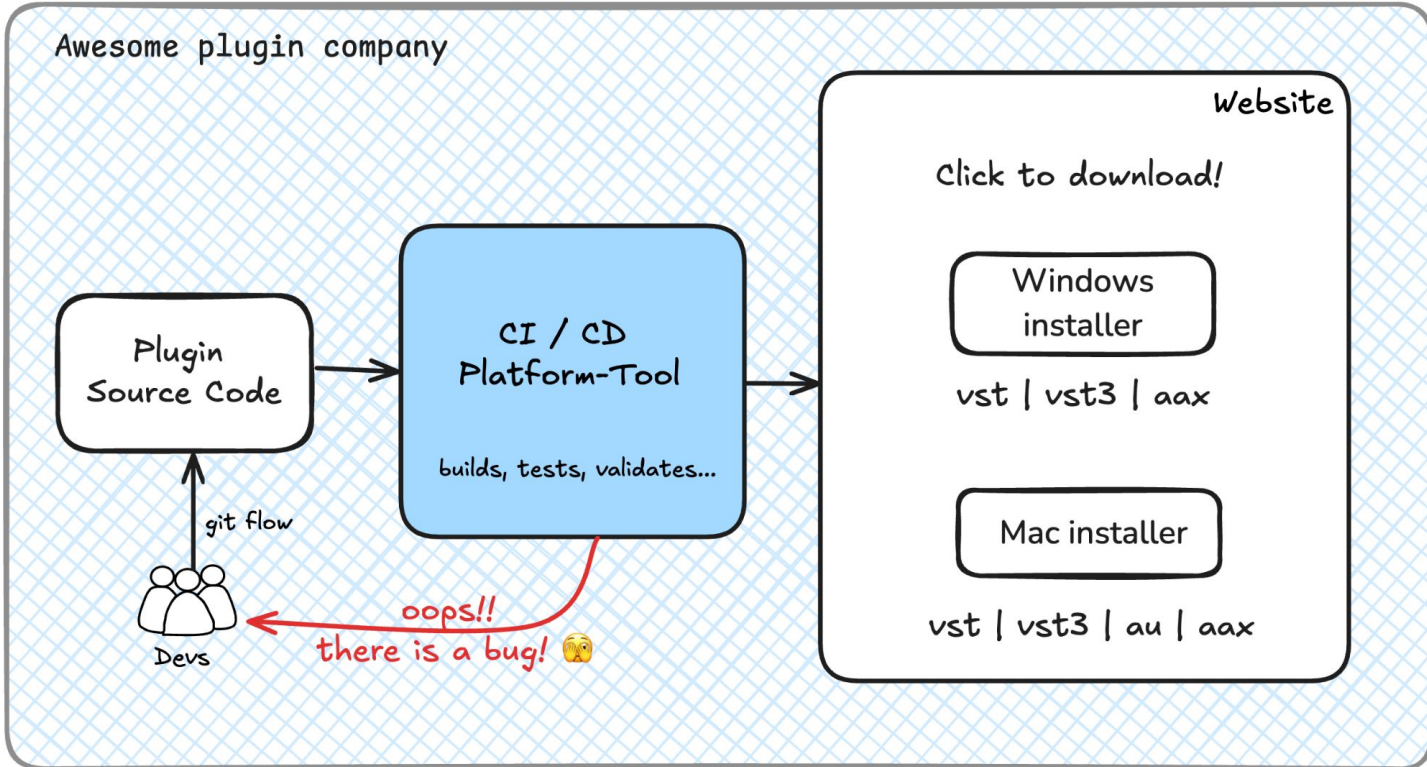


... and broke

The Solution - CI/CD Intuition for a plugin company



How does it work? - CI/CD Intuition for a plugin company



Why GitHub Actions?

Integrated with GitHub

Cross-Platform

Community support

Reusable Workflows -> Extensive Marketplace

Real-Time Feedback



GitHub Actions

Why GitHub Actions?

Free!!



Why GitHub Actions?

Ok, not exactly...

Awesome **free tier**

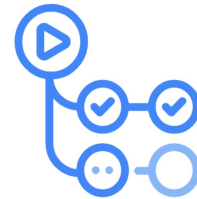
¯_ (ツ) _ /



Plan	Storage	Minutes (per month)
GitHub Free	500 MB	2,000
GitHub Pro	1 GB	3,000
GitHub Free for organizations	500 MB	2,000
GitHub Team	2 GB	3,000
GitHub Enterprise Cloud	50 GB	50,000

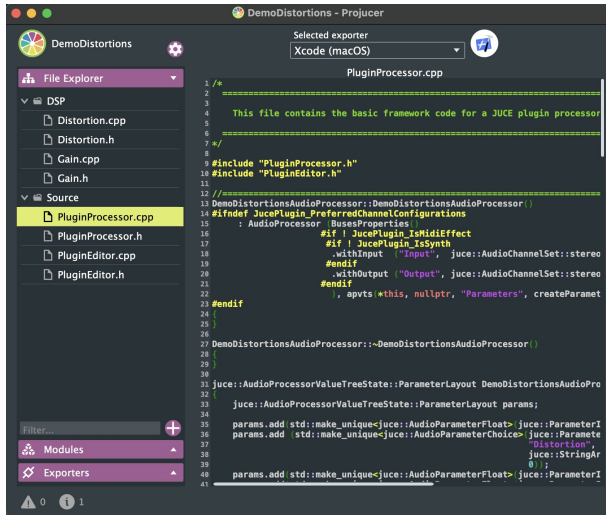
Check: [About billing for GitHub Actions](#)

Ok, What do we need



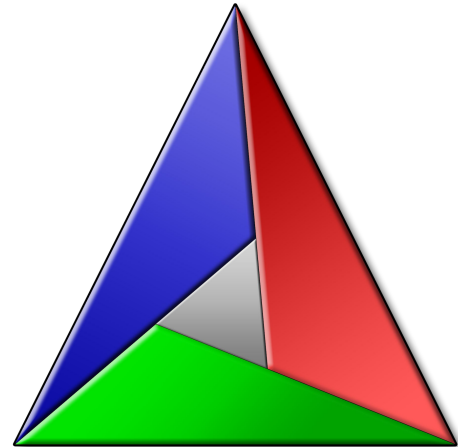
GitHub Actions

Projucer → CMake



The screenshot shows the Projucer IDE interface. The left sidebar displays a file explorer with a tree view containing folders for DSP, Source, and Modules, and files for Distortion.cpp, Distortion.h, Gain.cpp, Gain.h, PluginProcessor.cpp, PluginProcessor.h, PluginEditor.cpp, and PluginEditor.h. The main editor window shows the code for PluginProcessor.cpp, which includes headers and implements the JUCE plugin processor interface. The code includes comments and preprocessor directives for handling different audio channel sets and parameters.

```
1 //
2
3
4 This file contains the basic framework code for a JUCE plugin processor
5
6
7 //
8
9 #include "PluginProcessor.h"
10 #include "PluginEditor.h"
11
12 //
13 DemoDistortionsAudioProcessor::DemoDistortionsAudioProcessor()
14 #ifndef JUCE_PLUGIN_PREFERRED_CHANNEL_CONFIGURATIONS
15 : AudioProcessor(BusesProperties
16 : AudioProcessor( #if ! JUCE_PLUGIN_IS_MIDI_EFFECT
17 : #if ! JUCE_PLUGIN_IS_SYNTH
18 : #endif
19 : #endif
20 : #endif
21 : #endif
22 : #endif
23 : #endif
24
25
26
27 DemoDistortionsAudioProcessor::~DemoDistortionsAudioProcessor()
28
29
30
31 juce::AudioProcessorValueTreeState::ParameterLayout DemoDistortionsAudioPro
32
33 juce::AudioProcessorValueTreeState::ParameterLayout params;
34
35 params.add std::make_unique<juce::AudioParameterFloat> juce::ParameterI
36 params.add std::make_unique<juce::AudioParameterChoice> juce::Paramete
37
38
39
40 params.add std::make_unique<juce::AudioParameterFloat> juce::ParameterI
41
```



CMake in a nutshell

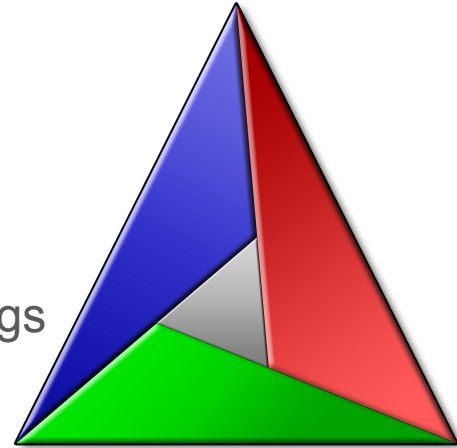
CMake is:

- A build management tool
- Used for compiling software projects

How does it work?

