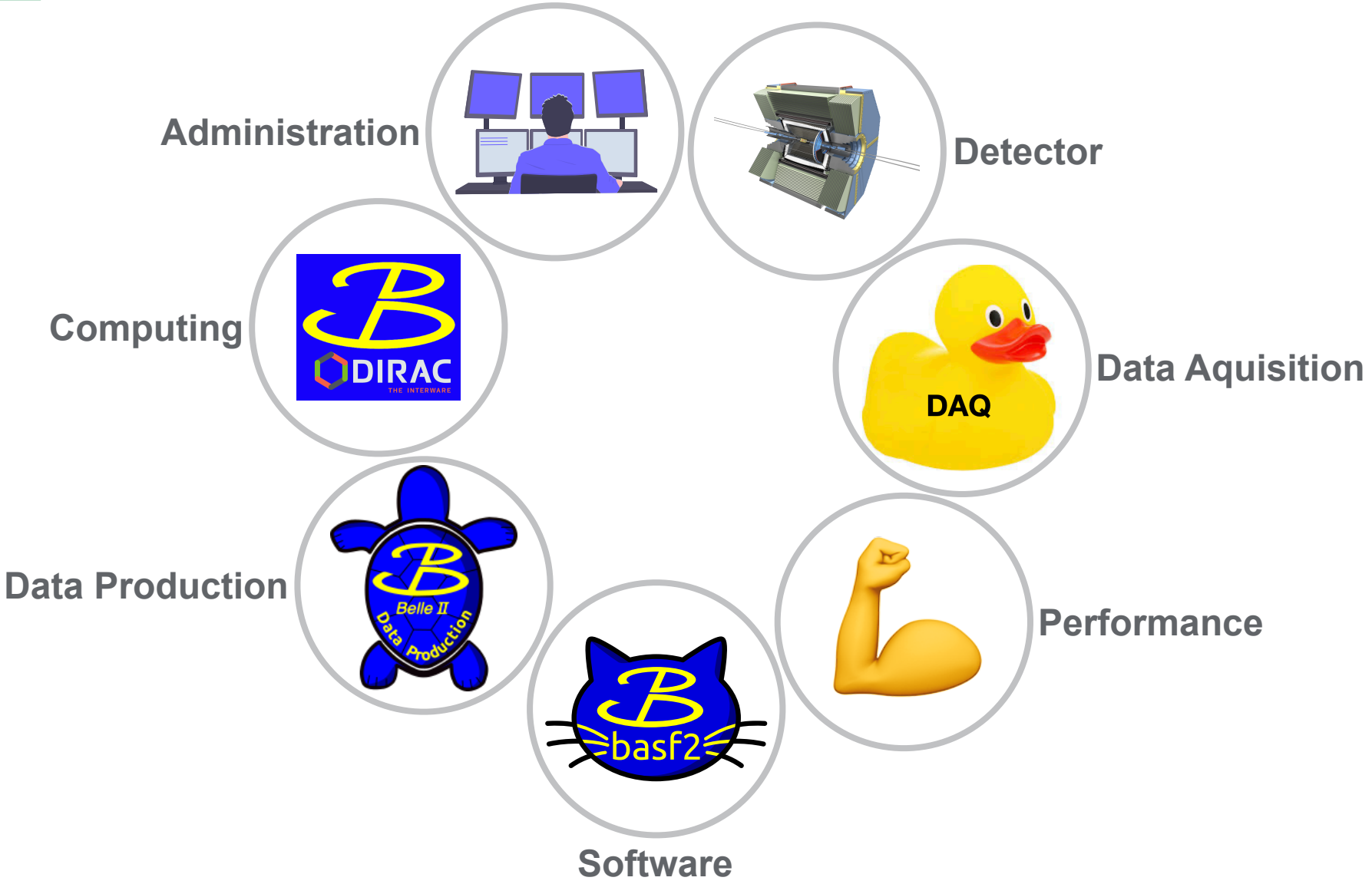
LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN
FACULTY OF PHYSICS
EXPERIMENTAL FLAVOR PHYSICS

