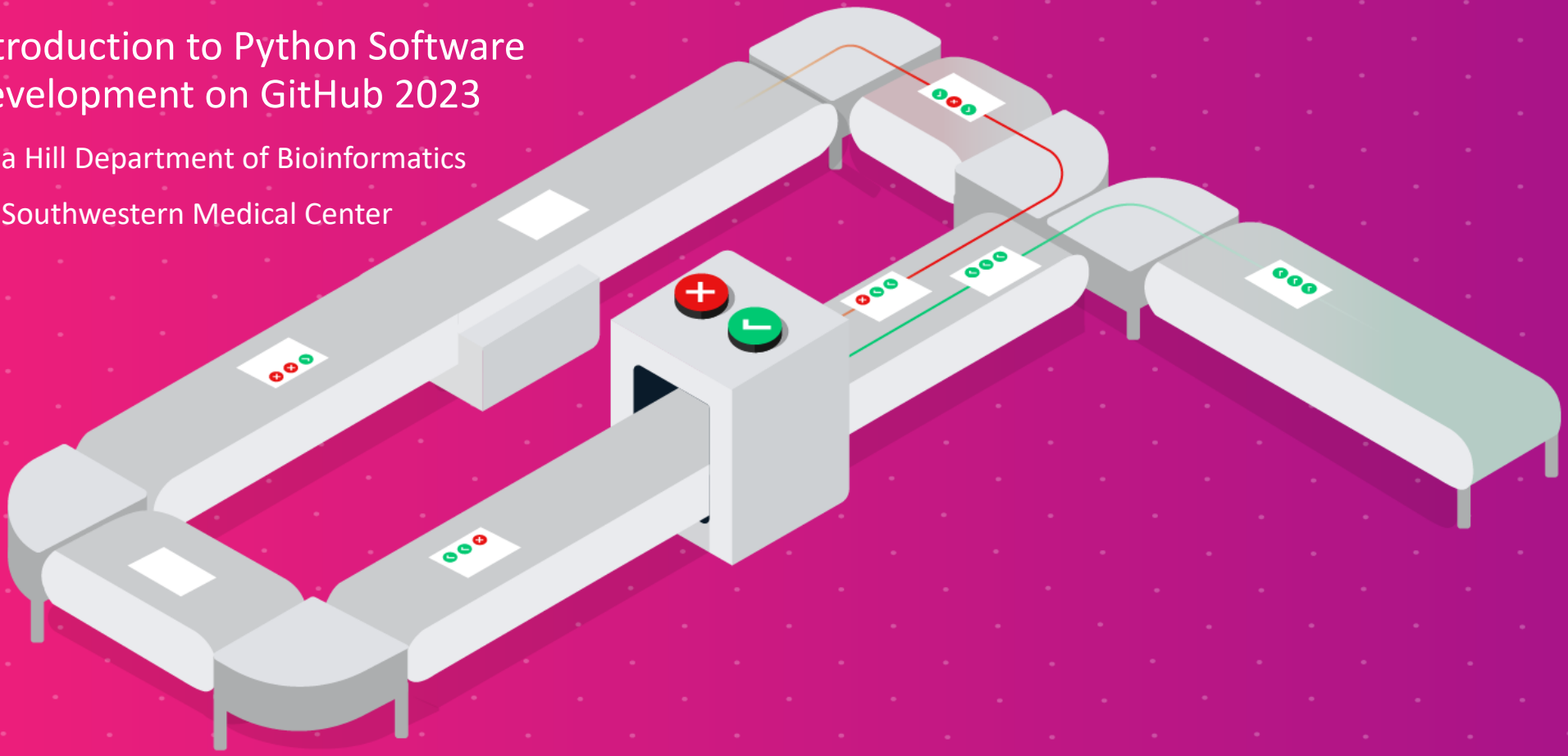
CI/CD Fundamentals

Introduction to Python Software Development on GitHub 2023

Lyda Hill Department of Bioinformatics