- Describes project files, dependencies, and settings
- Generates IDE's projects files

JUCE Plugin: Work from terminal



Great CMake resources for plugin dev

Templates: [Pamplejuce](#) (Sudara), [Plugin Template](#) (Jan Wilczek - WolfSound)

Sudara Blog: [How to use CMake with JUCE](#)

Open Source projects: [OJD](#), [BYOD](#) (or any from chowdsp)

TAP: [CMake for JUCE Developers \(#1\): Why CMake?](#)

ECT Meeting (in spanish): [Un paseo por la CMake API de JUCE](#)

ADC21: [CI/CD for Audio Plugin Development - Jatin Chowdhury - ADC21](#)

Github Actions: Workflows basics

What is a Workflow?

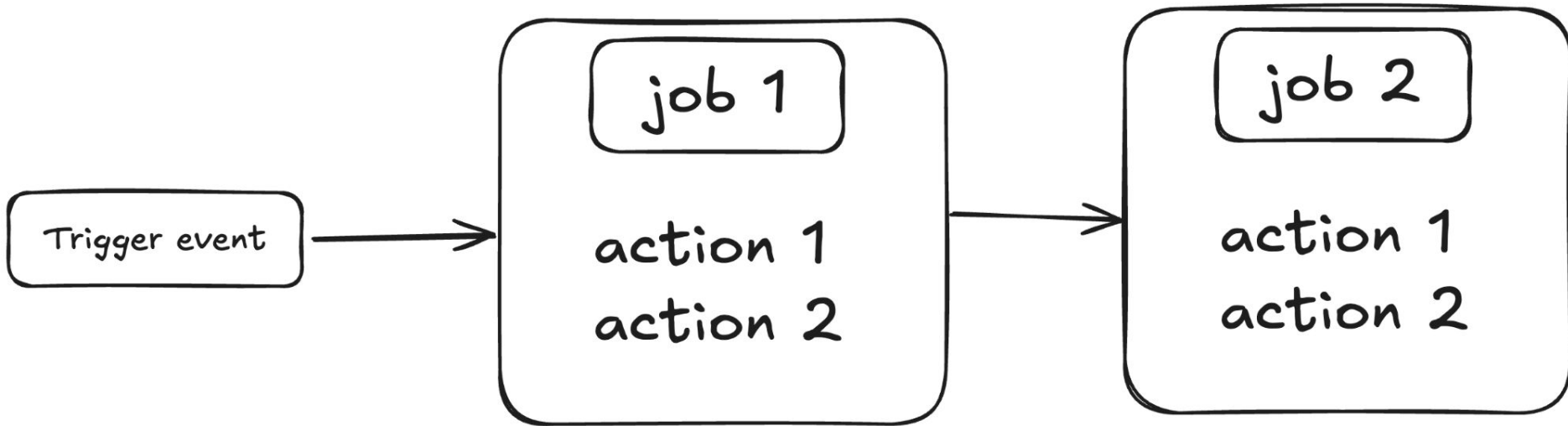
- Automated processes that run one or more tasks
- Configured using a YAML file in your repository

Examples of use cases:

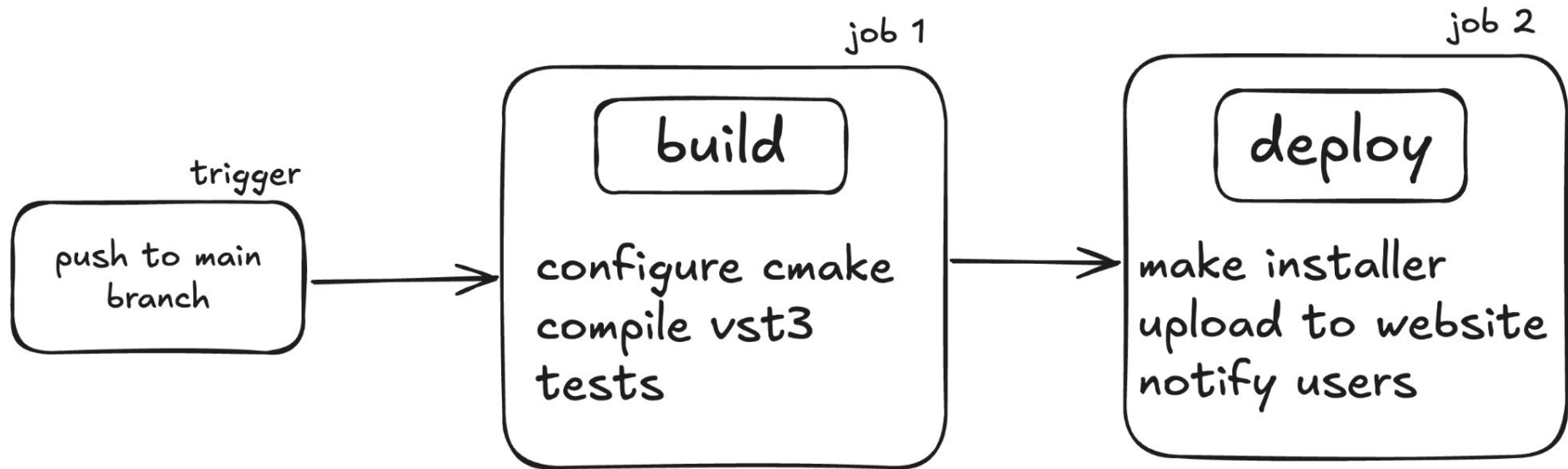
- Building and testing pull requests.
- Deploying your application every time a release is created.
- Adding a label whenever a new issue is opened.

[docs](#)

The Pipeline



The Pipeline



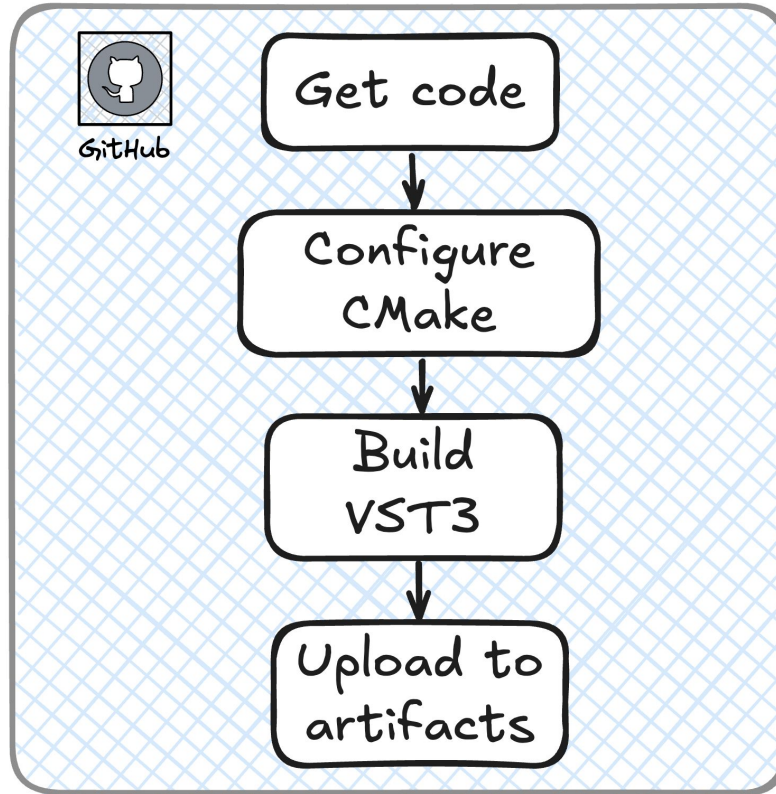
YAML

(YAML Ain't Markup Language)