Software Development at Belle II

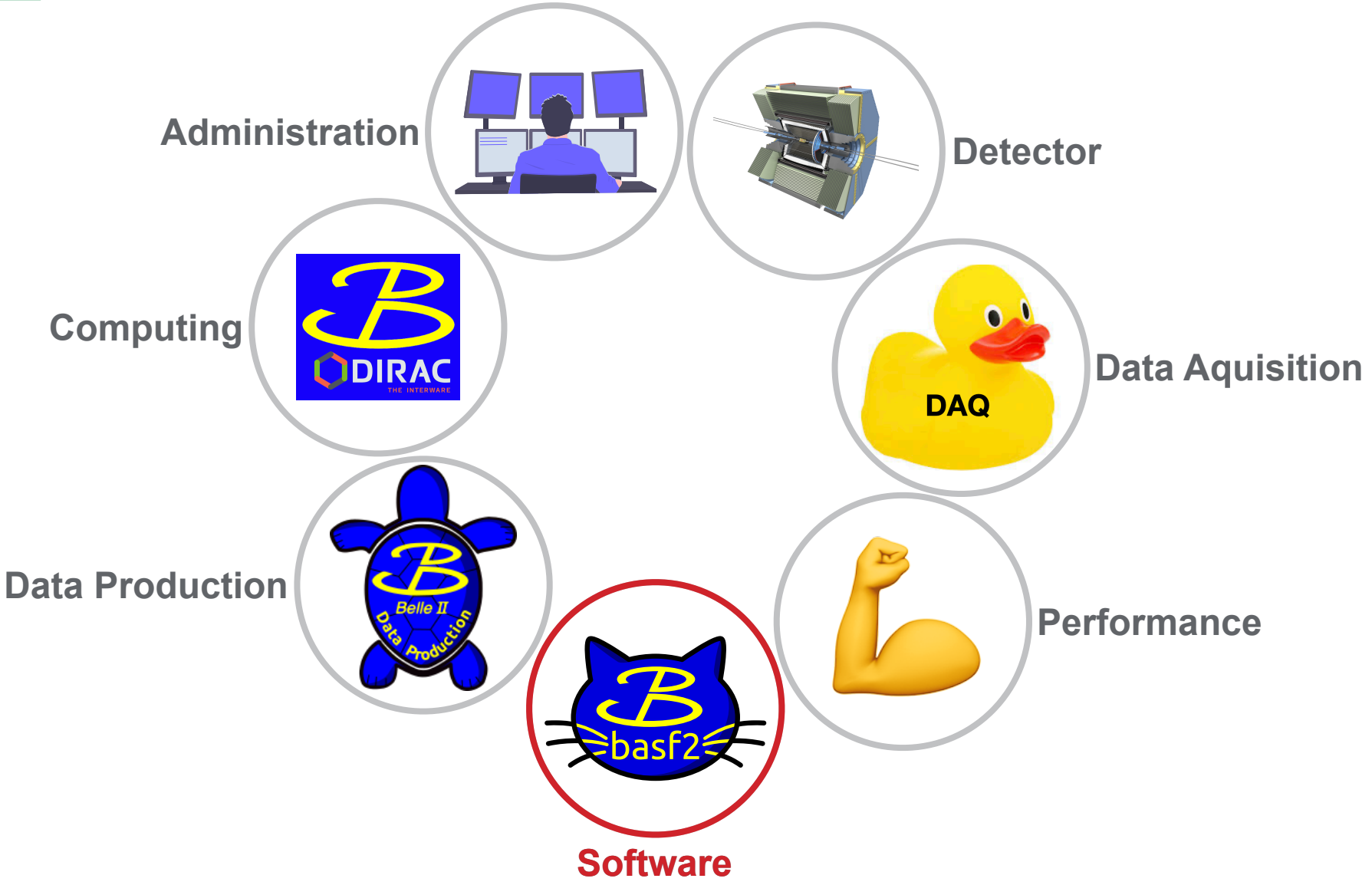
Belle II Academy 2026
04.05.2026

Alexander Heidelbach, Giacomo De Pietro, Thomas Kuhr

Where do we have Software in Belle II?

