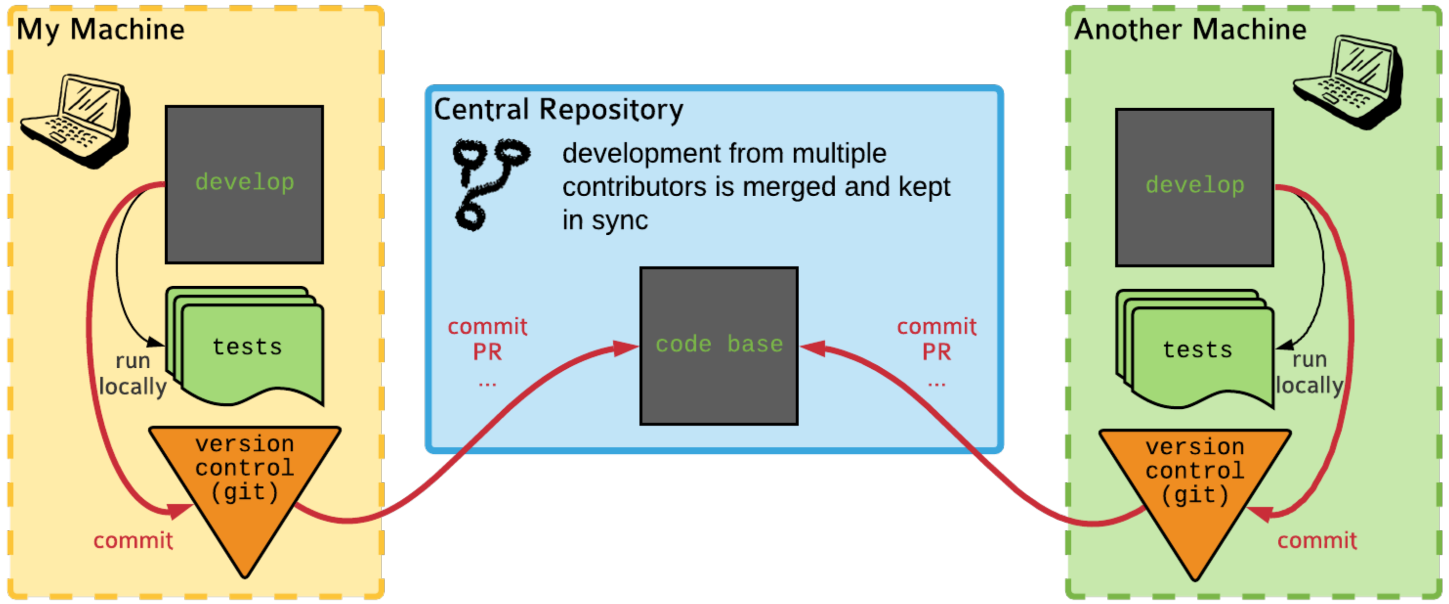
Continuous Integration

Because you’re worth it, continuously

Lisa Schwetlick and Pietro Berkes

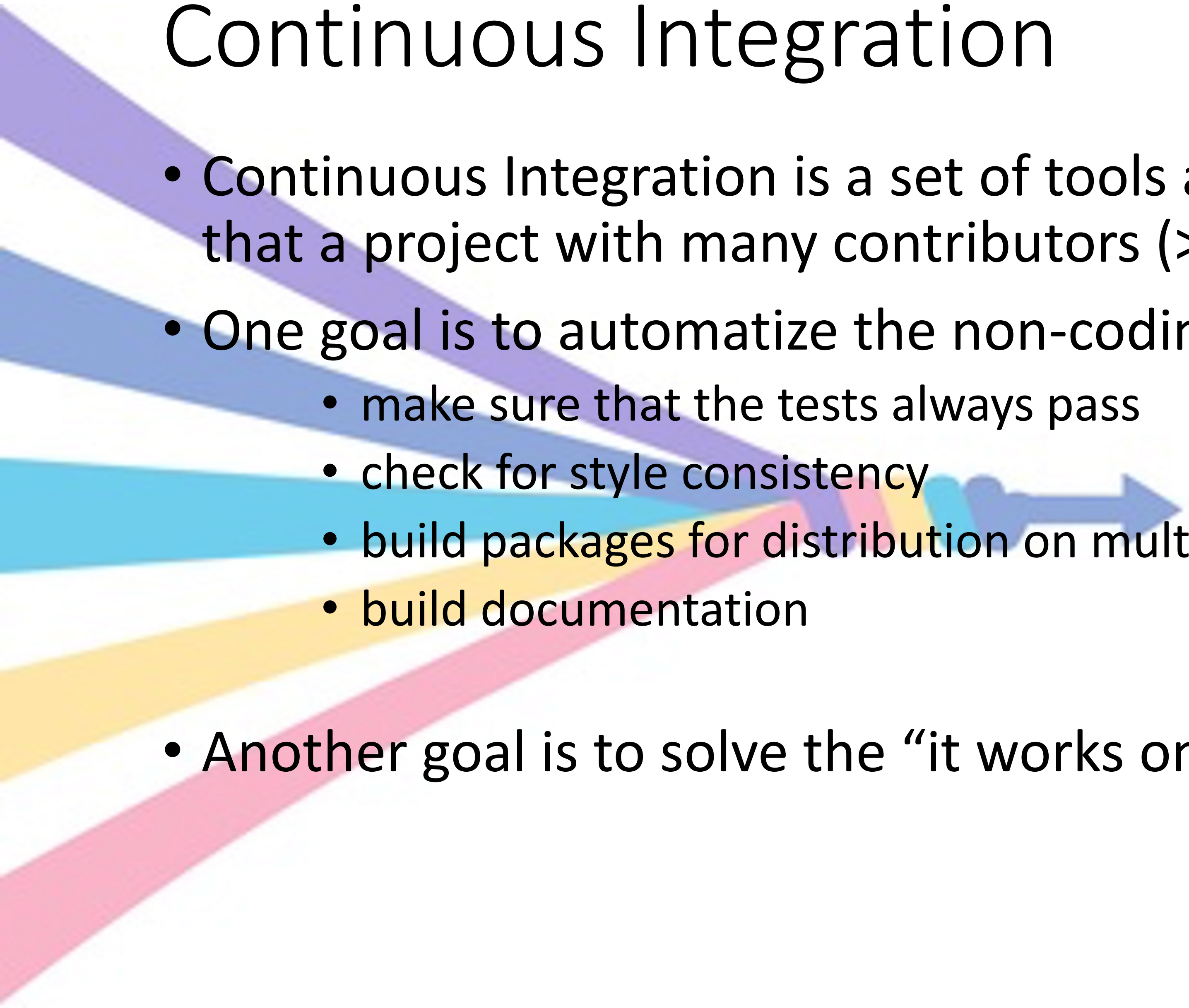
Collaborative Development without CI



Potential issues

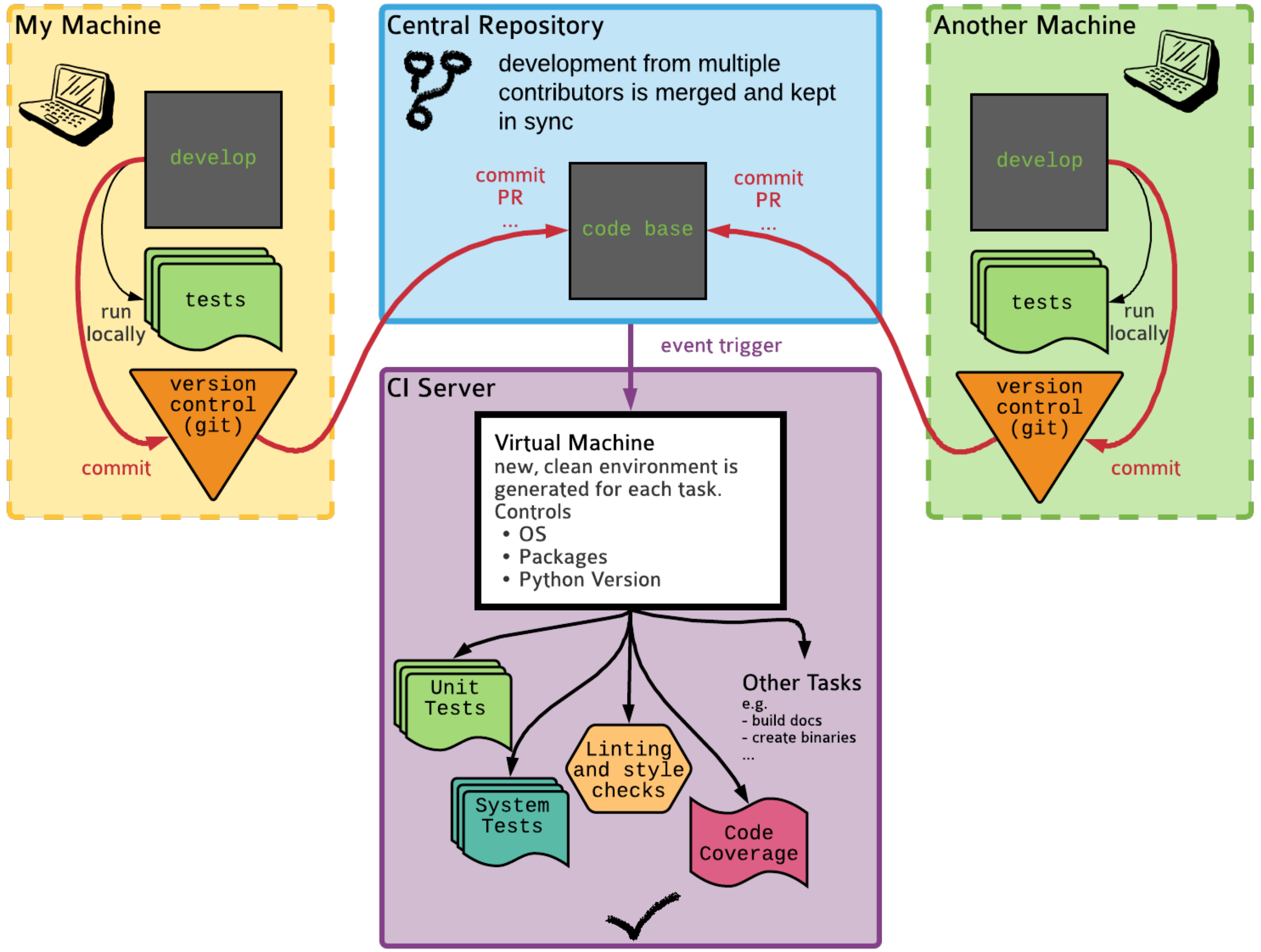
- The tests might pass on one machine and/or the other, but not in a third-party environment (versions, OS, etc.)
- A maintainer needs to ensure that the software works on all the supported combinations of versions / OSs
- A maintainer needs to create and upload artifacts like binary packages, documentation, etc

Continuous Integration



- Continuous Integration is a set of tools and practices to make sure that a project with many contributors (>= 1) runs smoothly
- One goal is to automatize the non-coding tasks:
    - make sure that the tests always pass
    - check for style consistency
    - build packages for distribution on multiple architectures
    - build documentation
- Another goal is to solve the “it works on my machine” problem

Collaborative Development with CI



The CI tasks that you’ll find 95% of the time

- **Task 1: Run test when a PR is created**
  - **Event trigger:** PR is created
  - **Action:** Run all tests for different Python versions

- **Task 2: Release package when version is bumped**
  - **Event trigger:** Version is bumped
  - **Action:** Create binary packages for Linux, Mac, Windows and upload them to a package repository

- **Task 3: Publish documentation on request**
  - **Event trigger:** Repository is tagged in a certain way
  - **Action:** Build and publish the documentation