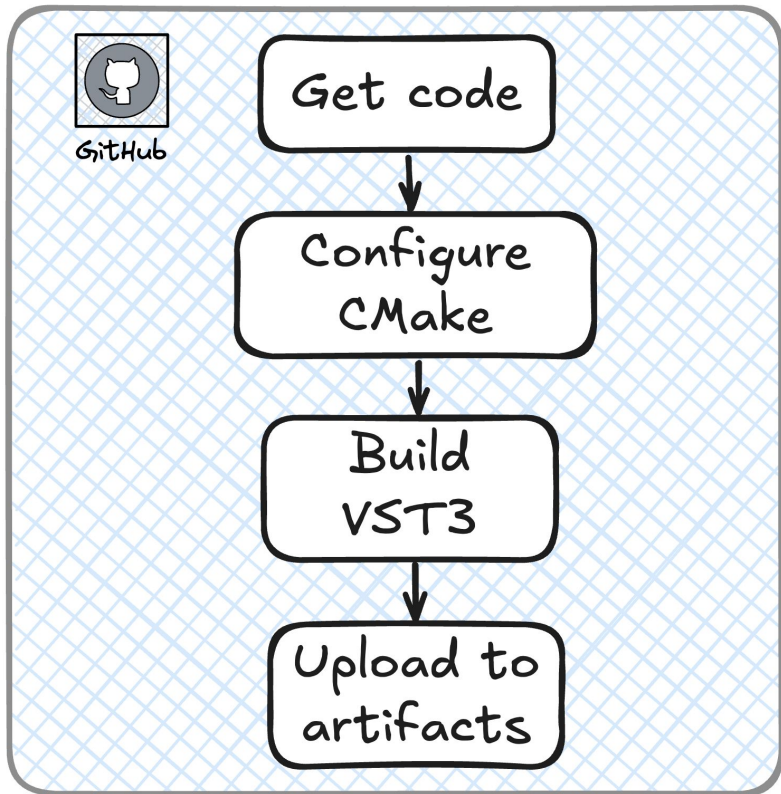
Human-readable data serialization
format commonly used for
configuration files.

```
1  audio_programmer:
2    name: "ReverbLord69420 🕶️"
3    age: "Forever young"
4    skill_set:
5      - "XCode Warning Generator 🕶️"
6      - "ChatGPT"
7      - "sin(x) = x"
8    common_sayings:
9      - "Why is there a delay? 😞"
10     - "It works on my machine 🙄"
11   toolkit:
12     DAW: "Audacity of course"
13     plugins:
14       - "Reverb MAX 3000 🎚️"
15       - "Mariachinator 🌮"
16   memes_referenced_per_day: 47
```

Test case 1: Build a JUCE VST3 plugin

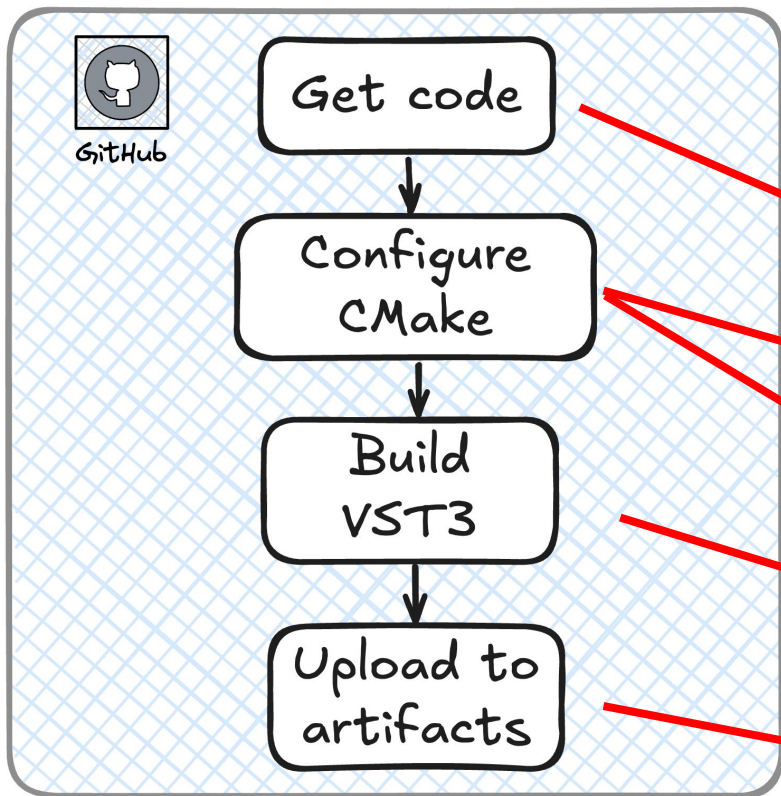


Test case 1: Build a JUCE VST3 plugin



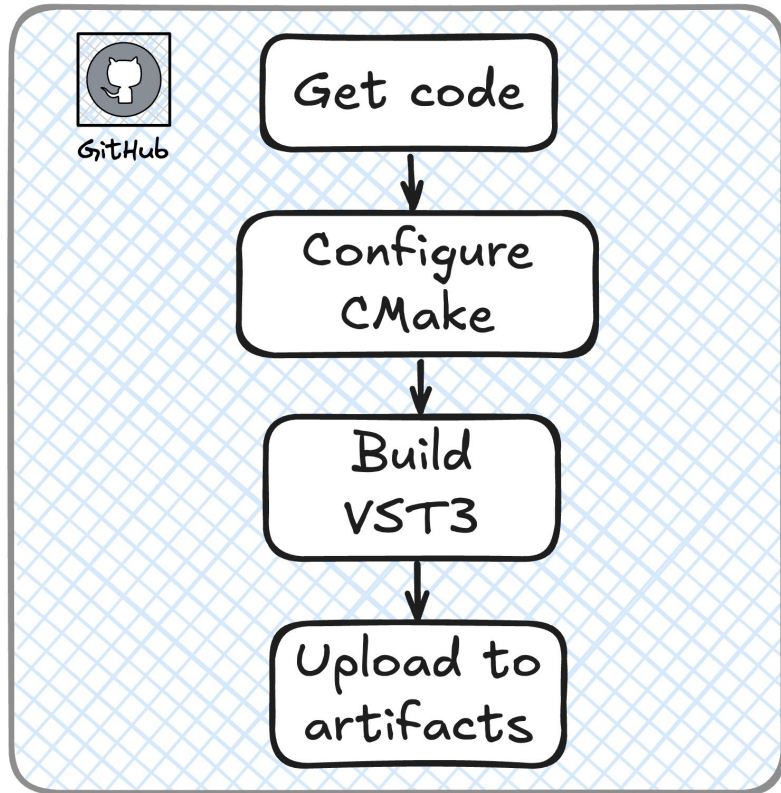
```
1 name: Mac Pipeline
2
3 > on: ...
7 > env: ...
10 ∨ jobs:
11 ∨ build:
12 | name: Build on macos-14
13 | runs-on: macos-14
14 ∨ | steps:
15 > | - name: Checkout code...
19 |
20 | # Ensure the latest version of CMake is installed
21 > | - name: Get latest CMake...
23 |
24 | # Configure the project using CMake
25 ∨ | - name: Configure CMake
26 | | run: cmake -Bbuild
27 |
28 | # Build the plugin in Release mode
29 ∨ | - name: Build Plug-In
30 | | run: sudo cmake --build build --config Release
31 |
32 | # Upload the built plugin in github actions artifact
33 > | - name: Upload Builds to Artifact...
```

Test case 1: Build a JUCE VST3 plugin



```
1 name: Mac Pipeline
2
3 > on: ...
7 > env: ...
10 ∨ jobs:
11 ∨ build:
12   name: Build on macos-14
13   runs-on: macos-14
14   ∨ steps:
15     - name: Checkout code...
19
20   # Ensure the latest version of CMake is installed
21   > - name: Get latest CMake...
23
24   # Configure the project using CMake
25   ∨ - name: Configure CMake
26     run: cmake -Bbuild
27
28   # Build the plugin in Release mode
29   ∨ - name: Build Plug-In
30     run: sudo cmake --build build --config Release
31
32   # Upload the built plugin in github actions artifact
33   > - name: Upload Builds to Artifact...
```