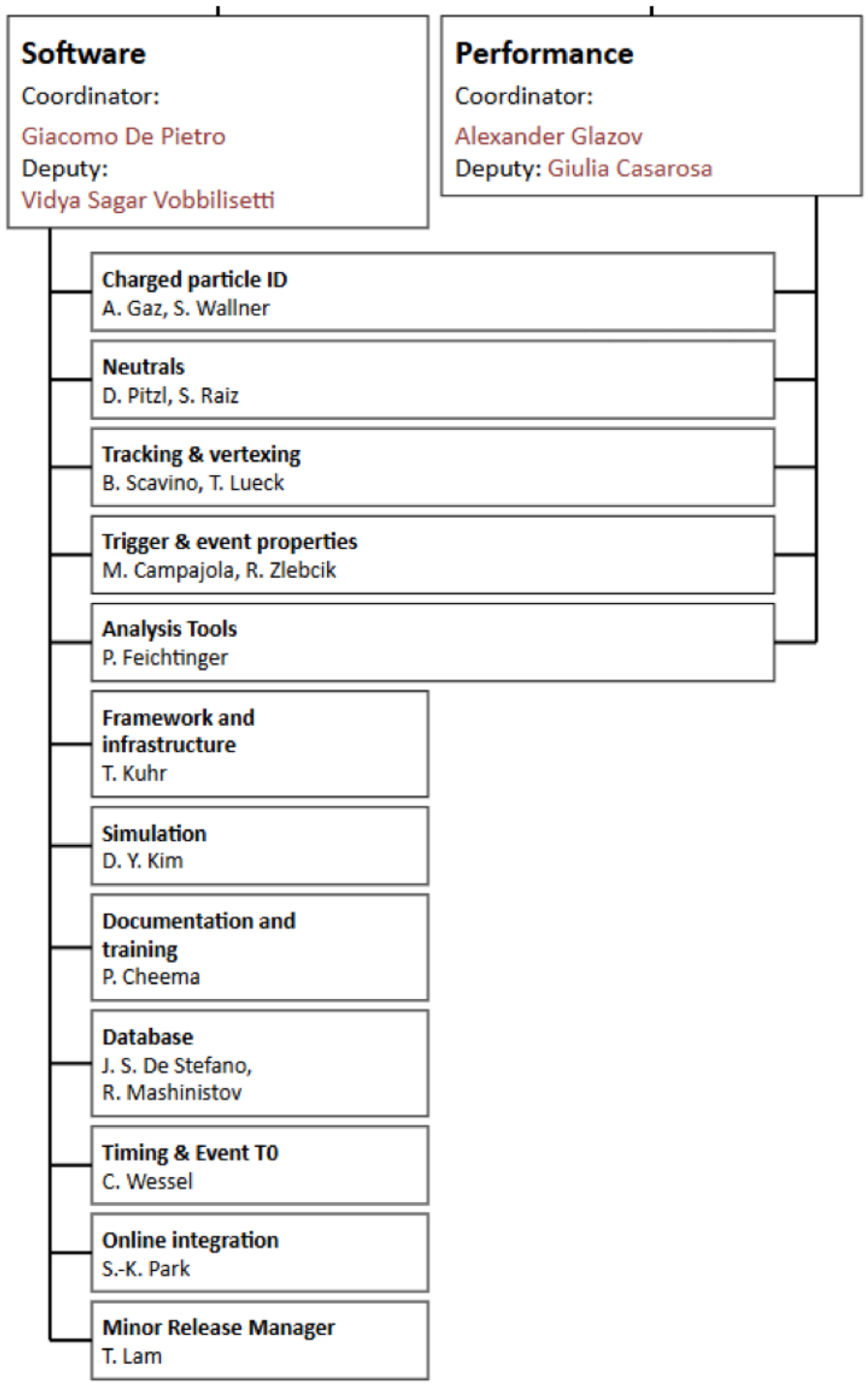


Where do we have Software in Belle II?



The Software Team

- Few **shared** groups between **software** and **performance**
- Few additional “software-only” groups
  - Mostly “one-man-band” than actual groups…
- basf2 librarians are also considered part of the software group
  - 37 packages: each package has between 1-3 librarians



The Software Packages

Belle II / Software

Software

Subgroups and projects | Shared projects | Shared groups | Inactive

Search (3 character minimum) | Name

C Conditions Database	B belle_legacy	S SysVar
E examples-data-creation	D display	T tools
M MVA training scripts	D docker-images	T topoana
Tracking	E examples-data	V validation-data
T Training	E externals	V versioning
b2luigi Owner b2luigi - bringing batch 2 luigi	F FastBDT	
B basf2	L LightReleaseValidation	
B belle2style	S starterkit-data	

Main Focus

- basf2
- externals
- tools
- b2luigi
- buildbot
- SysVar
- FastBDT
- …

basf2 Overview

Code
Documentation

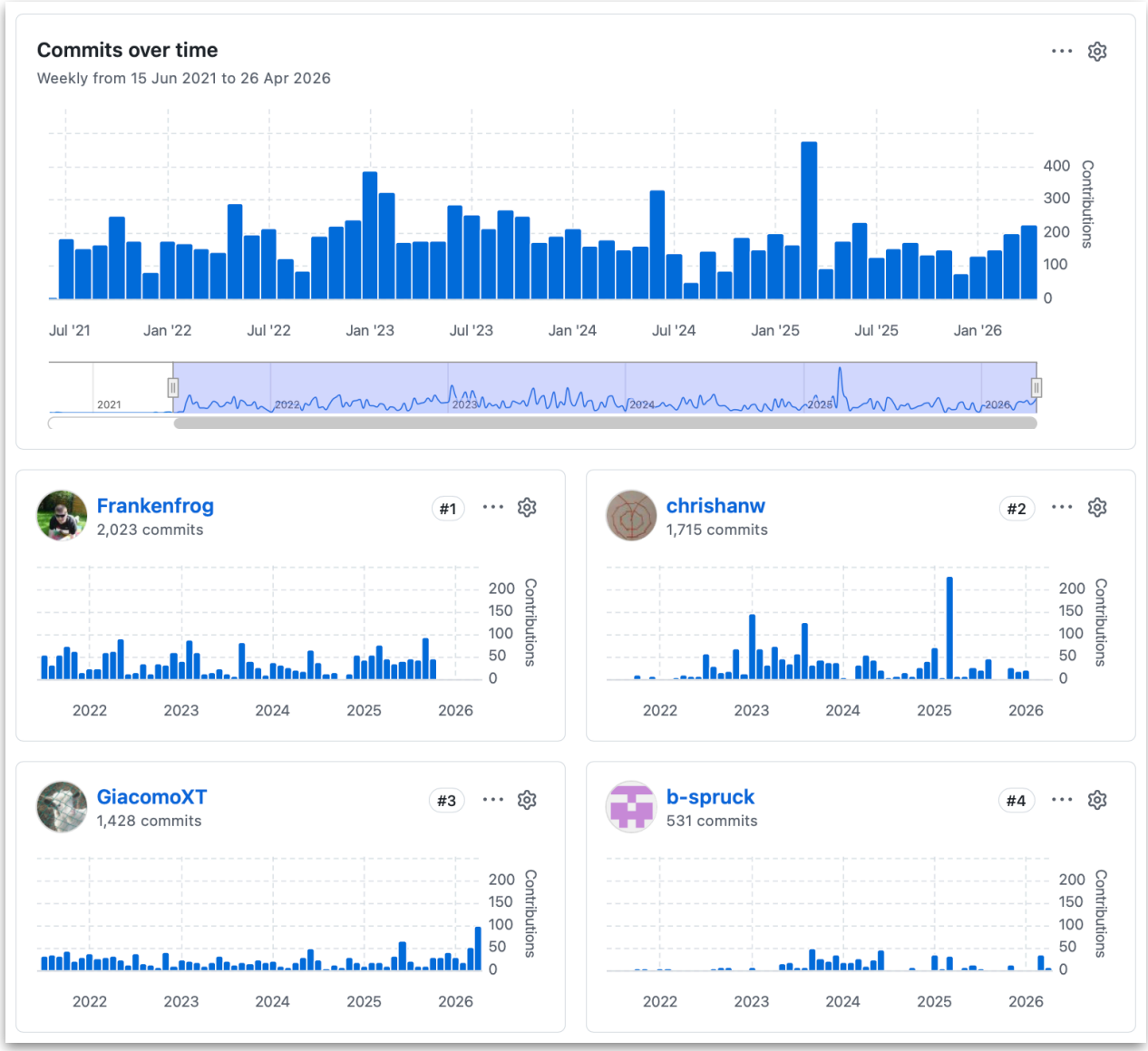
The largest software project in the collaboration

