


Sure, here's a step-by-step guide to set up GitHub Actions to automatically update your Anaconda environment when a new version is released:

Step 1: Create a GitHub repository:

If you haven't already, create a GitHub repository where you'll host your project.

Step 2: Add an environment file to your repository:

Create a YAML file (e.g., `environment.yml`) in your repository's root directory. This file will specify the packages and versions you want to include in your Anaconda environment. Here's an example `environment.yml`:

Code Blame 22 lines (22 loc) · 391 Bytes  Code 55% faster with GitHub Copilot

```
1  # To use:
2  #
3  # $ conda env create -f environment.yml # `mamba` works too for this command
4  # $ conda activate demommath
5  #
6  name: demommath2
7  channels:
8    - defaults
9  dependencies:
10   - python #need to pin to avoid issues with builds
11   - cython>=3.0
12   - setuptools
13   - pkg-config
14   - meson-python
15   - pip
16   - spin
17   - ccache
18   # For testing
19   - pytest
20   - pytest-cov
21   - pytest-xdist
22   - numpy
```

Step 3. Set up GitHub Actions:

Create a directory named `.github/workflows` in your repository if it doesn't exist already. Inside this directory, create a YAML file (e.g., `update_environment.yml`) to define your GitHub Actions workflow.

Choose a workflow

Build, test, and deploy your code. Make code reviews, branch management, and issue triaging work the way you want. Select a workflow to get started.

Skip this and [set up a workflow yourself](#) →

Q Search workflows

Suggested for this repository

Python Package using Anaconda
 By GitHub Actions
 Create and test a Python package on multiple Python versions using Anaconda for package management.
[Configure](#) Python

Publish Python Package
 By GitHub Actions
 Publish a Python Package to PyPI on release.
[Configure](#) Python

Django
 By GitHub Actions
 Build and Test a Django Project
[Configure](#) Python

Pylint
 By GitHub Actions
 Lint a Python application with pylint.
[Configure](#) Python

Python application
 By GitHub Actions
 Create and test a Python application.
[Configure](#) Python

Python package
 By GitHub Actions
 Create and test a Python package on multiple Python versions.
[Configure](#) Python

Step 4. Define the workflow:

Open `update_environment.yml` and define the workflow to update your Anaconda environment whenever changes are pushed to the repository or on a schedule. Here's a basic example:

```

``yaml
name: Update Anaconda Environment

on:
  push:
    branches:
      - main
  schedule:
    - cron: '0 0 * * *' # Run daily

jobs:
  update:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout repository
        uses: actions/checkout@v2

      - name: Set up Anaconda
        uses: conda-incubator/setup-miniconda@v2
  
```

with:

python-version: '3.10'

environment-file: environment.yml

- name: Update environment

run: conda env update -n myenv -f environment.yml --prune

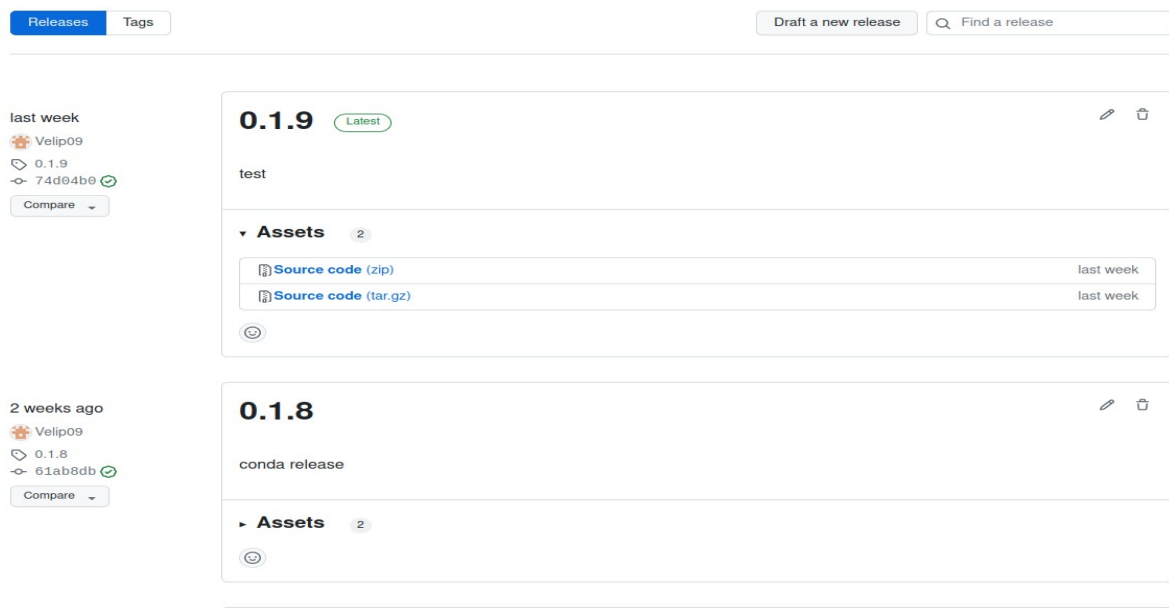
...

This workflow will:

- Run on every push to the `main` branch.
- Run daily at midnight to check for updates.
- Set up a Python environment based on the `environment.yml` file.
- Update the environment with any new package versions specified in `environment.yml`.

Step 5: Commit and push your changes:

Commit both `environment.yml` and the
`.github/workflows/update_environment.yml` files to your repository and
push the changes to GitHub.



Step 6: Monitor workflow runs:

You can monitor the execution of the GitHub Actions workflow in the "Actions" tab of your repository on GitHub. Ensure that the workflow runs successfully and updates your Anaconda environment as expected.

Build (and upload)

succeeded last week in 4m 47s

Beta Give feedback

Search logs

Re-run

Settings

> ✓ Set up job	2s
> ✓ Run actions/checkout@v4.1.1	0s
> ✓ Set up Python	21s
> ✓ Update conda and conda-build	30s
> ✓ Set up Anaconda	25s
> ✓ Install Dependencies	15s
> ✓ Display Conda Configuration	0s
> ✓ Build Package	2m 53s
> ✓ Upload conda package	16s
> ✓ Post Set up Anaconda	1s
> ✓ Post Set up Python	0s
> ✓ Post Run actions/checkout@v4.1.1	0s
> ✓ Complete job	0s

That's it! Your Anaconda environment will now be automatically updated whenever changes are pushed to the repository or on a daily schedule. Adjust the workflow triggers and schedule according to your specific requirements.