UT Southwestern Medical Center



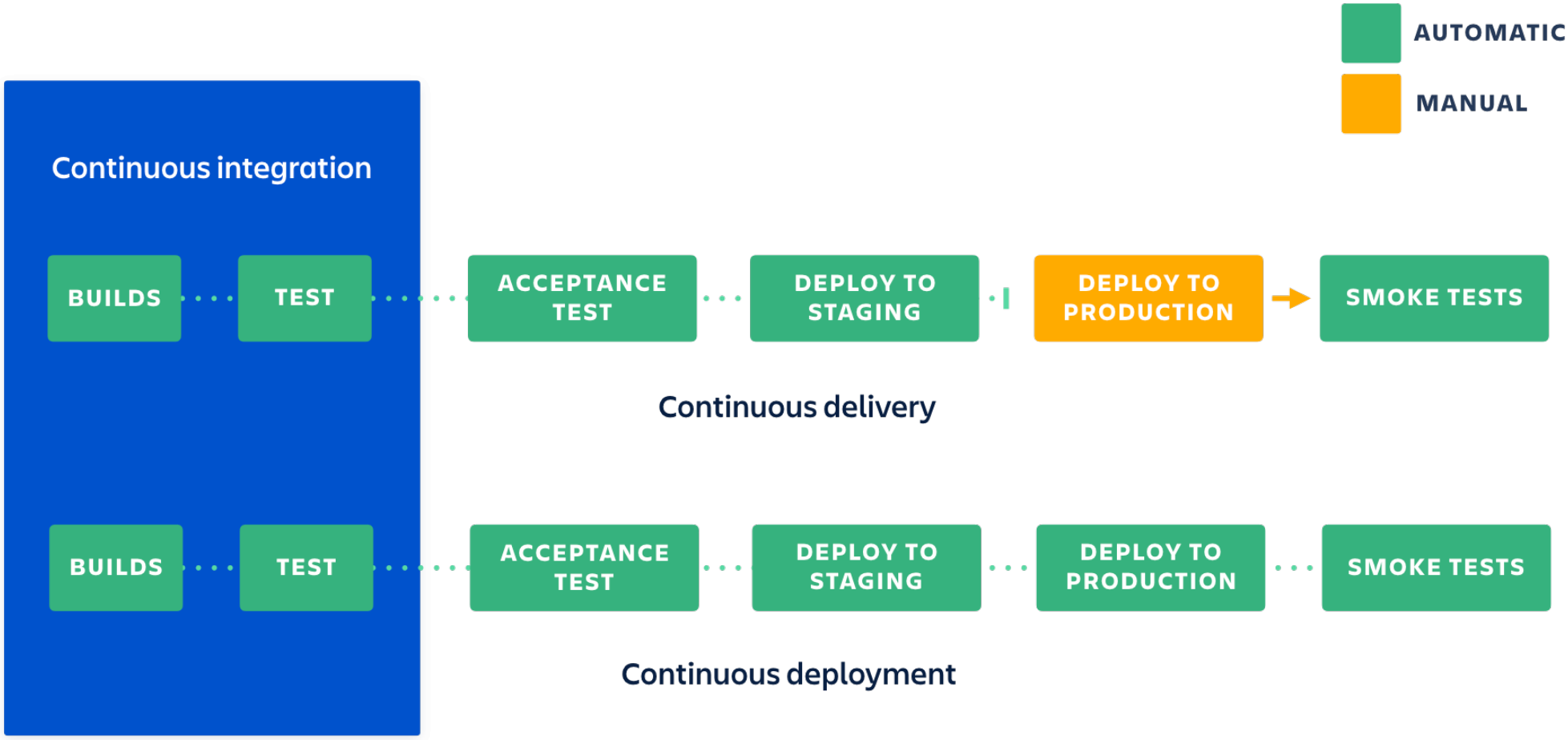
https://about.codecov.io/blog/common-steps-in-a-complete-continuous-integration-workflow/