Test case 1: Build a JUCE VST3 plugin



← Mac Pipeline

✓ basic gha build #1

✓ Build on macos-14 ▾

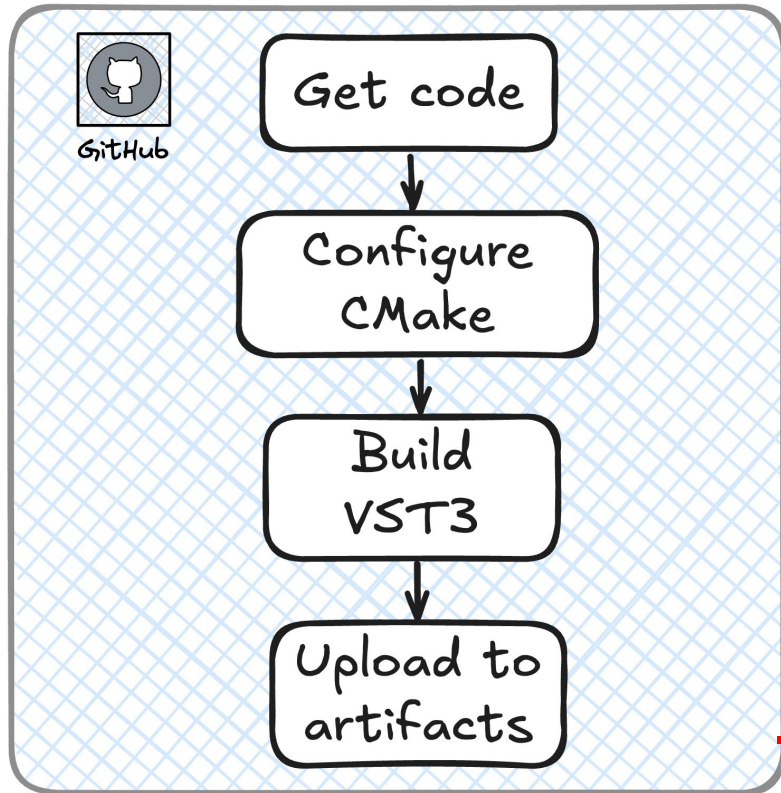
Build on macos-14

succeeded 20 minutes ago in 2m 0s

- > ✓ Set up job
- > ✓ Checkout code
- > ✓ Get latest CMake
- > ✓ Configure CMake
- ▾ ✓ Build Plug-In

```
1 ▶ Run sudo cmake --build build --config Release
7 [ 2%] Generating ADC24RSR_artefacts/JuceLibraryCode/JuceLibraryCode.o
8 [ 4%] Building CXX object CMakeFiles/ADC24RSR.dir/SourceCode/ADC24RSR.cpp.o
9 [ 6%] Building CXX object CMakeFiles/ADC24RSR.dir/SourceCode/ADC24RSR.cpp.o
10 [ 8%] Building CXX object CMakeFiles/ADC24RSR.dir/_deps/juce_audio_processors/src/modules/juce_audio_processors/juce_audio_processors.o
11 [ 10%] Building CXX object CMakeFiles/ADC24RSR.dir/_deps/juce_audio_processors/src/modules/juce_audio_processors/juce_audio_processors.o
12 [ 12%] Building CXX object CMakeFiles/ADC24RSR.dir/_deps/juce_audio_processors/src/modules/juce_audio_processors/juce_audio_processors.o
13 [ 14%] Building CXX object CMakeFiles/ADC24RSR.dir/_deps
```

Test case 1: Build a JUCE VST3 plugin



Triggered via push 20 minutes ago

fergarciadc pushed 2ff919e
[feature/01-basic-gha-work...](#)