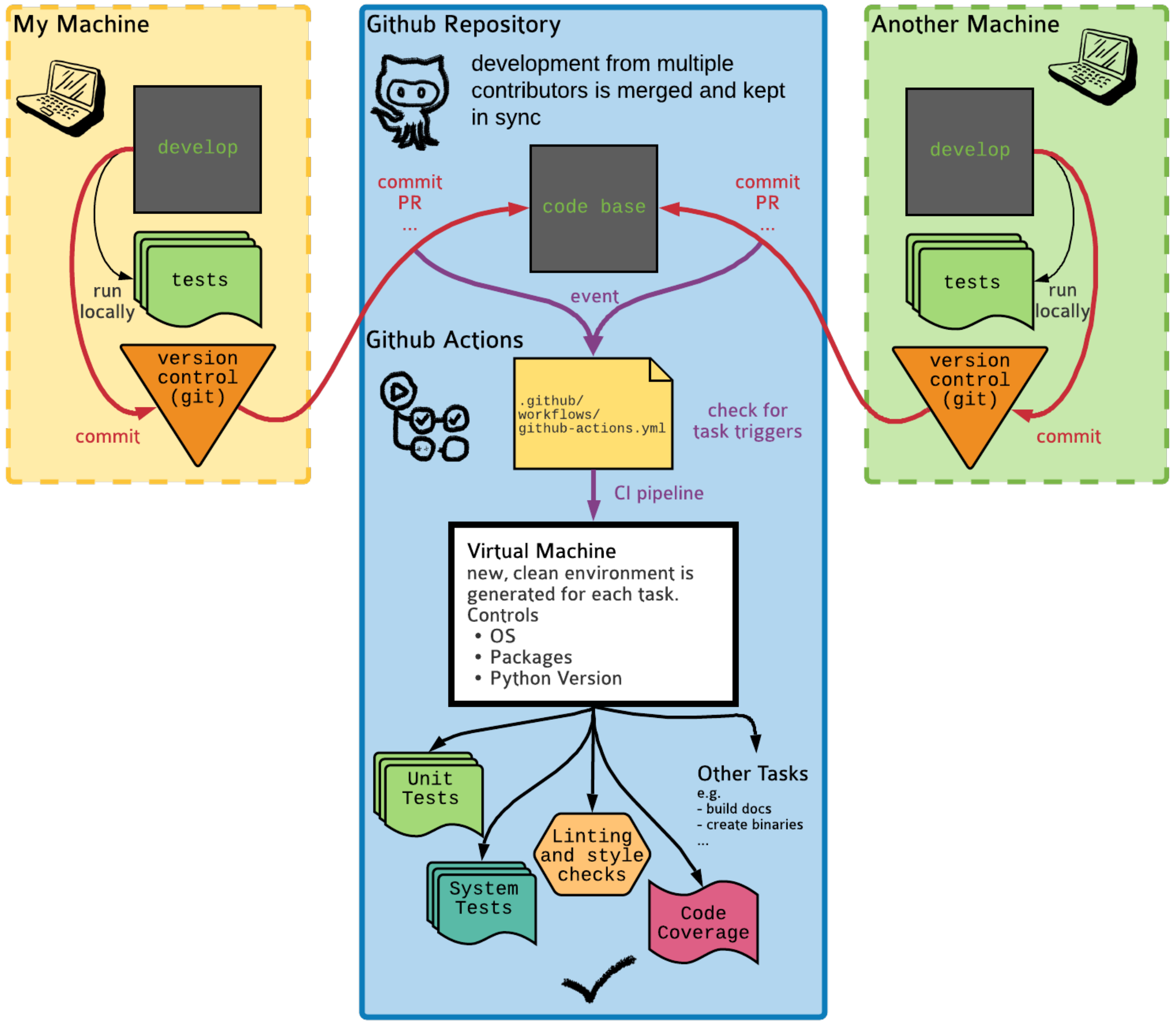
CI options

Travis CI

circleci

GitHub Actions is at the moment the preferred choice for many open source projects. It is very flexible and well integrated with GitHub.