UT Southwestern Medical Center

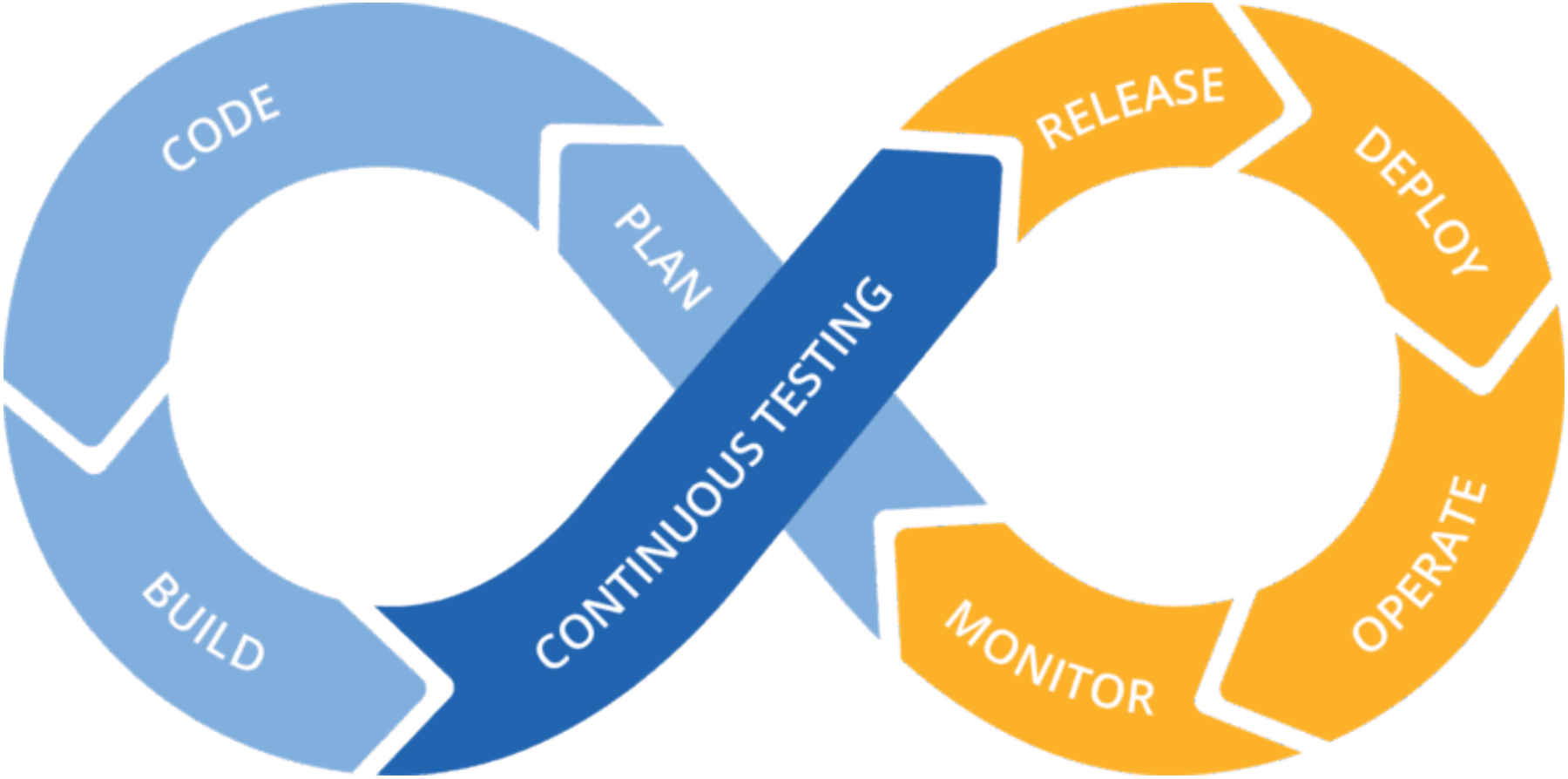
UTSW Department of Bioinformatics

What does CI/CD stand for?