Status	Total duration	Billable time	Artifacts
Success	2m 8s	2m	1

01-mac-workflow.yml

on: push

Build on macos-14 2m 0s

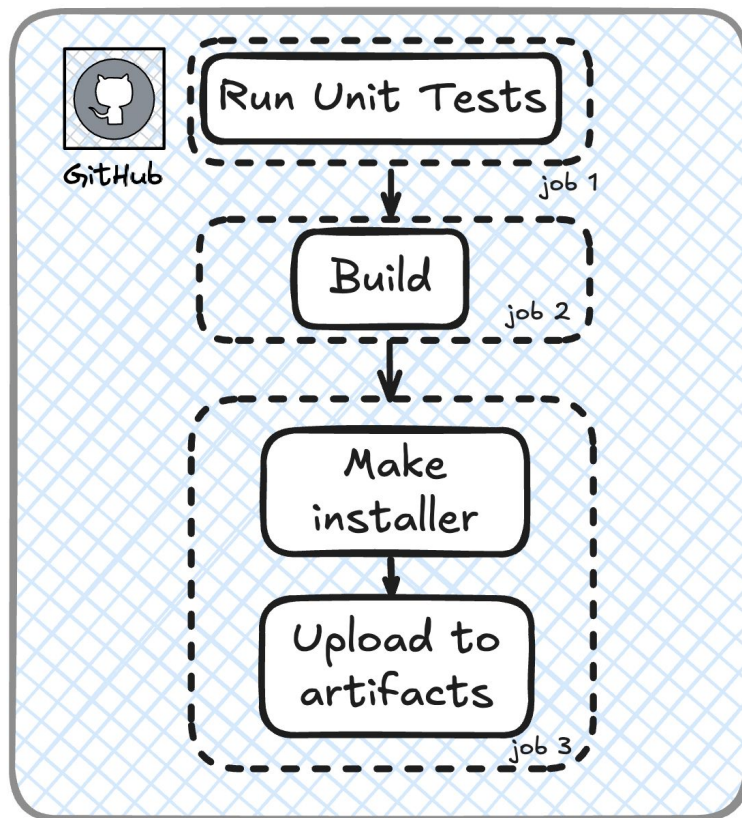


Artifacts

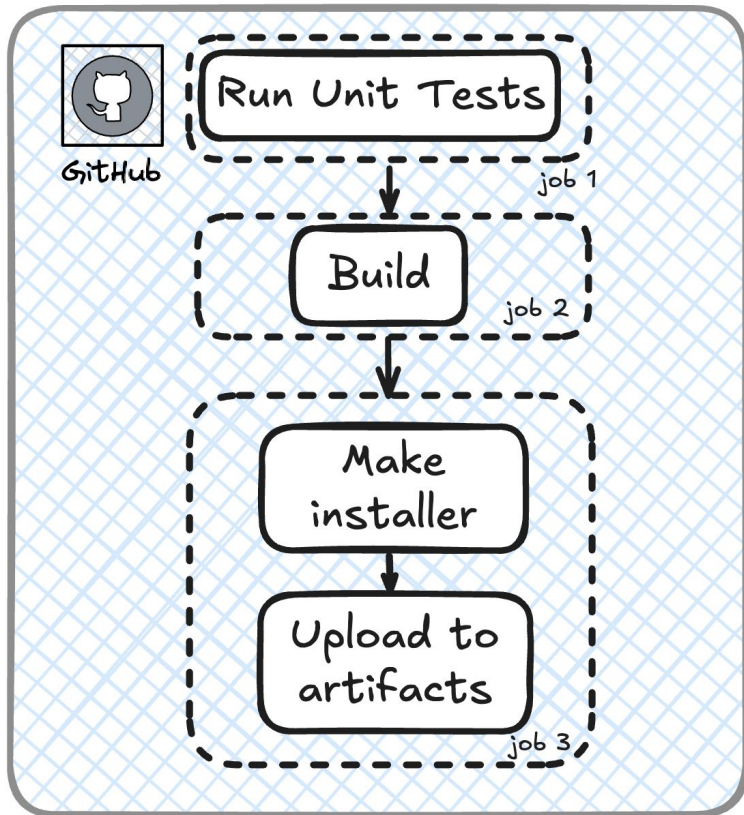
Produced during runtime

Name	Size		
ADC24RSR-builds-artifact	11.6 MB		

Test case 2: Jobs dependencies and run on condition

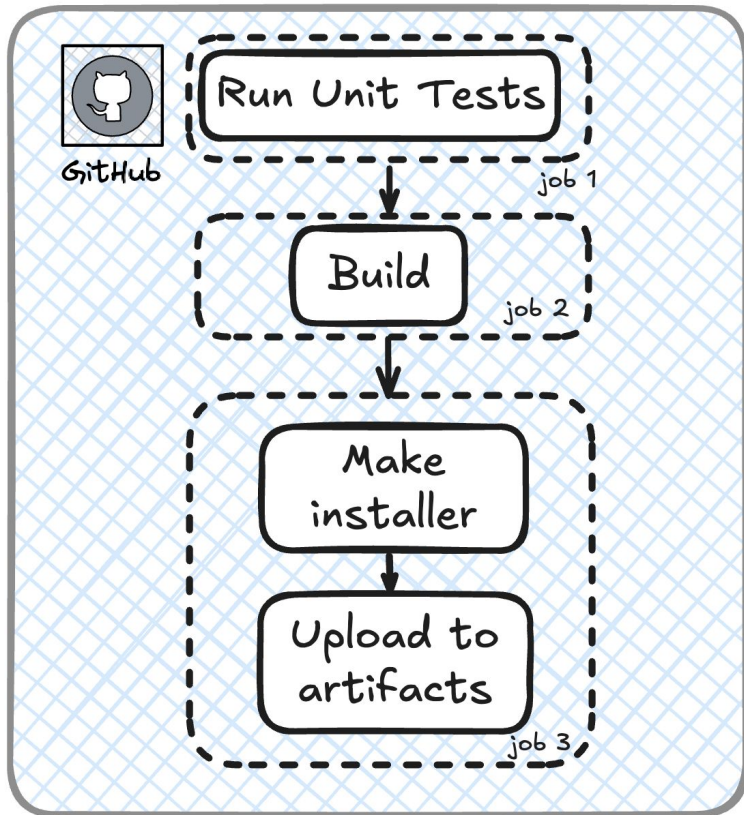


Test case 2: Jobs dependencies and run on condition



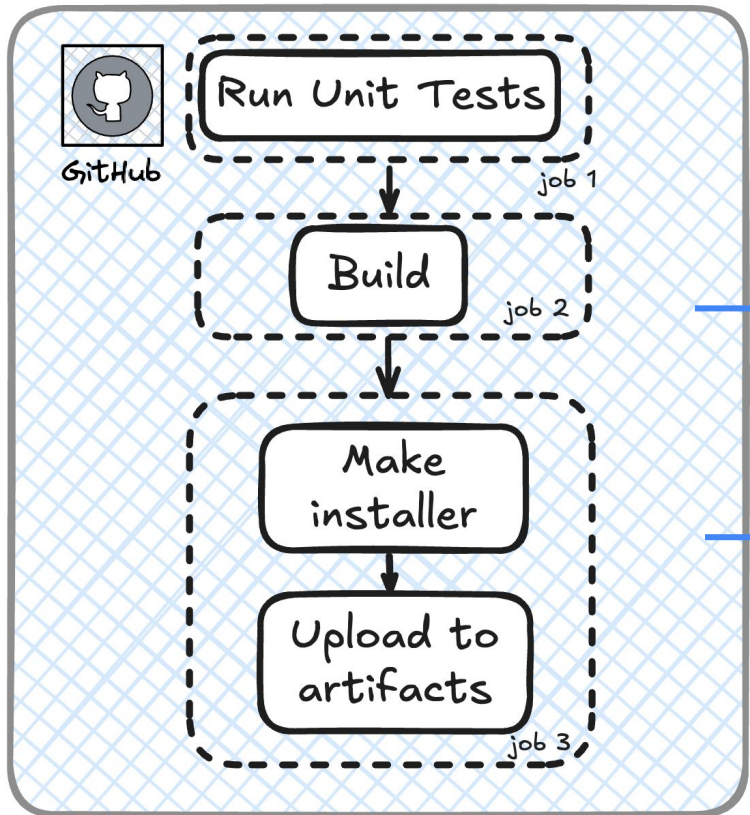
```
jobs:
  unit-test:
    name: Unit Testing on macos-14
    runs-on: macos-14
    if: contains(github.event.head_commit.message, '/mac-release')
    steps: ...
  build:
    name: Build on macos-14
    needs: [unit-test]
    runs-on: macos-14
    if: contains(github.event.head_commit.message, '/no-build') == false
    steps: ...
  create-dmg:
    name: Create DMG Package
    needs: [build]
    runs-on: macos-14
    if: contains(github.event.head_commit.message, '/no-installer') == false
    steps:
      - name: Download Plugins Folder Artifact ...
      - name: Create DMG File ...
      - name: Upload DMG Artifact ...
```


Test case 2: Jobs dependencies and run on condition



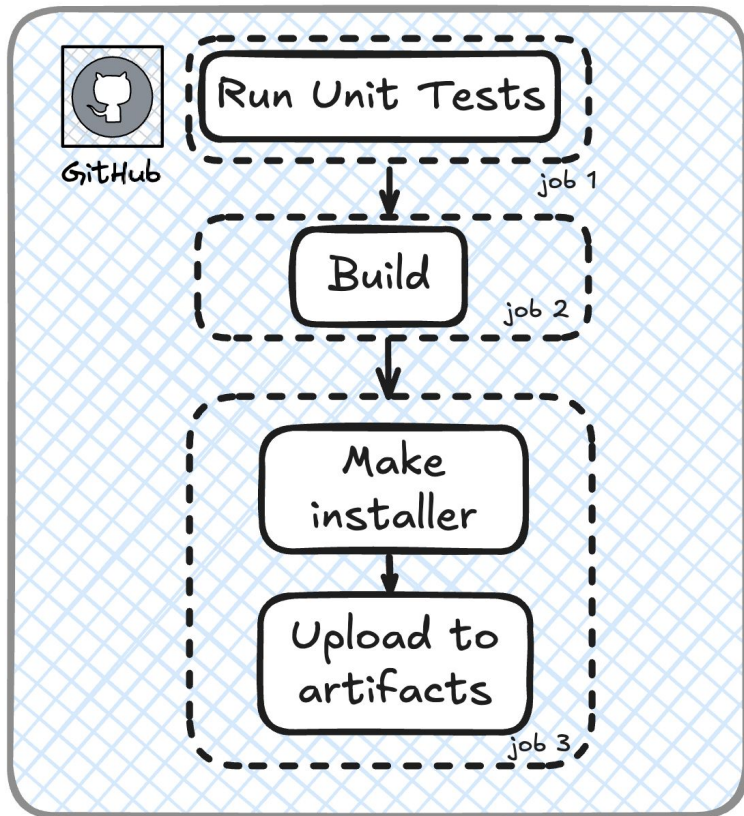
```
jobs:
  unit-test:
    name: Unit Testing on macos-14
    runs-on: macos-14
    if: contains(github.event.head_commit.message, '/mac-release')
    steps: ...
  build:
    name: Build on macos-14
    needs: [unit-test]
    runs-on: macos-14
    if: contains(github.event.head_commit.message, '/no-build') == false
    steps: ...
  create-dmg:
    name: Create DMG Package
    needs: [build]
    runs-on: macos-14
    if: contains(github.event.head_commit.message, '/no-installer') == false
    steps:
      - name: Download Plugins Folder Artifact ...
      - name: Create DMG File ...
      - name: Upload DMG Artifact ...
```

Test case 2: Jobs dependencies and run on condition



```
jobs:
  unit-test:
    name: Unit Testing on macos-14
    runs-on: macos-14
    if: contains(github.event.head_commit.message, '/mac-release')
    steps: ...
  build:
    name: Build on macos-14
    needs: [unit-test]
    runs-on: macos-14
    if: contains(github.event.head_commit.message, '/no-build') == false
    steps: ...
  create-dmg:
    name: Create DMG Package
    needs: [build]
    runs-on: macos-14
    if: contains(github.event.head_commit.message, '/no-installer') == false
    steps:
      - name: Download Plugins Folder Artifact ...
      - name: Create DMG File ...
      - name: Upload DMG Artifact ...
```

Test case 2: Jobs dependencies and run on condition



Triggered via push 6 minutes ago

fergarcia1dc pushed -> bf4d8ed feature/02-basic-gha-work...