Collaborative Development with GitHub Actions



GitHub acts as both the central repository and the CI server, but the rest is the same

GitHub Actions basic ideas

An event occurs, it has an associated commit SHA
(e.g., a PR is opened or a commit tag is pushed)

⬇

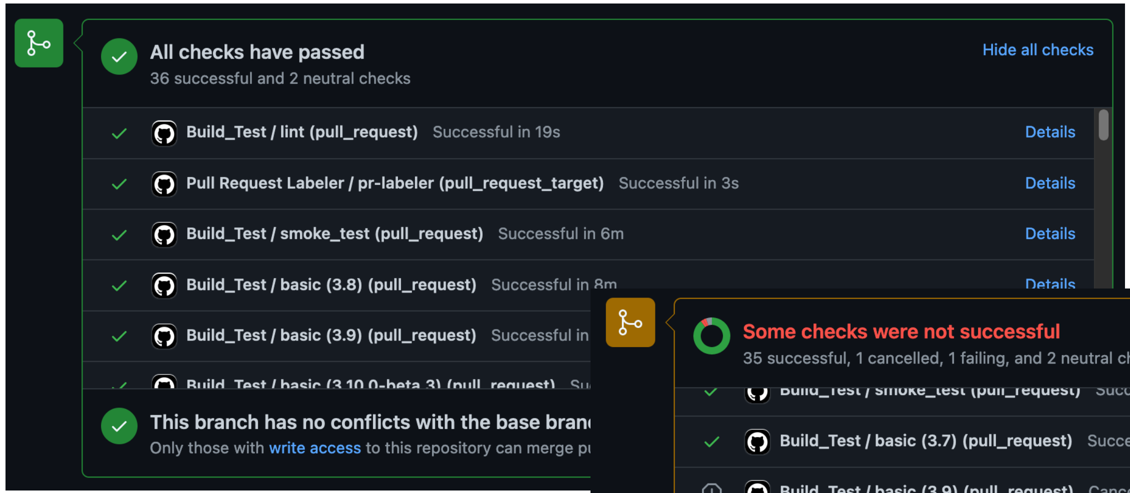
GitHub searches for config files in `.github/workflows` at that SHA, and looks if there is a trigger that matches the event

⬇

It then creates a virtual machine as specified in the config file and runs the commands listed there

GitHub Actions basic ideas

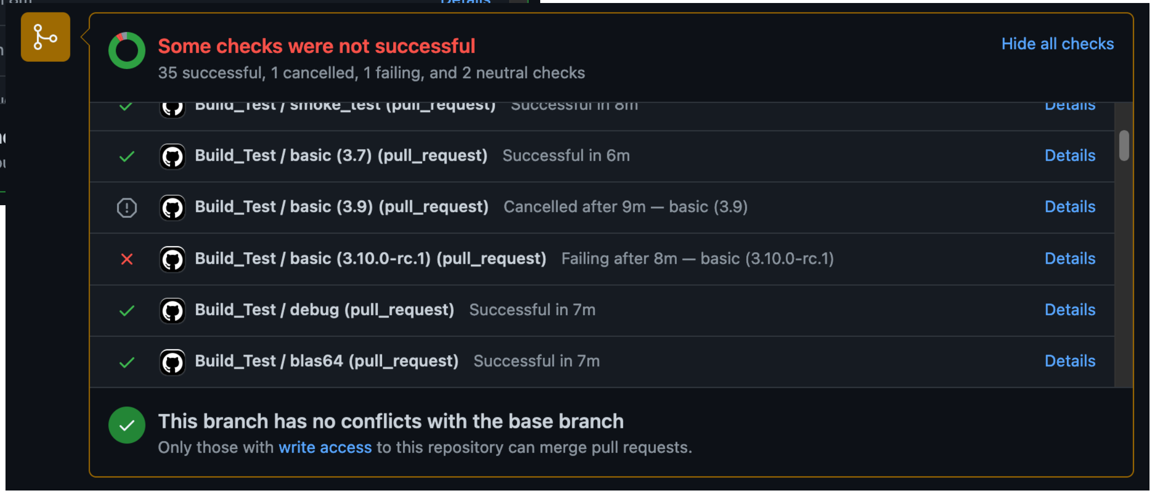
- The outcome is logged and if the job exits cleanly it is marked as ”passed” otherwise “failed”