- Development started more than 10 years ago
- 37 packages:
  - **Core**: `framework`, `site_scons`
  - **Subdetector**: `arich`, `cdc`, `ecl`, `klm`, `pxd`, `svd`, `top`, `vxd`
  - **Data Taking**: `daq`, `hlt`, `rawdata`, `trg`
  - **Data Quality**: `alignment`, `calibration`, `dqm`, `validation`
  - **Data Storage**: `mdst`, `skim`
  - **MC**: `decfiles`, `generators`, `geometry`, `simulation`, `structure`
  - **Background**: `background`, `beast`, `ir`
  - **Offline Analysis**: `analysis`, `b2bii`, `mva`
  - **Documentation**: `display`, `masterclass`, `online_book`
  - **Reconstruction**: `genfit2`, `reconstruction`, `tracking`

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

basf2 in Numbers

- ~58k commits up to July 2021
- ~20k commits since July 2021
- Many lines of code…
  - ~54% of code written in C++
  - ~27% in Fortran
  - ~16% in Python
  - rest in C, RST, etc.
- Largely unit-tested
  - 1011 “Google” tests
  - 79 framework tests
  - 305 non-framework tests



basf2 Releases

Light Release

- It contains only a subset of the packages
- One tag every ~2 months

Full Release

- It contains all the packages
- One major tag per year

Upgrade Release

- Tag of the upgrade branch
- Used for developing and testing new features/ geometries for the Belle II upgrade