Status: **Success** Total duration: **2m 45s**

Billable time: **4m** Artifacts: **2**

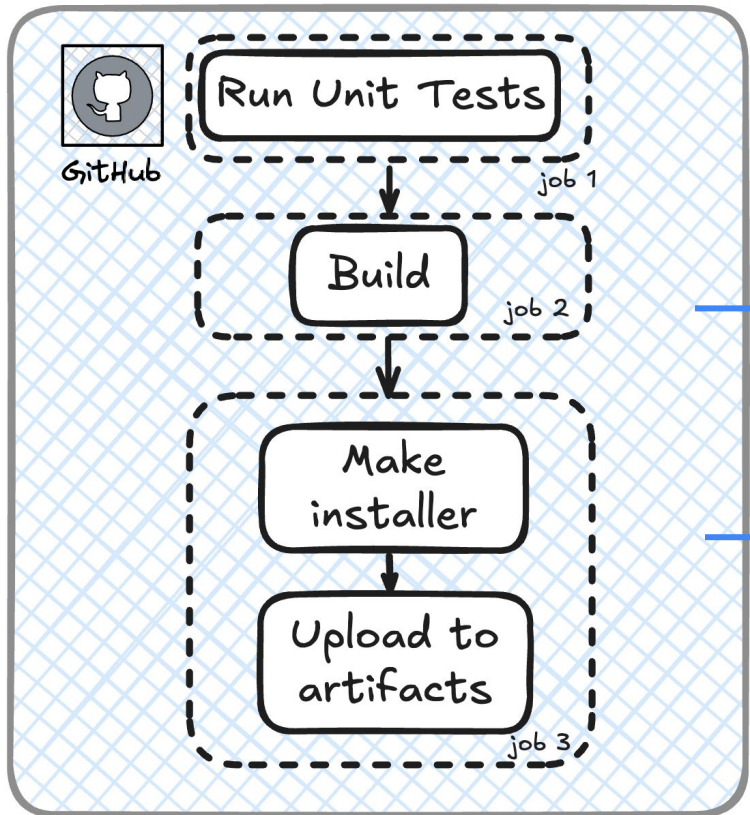
02-mac-workflow.yml
on: push

Unit Testing on macos-14 5s — Build on macos-14 1m 58s — Create DMG Package

Artifacts
Produced during runtime

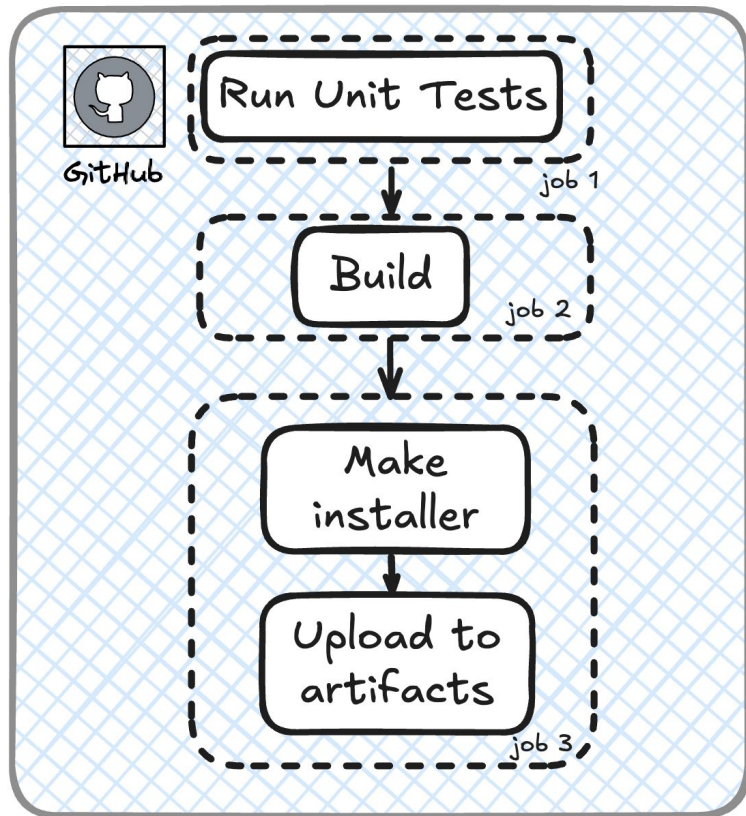
Name	Size	
ADC24RSR-builds-artifact	11.6 MB	📄 🗑️
ADC24RSR-dmg-artifact	14 MB	📄 🗑️

Test case 2: Jobs dependencies and run on condition



```
jobs:
  unit-test:
    name: Unit Testing on macos-14
    runs-on: macos-14
    if: contains(github.event.head_commit.message, '/mac-release')
    steps: ...
  build:
    name: Build on macos-14
    needs: [unit-test]
    runs-on: macos-14
    if: contains(github.event.head_commit.message, '/no-build') == false
    steps: ...
  create-dmg:
    name: Create DMG Package
    needs: [build]
    runs-on: macos-14
    if: contains(github.event.head_commit.message, '/no-installer') == false
    steps:
      - name: Download Plugins Folder Artifact ...
      - name: Create DMG File ...
      - name: Upload DMG Artifact ...
```

Test case 2: Jobs dependencies and run on condition



Triggered via push 6 minutes ago

fergarcia1c pushed -> bf4d8ed feature/02-basic-gha-work... **Success** Total duration 2m 45s