All checks have passed
36 successful and 2 neutral checks
Hide all checks

✓ Build_Test / lint (pull_request) Successful in 19s Details
✓ Pull Request Labeler / pr-labeler (pull_request_target) Successful in 3s Details
✓ Build_Test / smoke_test (pull_request) Successful in 6m Details
✓ Build_Test / basic (3.8) (pull_request) Successful in 8m Details
✓ Build_Test / basic (3.9) (pull_request) Successful in
✓ Build_Test / basic (3.10.0-beta.3) (pull_request) Su

This branch has no conflicts with the base bran
Only those with write access to this repository can merge p

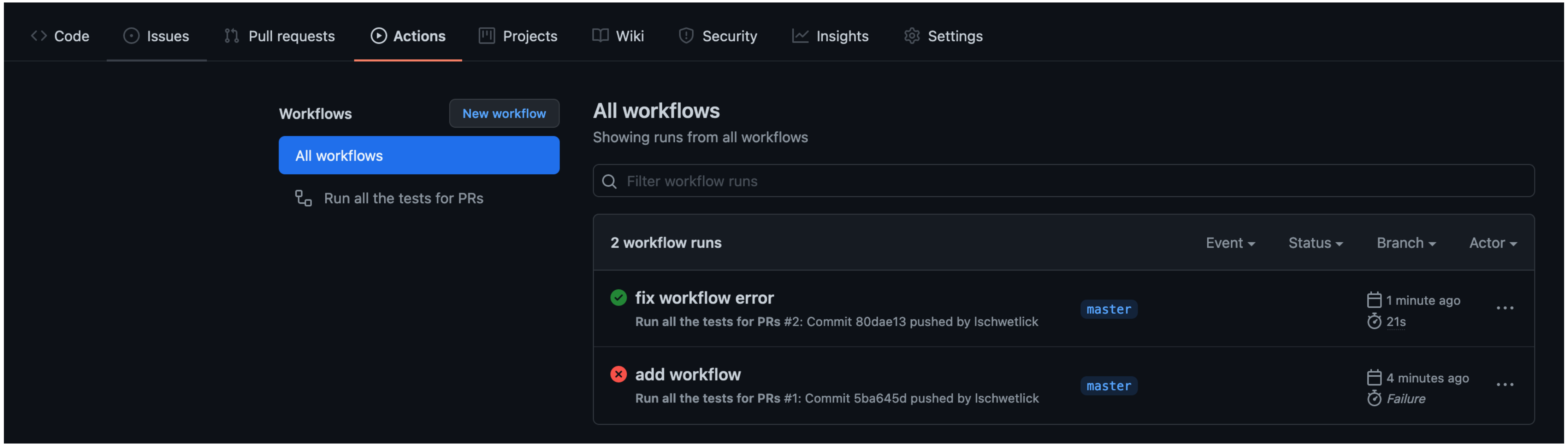


Some checks were not successful
35 successful, 1 cancelled, 1 failing, and 2 neutral checks
Hide all checks

✓ Build_Test / smoke_test (pull_request) Successful in 6m Details
✓ Build_Test / basic (3.7) (pull_request) Successful in 6m Details
ⓘ Build_Test / basic (3.9) (pull_request) Cancelled after 9m — basic (3.9) Details
✗ Build_Test / basic (3.10.0-rc.1) (pull_request) Failing after 8m — basic (3.10.0-rc.1) Details
✓ Build_Test / debug (pull_request) Successful in 7m Details
✓ Build_Test / blas64 (pull_request) Successful in 7m Details

This branch has no conflicts with the base branch
Only those with write access to this repository can merge pull requests.