- **C**ontinuous **I**ntegration
- **C**ontinuous **D**elivery
- **C**ontinuous **D**eployment



https://www.atlassian.com/continuous-delivery/principles/continuous-integration-vs-delivery-vs-deployment



https://deploybot.com/blog/continuous-development

Vocabulary

- **Build**

Vocabulary

- **Build** – Convert source code files into a standalone software that anyone can run on their machine.

Vocabulary

- **Build** – Convert source code files into a standalone software that anyone can run on their machine.

- **Test**

Vocabulary

- **Build** – Convert source code files into a standalone software that anyone can run on their machine.

- **Test** – Evaluate and verify software can do what it is supposed to do.

How do we know this function works?

```
1  def square(x):
2      """ Return the square of a number. """
3      return x*x
4
```

We check it with a unit test

```
1  def square(x):
2      """ Return the square of a number. """
3      return x*x
4
5  def test_square():
6      """ Ensure our square() function works. """
7      assert square(2) == 4
8
```

Test subgroups

- **Acceptance test** – A group of unit tests that ensure the software meets specifications e.g. of a contract.

- **Smoke test** – A group of unit tests that act as a sanity check for severe failures. If you run the software, does smoke come out of the computer?

Vocabulary

- **Build** – Convert source code files into a standalone software that anyone can run on their machine.
- **Test** – Evaluate and verify software can do what it is supposed to do.
- **Release**

Vocabulary

- **Build** – Convert source code files into a standalone software that anyone can run on their machine.
- **Test** – Evaluate and verify software can do what it is supposed to do.
- **Release** – A build that is a new or upgraded version of the software.

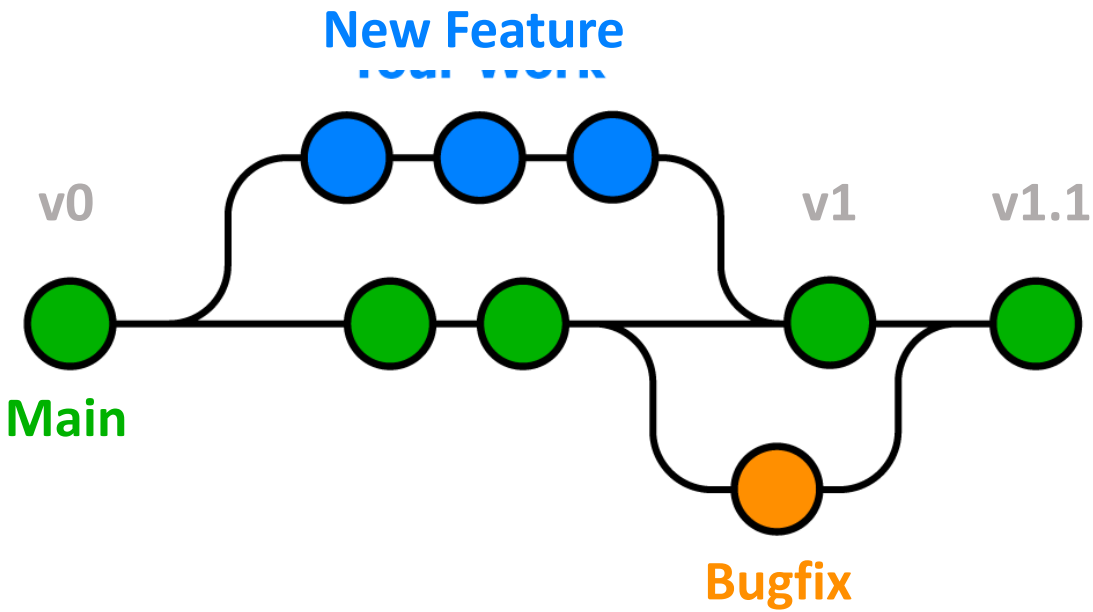
Vocabulary

- **Build** – Convert source code files into a standalone software that anyone can run on their machine.
- **Test** – Evaluate and verify software can do what it is supposed to do.
- **Release** – A build that is a new or upgraded version of the software.
- **Deploy**

Vocabulary

- **Build** – Convert source code files into a standalone software that anyone can run on their machine.
- **Test** – Evaluate and verify software can do what it is supposed to do.
- **Release** – A build that is a new or upgraded version of the software.
- **Deploy** – Make the software available for use.