Billable time 4m Artifacts 2

ADC24RSR

2 items

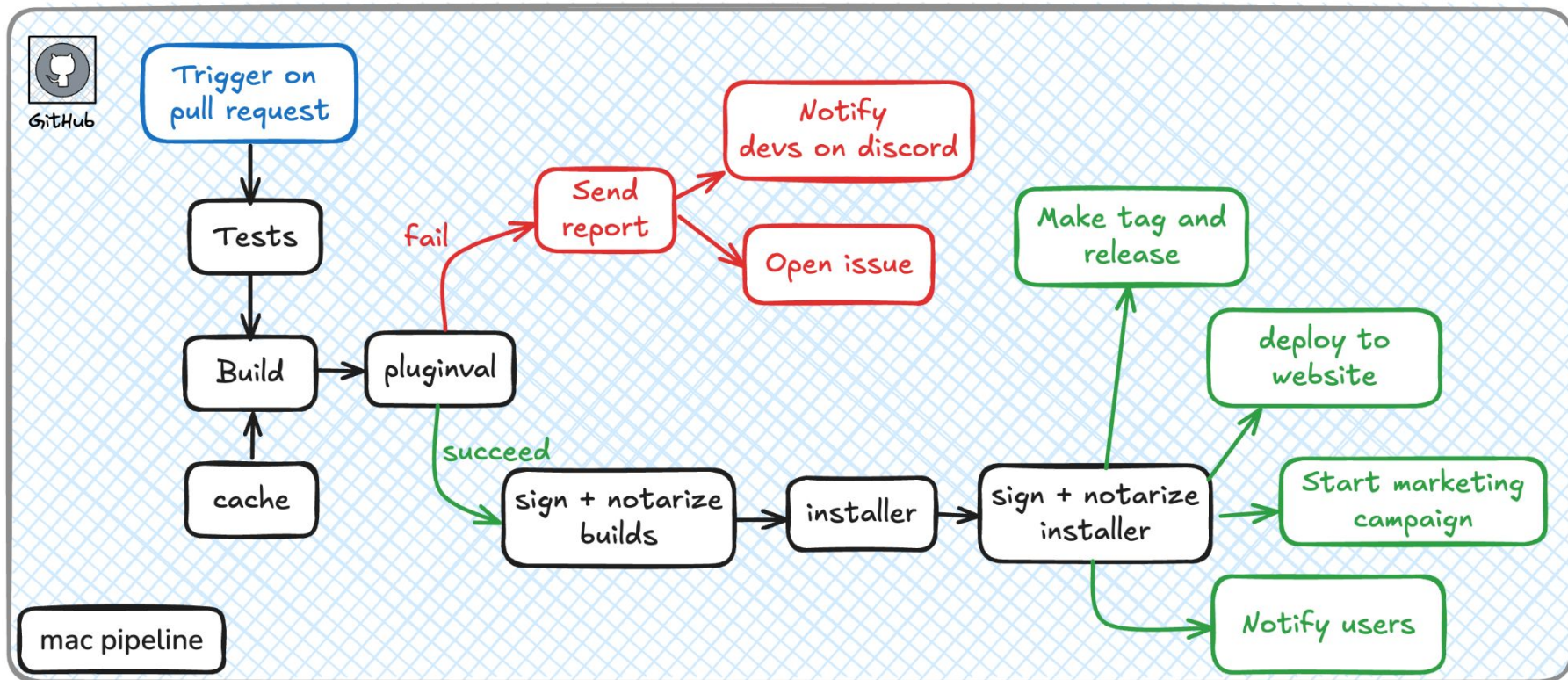
ADC24RSR.component ADC24RSR.vst3

Unit Testing on mac... Create DMG Package

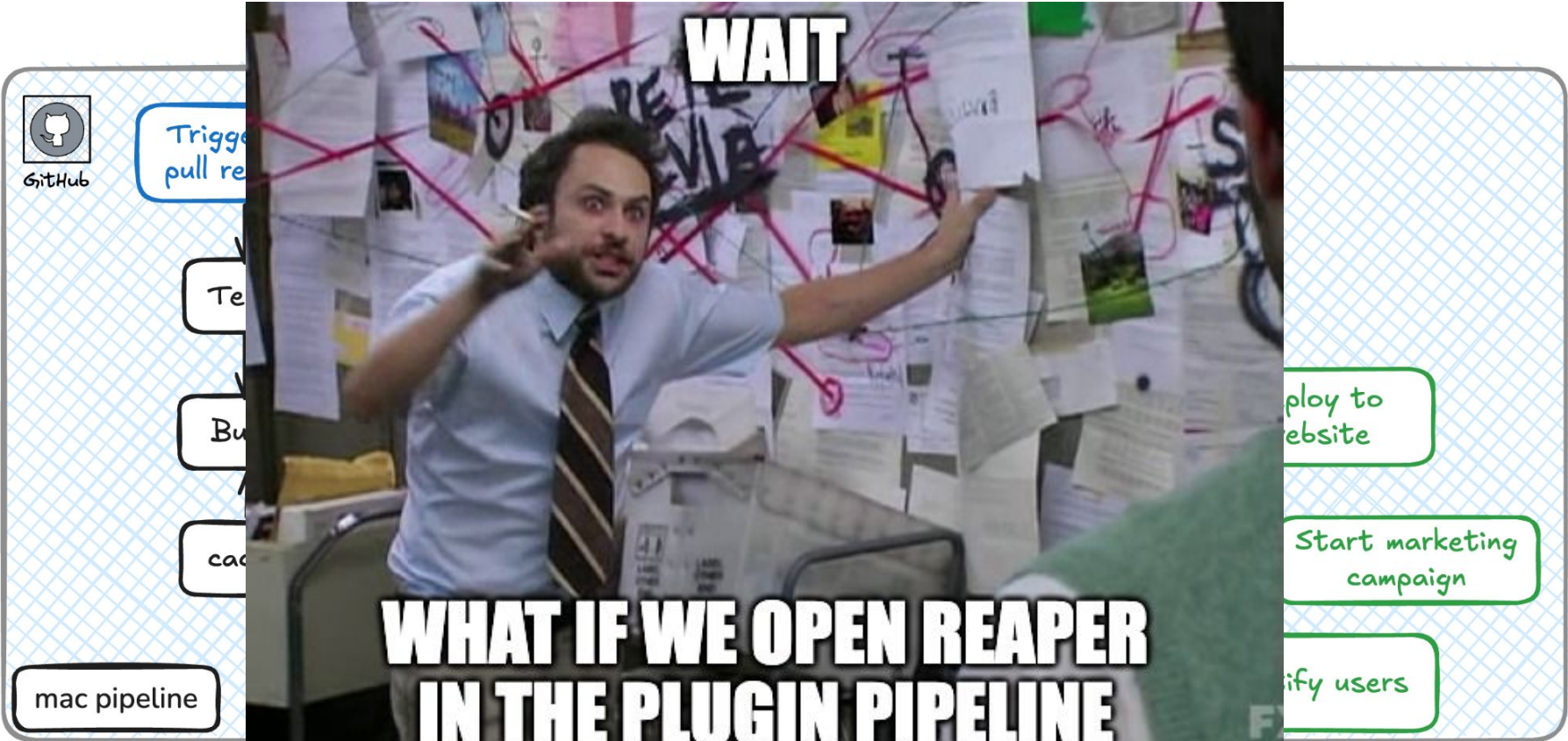
Artifacts
Produced during runtime

Name	Size	
ADC24RSR-builds-artifact	11.6 MB	📄 🗑️
ADC24RSR-dmg-artifact	14 MB	📄 🗑️

Pipeline idea



Pipeline idea



Focus on the things you **really** need



You've used 75% of included services for the cepstrum-dsp account



GitHub Actions usage

Hello! We wanted to provide you with an update on your usage and spending.

Private repository usage 1,511.00 of 2,000.00 mins included



You've used 75% of included services for GitHub Actions.
To continue using Actions & Packages uninterrupted, update your spending limit.

Your usage will reset on November 01, 2024.

[Update spending limit](#)

Optimize build process and artifacts storage

```
.github > workflows > ! 03-mac-workflow.yml
12 jobs:
35   build:
41     steps:
50 >     - name: Configure CMake
51 >       run: |
52 >         cmake -B build \
53 >           -DCMAKE_BUILD_TYPE=Release
54 >
55 >     - name: Build Plug-In
56 >       run: |
57 >         sudo cmake --build build --config Release --parallel 3
58 >
59 >     - name: Move plugins to builds folder...
65 >
66 >     - name: Upload Builds to Artifact
67 >       uses: actions/upload-artifact@v4
68 >       with:
69 >         name: "${{ env.PLUGIN_NAME }}-builds-artifact"
70 >         path: ${ env.PLUGINS_FOLDER }/builds
71 >         retention-days: 5 # Limit artifact retention
72 >
73 >     create-dmg: ...
104 >
105 >     cleanup-artifacts: ...
```

Protip: Save minutes using cache!

```
.github > workflows > ! 04-mac-workflow.yml
12 jobs:
35   build:
41     steps:
40
47     - name: Get latest CMake
48       uses: lukka/get-cmake@latest
49
50 > - name: Setup ccache...
54
55     - name: Configure CMake
56       run: |
57         cmake -B build \
58           -DCMAKE_BUILD_TYPE=Release \
59           -DCMAKE_C_COMPILER_LAUNCHER=ccache \
60           -DCMAKE_CXX_COMPILER_LAUNCHER=ccache
61
62     - name: Build Plug-In
63       run: |
64         sudo cmake --build build --config Release --parallel 3
65       env:
66         CCACHE_DIR: ${github.workspace}/.ccache
67
```

Before cache

← Mac Pipeline

✓ test without cache /mac-release /no-cleanup #33

Re-run all jobs



Summary

Jobs

- ✓ Unit Testing on macos-14
- ✓ Build on macos-14
- ✓ Create DMG Package
- ✓ Cleanup Artifacts

Run details

- 🔄 Usage
- 📄 Workflow file

Triggered via push 3 days ago

🌐 fergarciadlc pushed -> 1b2d7cb `main`

Artifacts

2

Status

Success

Total duration

3m 39s

Billable time

6m

mac-workflow.yml

on: push



After cache

← Mac Pipeline

✓ test with cache /mac-release /no-cleanup #34

Re-run all jobs



Summary

Jobs

- ✓ Unit Testing on macos-14
- ✓ Build on macos-14
- ✓ Create DMG Package
- ✓ Cleanup Artifacts