GitHub Actions



All workflows

Showing runs from all workflows

2 workflow runs

fix workflow error
Run all the tests for PRs #2: Commit 80dae13 pushed by lschwetlick
master
1 minute ago
21s

add workflow
Run all the tests for PRs #1: Commit 5ba645d pushed by lschwetlick
master
4 minutes ago
Failure

GitHub config file: Example
Run tests every time a PR is opened or a commit is pushed

The configuration file is saved in `.github/workflows` , with a name related to its task, e.g. `run-tests.yml`

```yaml
name: Run all the tests for PRs
on:
  [push, pull_request]

jobs:
  run-tests:
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@v2
    - name: Set up Python
      uses: actions/setup-python@v2
      with:
        python-version: 3.9
    - name: Install dependencies
      run:
        python -m pip install pytest numpy
    - name: Test with pytest
      run:
        pytest -sv hands_on/pyanno_voting
```

Specifies the events that trigger the jobs below

The type of virtual machine used to run the workflow

Multiple steps are used to set up the environment so that we can run the tests.

Notice the use of community actions

The command that we wanted to execute all along

GitHub Actions reference

- Introduction: https://docs.github.com/en/actions/learn-github-actions/introduction-to-github-actions
- Events that can trigger actions, and their config options: https://docs.github.com/en/actions/reference/events-that-trigger-workflows#pull_request
- Catalog of community actions: https://github.com/marketplace?type=actions

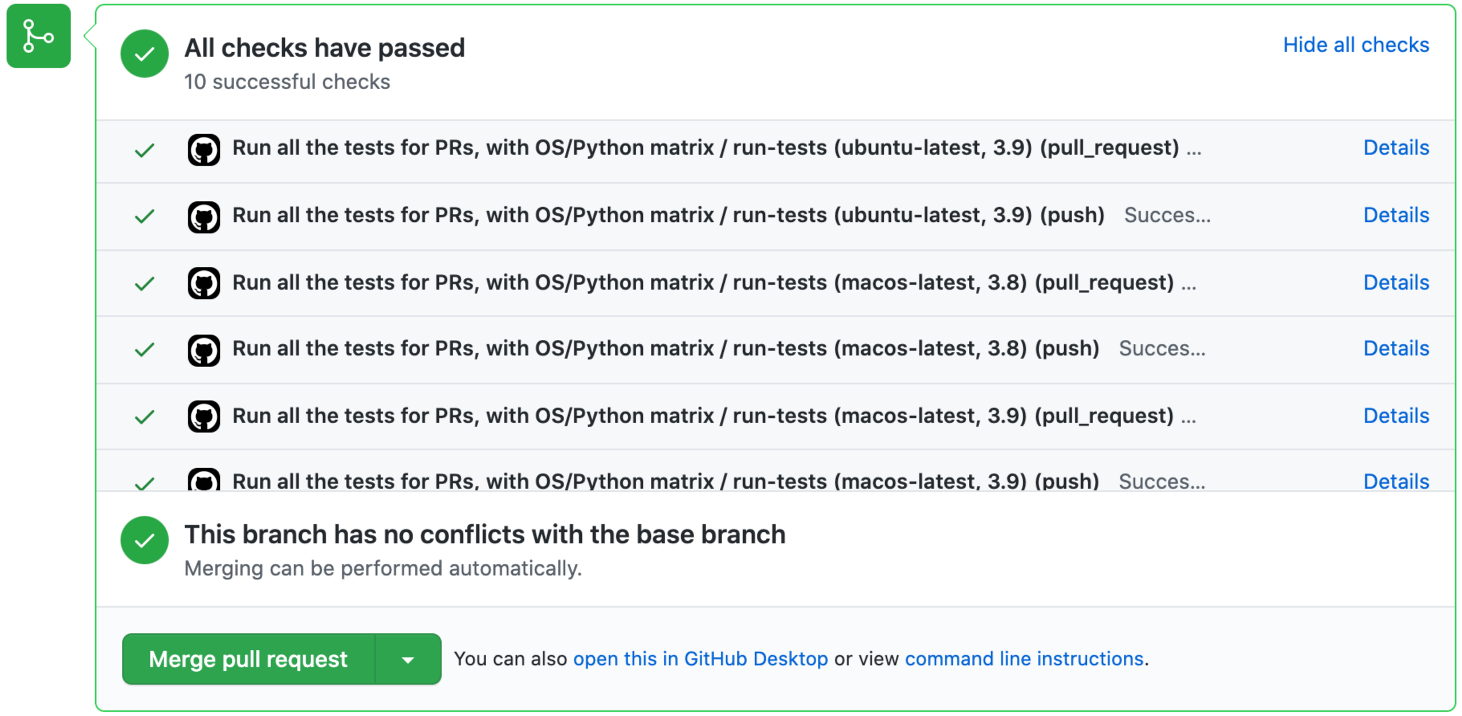
Hands On!