The goal of CI/CD is to ensure developments do not stray far from the main branch



Why is CI/CD useful?

- Ensures disparate parts of the code base work together throughout development, preventing integration challenges.
- Protects against release of broken software.
- Allows for fast feedback from users and fast fixes from developers.

How do we implement CI/CD in practice?

- **Version control** (`git`)
- **Automatic testing** (`pytest`)
- **Automatic building** (`setuptools`, `pyproject.toml`)
- **Automatic deployment** (`twine`, `PyPI`)

Additional CI/CD tools in the workflow

- **Code quality** (linter such as `ruff`, code formatter such as `black`)
- **Test coverage check** (`codecov`)
- **Documentation** (`sphinx`, `numpydoc`)
- **Security checks** (`CodeQL`)

Local CI/CD workflows

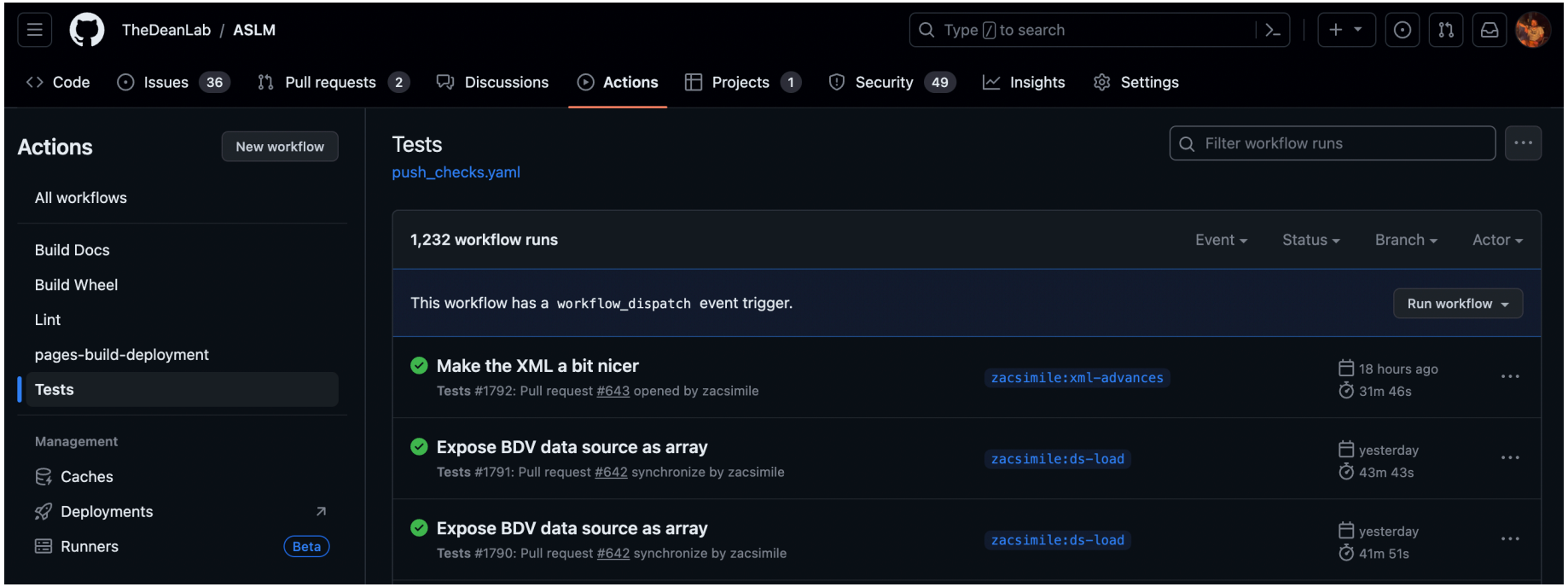
- Can run some tools, such as the linter and code formatter, before pushing code to the repository