tools

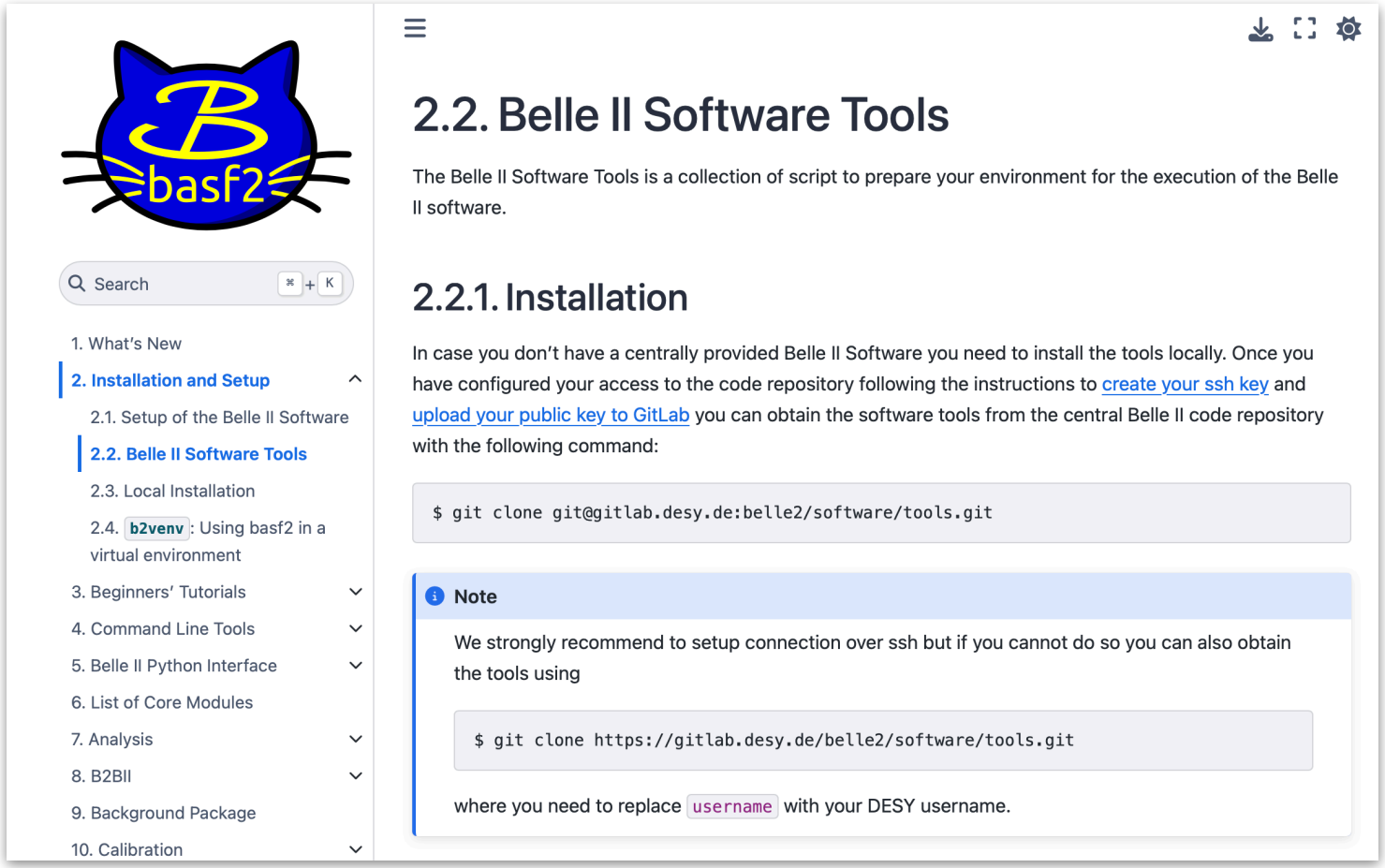
Code Documentation

Set of basic tools for setting up basf2

- Analysis
- Development

Some Tools

- `b2setup`
- `b2install-prepare`
- `b2execute`
- `b2venv`
- …



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

externals

Code

The code to download and compile the "external" packages on which basf2 depends

- We currently have 78 packages we compile ourselves:
  - compilers (gcc, clang)
  - common HEP software (ROOT, Geant4, CLHEP, …)
  - generators (EvtGen, PYTHIA, PHOTOS, MadGraph, WHIZARD, Herwig, etc.)
  - many other libraries (curl, zlib, cmake, Git, Eigen, Boost, XRootD, etc.)
  - Python
- We also include 268 Python packages
  - Many packages are included by "popular demand" from users