Add a CI pipeline to the CI project!

1. In your local version of the project make a folder `.github/workflows`
2. Create a file called `run_test_on_push.yml`
3. Write your configuration file to run the tests every time someone pushes some commits or every time someone creates a pull request
4. Commit and push the changes to GitHub
5. Check the actions tab of your GitHub repo to see if it worked

Matrix configuration

- If your project supports multiple OSes, Python versions, and library version, you might want to run our tests on all the combinations of those



All checks have passed
10 successful checks
Hide all checks

✓ Run all the tests for PRs, with OS/Python matrix / run-tests (ubuntu-latest, 3.9) (pull_request) … Details
✓ Run all the tests for PRs, with OS/Python matrix / run-tests (ubuntu-latest, 3.9) (push) Succes… Details
✓ Run all the tests for PRs, with OS/Python matrix / run-tests (macos-latest, 3.8) (pull_request) … Details
✓ Run all the tests for PRs, with OS/Python matrix / run-tests (macos-latest, 3.8) (push) Succes… Details
✓ Run all the tests for PRs, with OS/Python matrix / run-tests (macos-latest, 3.9) (pull_request) … Details
✓ Run all the tests for PRs, with OS/Python matrix / run-tests (macos-latest, 3.9) (push) Succes… Details

This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request
You can also open this in GitHub Desktop or view command line instructions.

GitHub Actions workflow with matrix config

```
Name: Run all the tests for PRs, with OS/Python matrix

on:
  [push, pull_request]

jobs:
  run-tests:
    runs-on: ${{ matrix.os }}

    strategy:
      matrix:
        os: [ubuntu-latest, macos-latest]
        python-version: [3.8, 3.9]

    steps:
    - uses: actions/checkout@v2
    - name: Set up Python ${{ matrix.python-version }}
      uses: actions/setup-python@v2
      with:
        python-version: ${{ matrix.python-version }}
    - name: Install dependencies
      run:
        python -m pip install pytest numpy
    - name: Test with pytest
      run:
        pytest -sv hands_on/pyanno_votin
```

The strategy/matrix section specifies lists of parameters. The workflow is run for all combinations

GitHub Actions workflow with matrix config

```
Name: Run all the tests for PRs, with OS/Python matrix

on:
  [push, pull_request]

jobs:
  run-tests:
    runs-on: ${{ matrix.os }}

    strategy:
      matrix:
        os: [ubuntu-latest, macos-latest]
        python-version: [3.8, 3.9]

    steps:
    - uses: actions/checkout@v2
    - name: Set up Python ${{ matrix.python-version }}
      uses: actions/setup-python@v2
      with:
        python-version: ${{ matrix.python-version }}
    - name: Install dependencies
      run:
        python -m pip install pytest numpy
    - name: Test with pytest
      run:
        pytest -sv hands_on/pyanno_votin
```

This is how we refer to the matrix parameters in the config file

GitHub Actions reference

- Types of virtual machines available on GitHub Actions: https://docs.github.com/en/actions/using-github-hosted-runners/about-github-hosted-runners#supported-runners-and-hardware-resources
- `setup-python` community action, all available Python flavors and versions: https://github.com/marketplace/actions/setup-python

Hands On!

- Adapt your git actions configuration file `run_test_on_push.yml` to run the testing workflow on Python 3.7, 3.8, 3.9, and on Linux and Windows

Conclusions

- It takes a bit of time to set up and debug a Continuous Integration workflow, but it’s a good investment that can save you a lot of time later on!

Thank you!