- Can automatically run some actions using `pre-commit`

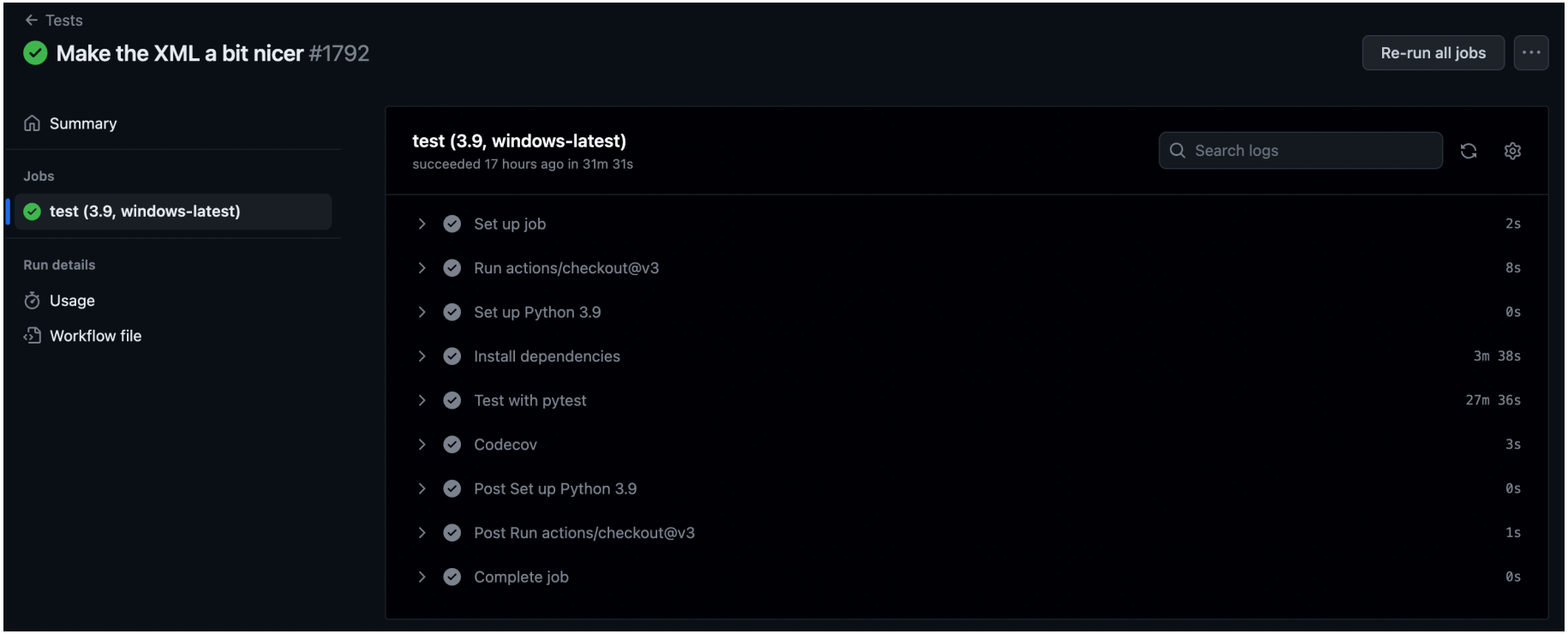
Running the CI/CD workflow

- Need a continuous integration tool
  - Bitbucket (https://bitbucket.org/product/features/pipelines)
  - Jenkins (https://jenkins.io)
  - AWS CodePipeline (https://aws.amazon.com/codepipeline)
  - CircleCI (https://circleci.com)
  - Azure (https://azure.microsoft.com/)
  - Gitlab (https://about.gitlab.com/)
  - GitHub (https://github.com/)
  - Etc.
- These tools use a YAML file (or similar) to describe a series of actions that make up a workflow.

GitHub Actions Dashboard



GitHub Actions Workflow Example



GitHub Actions Workflow Example