Run details

- 🕒 Usage
- 📄 Workflow file

Triggered via push 3 days ago

👤 fergarciadlc pushed ↻ 3c8485d `main`

Artifacts

2

Status

Success

Total duration

2m 32s

Billable time

5m

mac-workflow.yml

on: push

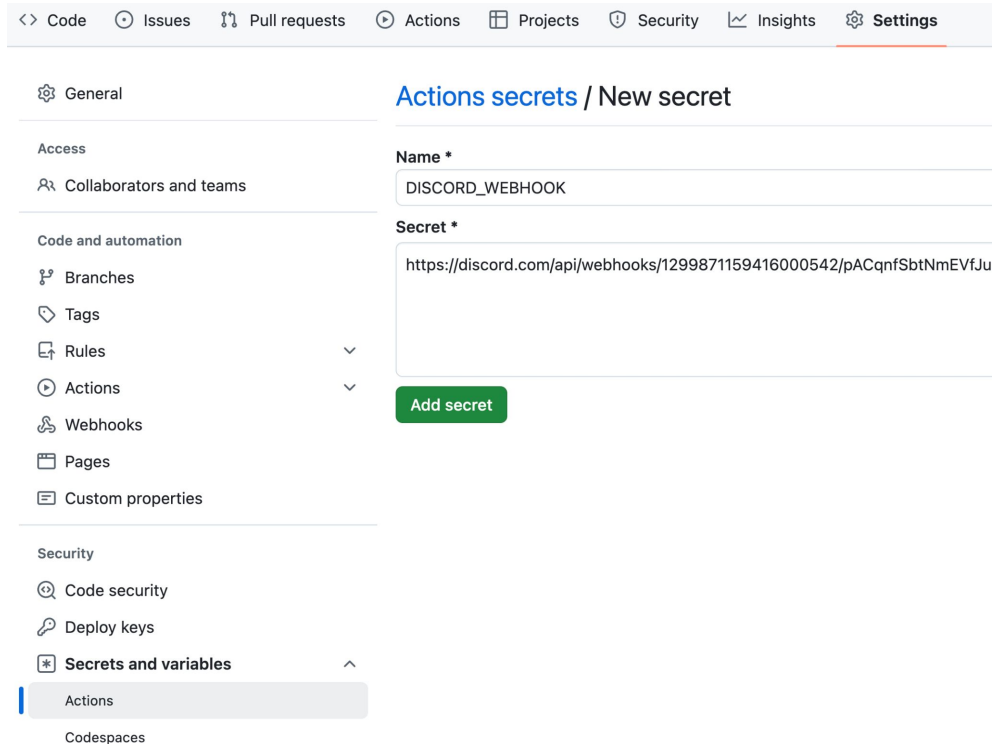
✓ Unit Testing on macos-14 11s

✓ Build on macos-14 1m 31s

Cleanup Artifacts 3s



Protip: Use secrets



The screenshot shows the GitHub Actions secrets configuration page. The top navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Security, Insights, and Settings. The left sidebar is divided into sections: General, Access, Code and automation, Security, and Secrets and variables. The 'Secrets and variables' section is expanded, showing 'Actions' selected. The main content area is titled 'Actions secrets / New secret' and contains a form with two fields: 'Name *' with the value 'DISCORD_WEBHOOK' and 'Secret *' with the value 'https://discord.com/api/webhooks/1299871159416000542/pACqnfSbtNmEVfJu'. A green 'Add secret' button is located below the form.

<> Code Issues Pull requests Actions Projects Security Insights Settings

General

Access

Collaborators and teams

Code and automation

Branches

Tags

Rules

Actions

Webhooks

Pages

Custom properties

Security

Code security

Deploy keys

Secrets and variables

Actions

Codespaces

Actions secrets / New secret

Name *

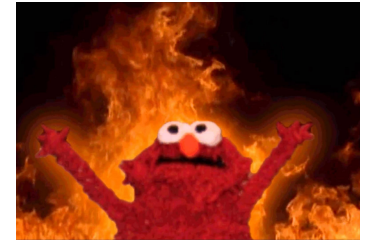
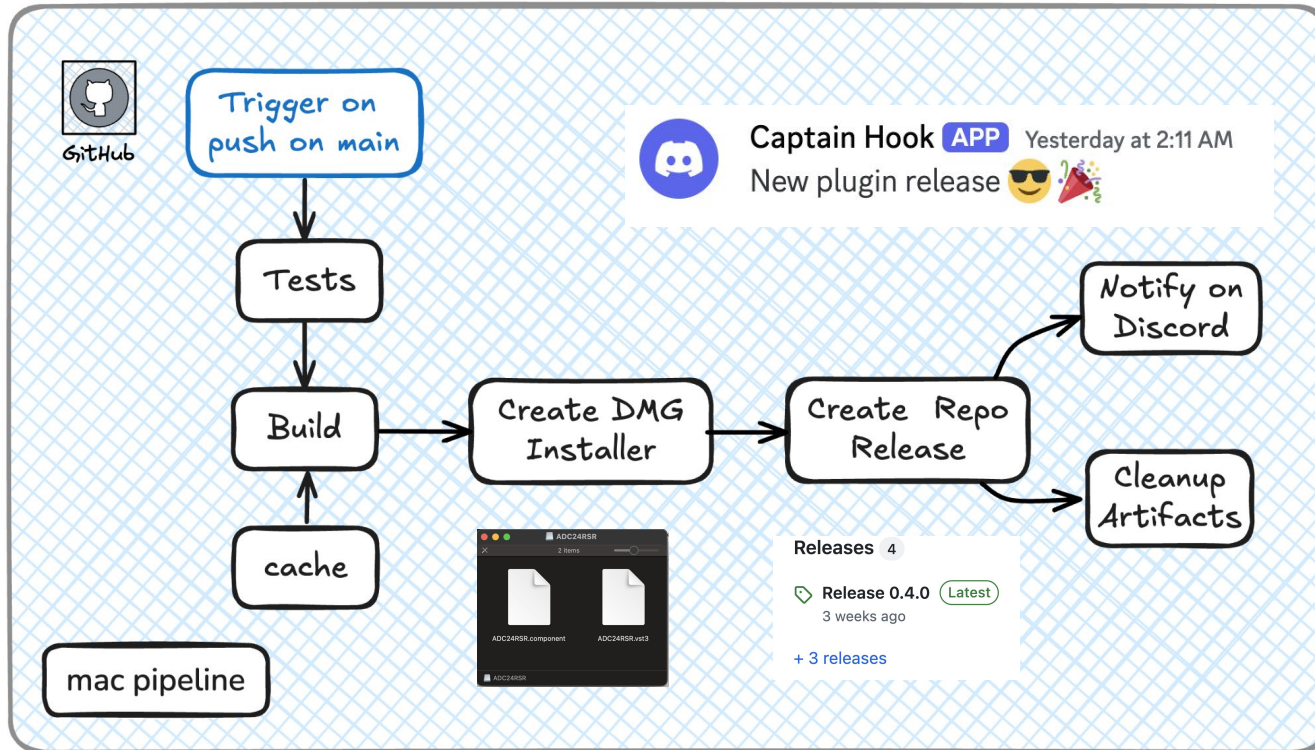
DISCORD_WEBHOOK

Secret *

https://discord.com/api/webhooks/1299871159416000542/pACqnfSbtNmEVfJu

Add secret

Your First Rock-Solid Release Pipeline!



Tips and tricks

Parallel

EnvVars

Secrets

Cached build

Personalized notifications to users for plugins updates

Check open source projects and “borrow” ideas

Use LLMs for debugging errors

Is that it?

Still lot of work to do!

Challenges:

- Mac: signing, notarization
- Windows: sign, EV cert
- AAX more complex



Takeaways

Using CI/CD is great for your audio plugins

Use and experiment with github actions

Be creative, but try not to complicate things

Focus on things you only need now, is ok to do some stuff manually...

Make mistakes, learn, share and have fun!!

Cool resources

[ChowDSP](#) ([Jatin Chowdhury](#))

[Pamplejuce](#) ([Sudara Williams](#))

[OJD](#) ([Janos](#))

ADC21: [CI/CD for Audio Plugin Development - Jatin Chowdhury - ADC21](#)

Thanks a lot

Ear Candy Technologies

ADC: Organizers & Volunteers

My mom

You, for your attention :)

Keep in touch!



Fernando Garcia de la Cruz

<https://fergarcia DLC.github.io/>

<https://www.linkedin.com/in/fergarcia DLC/>

<https://github.com/fergarcia DLC>