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

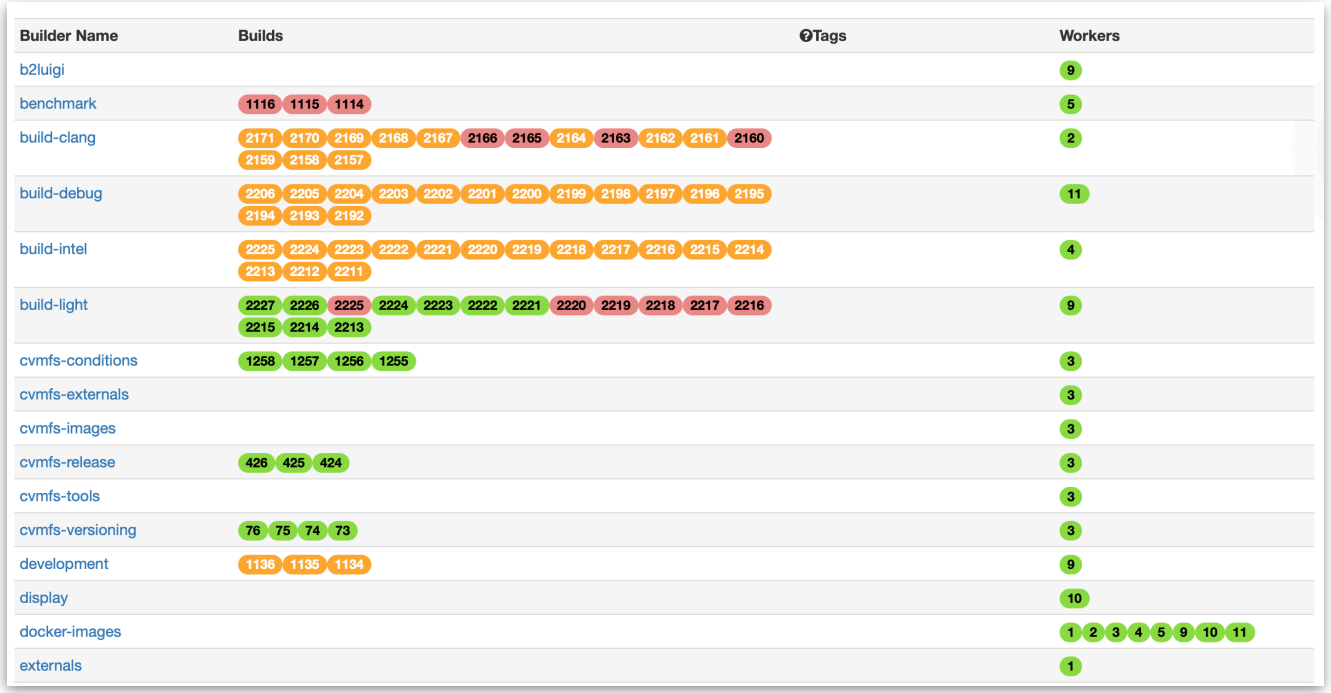
buildbot

Code

buildbot is an open-source software for building CI/CD pipelines

- build of basf2 releases
- deployment on cvmfs of basf2 releases, externals builds and updated tools
- update of globaltags cache on CVMFS
- build and deployment of many documentations
- trigger of docker images build and their deployment on CVMFS

- 4 main workers + KEKCC + LMU



Builder Name	Builds	Tags	Workers
b2luigi			9
benchmark	1116 1115 1114		5
build-clang	2171 2170 2169 2168 2167 2166 2165 2164 2163 2162 2161 2160 2159 2158 2157		2
build-debug	2206 2205 2204 2203 2202 2201 2200 2199 2198 2197 2196 2195 2194 2193 2192		11
build-intel	2225 2224 2223 2222 2221 2220 2219 2218 2217 2216 2215 2214 2213 2212 2211		4
build-light	2227 2226 2225 2224 2223 2222 2221 2220 2219 2218 2217 2216 2215 2214 2213		9
cvmfs-conditions	1258 1257 1256 1255		3
cvmfs-externals			3
cvmfs-images			3
cvmfs-release	426 425 424		3
cvmfs-tools			3
cvmfs-versioning	76 75 74 73		3
development	1136 1135 1134		9
display			10
docker-images			1 2 3 4 5 9 10 11
externals			1

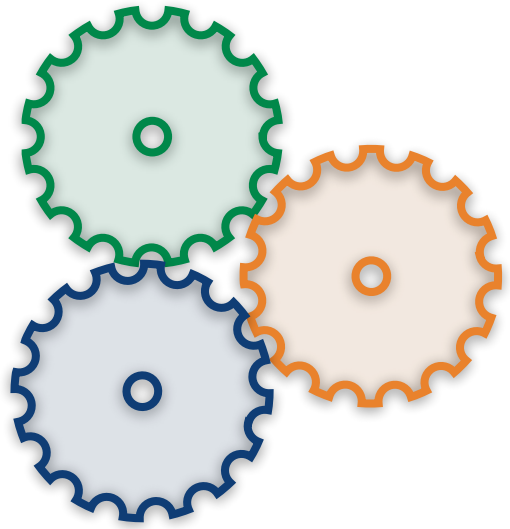
b2luigi

[Code Documentation]

- Helper package constructed around the popular workflow manager luigi
  - Main purpose is to ease building the task trees in everyday scenarios
- Since November 2023, maintenance has been in the hands of Belle II

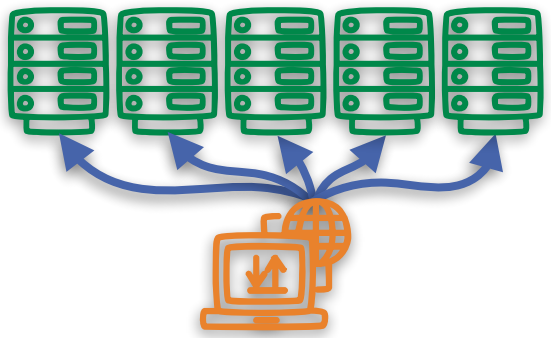
Workflow I/O

```
my_project=foo/
├─parameter=1/
│  ├─other_parameter=a/
│  └─other_parameter=b/
├─parameter=2/
│  ├─other_parameter=a/
│  └─other_parameter=b/
└─parameter=3/
```



Workflow Steering

Distribute Workflow



User Interface

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

SysVar

Code Documentation

Python package that streamlines the treatment of systematic uncertainties

- Corrections from the performance groups will be provided in a format that SysVar can handle
  - SysVar will be the recommended tool for applying the necessary offline corrections!

SysVar: A tool enhancing consistency in the treatment of systematics

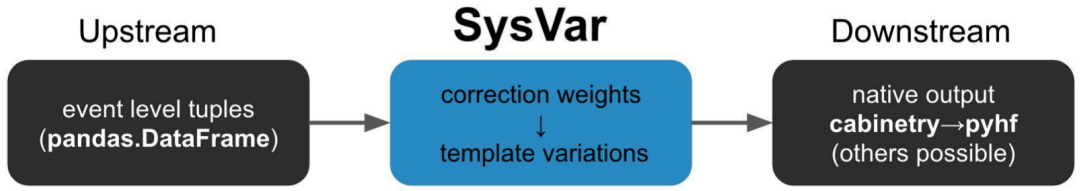
SysVar is a python based tool that allows to:

Anyone who corrects MC offline

Useful for

Anyone who runs template fits

1. Apply Data/MC corrections to a DataFrame
2. Generate Variations of Data/MC Corrections
3. Histogram MC to build templates and template variations for a non-parametric fit.
4. Diagonalize a cov matrix (channels x templates x bins) to produce eigenvariations and save them to implement Correlated Shape Variations
5. Identify the number of necessary eigendirections that the user should consider for an accurate analysis (different criteria are available)



SysVar sits between event-level data and the statistical model builder, converting correlated weight systematics into template variations

7

Talk

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

FastBDT

Code

Stochastic gradient-boosted decision trees for multivariate classification, usable standalone and via Python interface

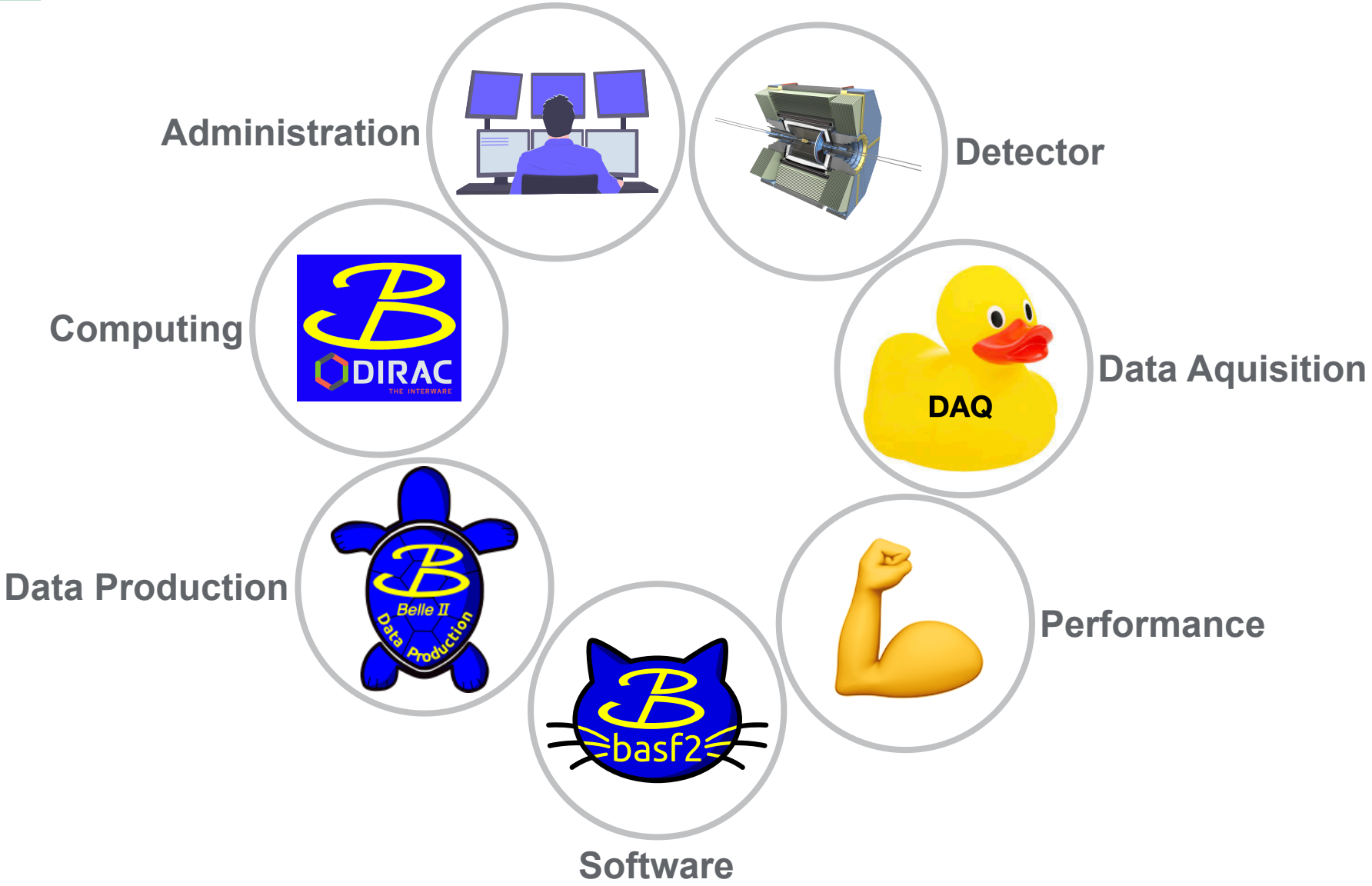
- Last summer, Belle II forked the original FastBDT repository from Thomas Keck and started maintaining its own version
- No new features or functional modifications are currently planned

```
for i in range(len(mean)):
    for j in range(i+1, len(mean)):
        cov[j][i] = cov[i][j]

N_train, N_test = 10000, 10000
data = np.random.multivariate_normal(mean, cov, N_train + N_test)
X_train, y_train = data[:N_train, 1:], data[:N_train, 0] > 0
X_test, y_test = data[N_train:, 1:], data[N_train:, 0] > 0

# Train FastBDT using its PythonInterface, which is based on the SKLearn classifiers
clf = FastBDT.Classifier()
clf.fit(X=X_train, y=y_train)
p = clf.predict(X_test)
global_auc = FastBDT.calculate_roc_auc(p, y_test)
print("Global AUC", global_auc)
```


HELP
WANTED


HELP
WANTED
NEEDED

Where do we ~~have~~ need Software in Belle II?



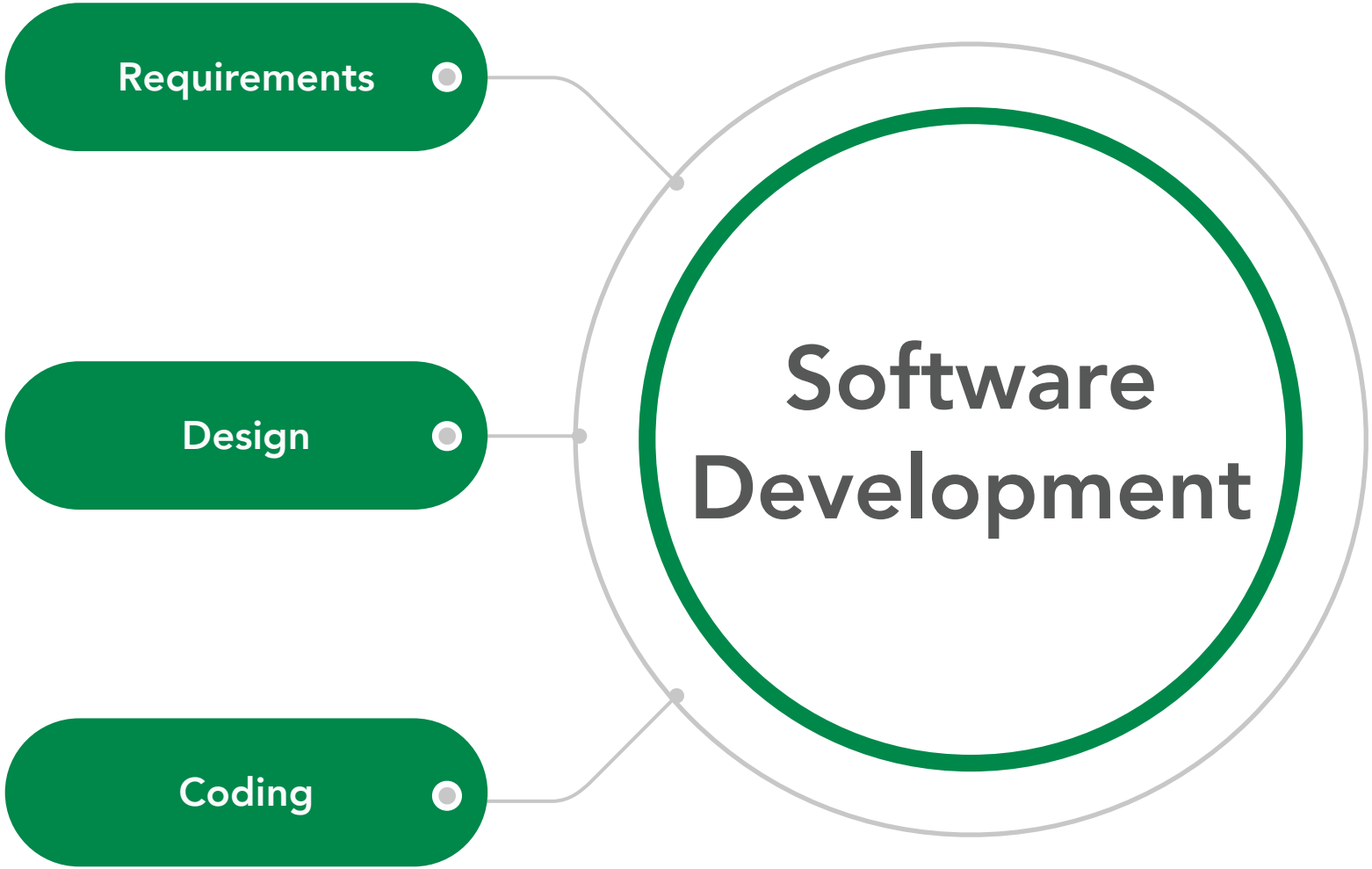
What is Software Development?

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

What is Software Development?

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

What is Software Development?



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

What is Software Development?

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

What is Software Development?

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

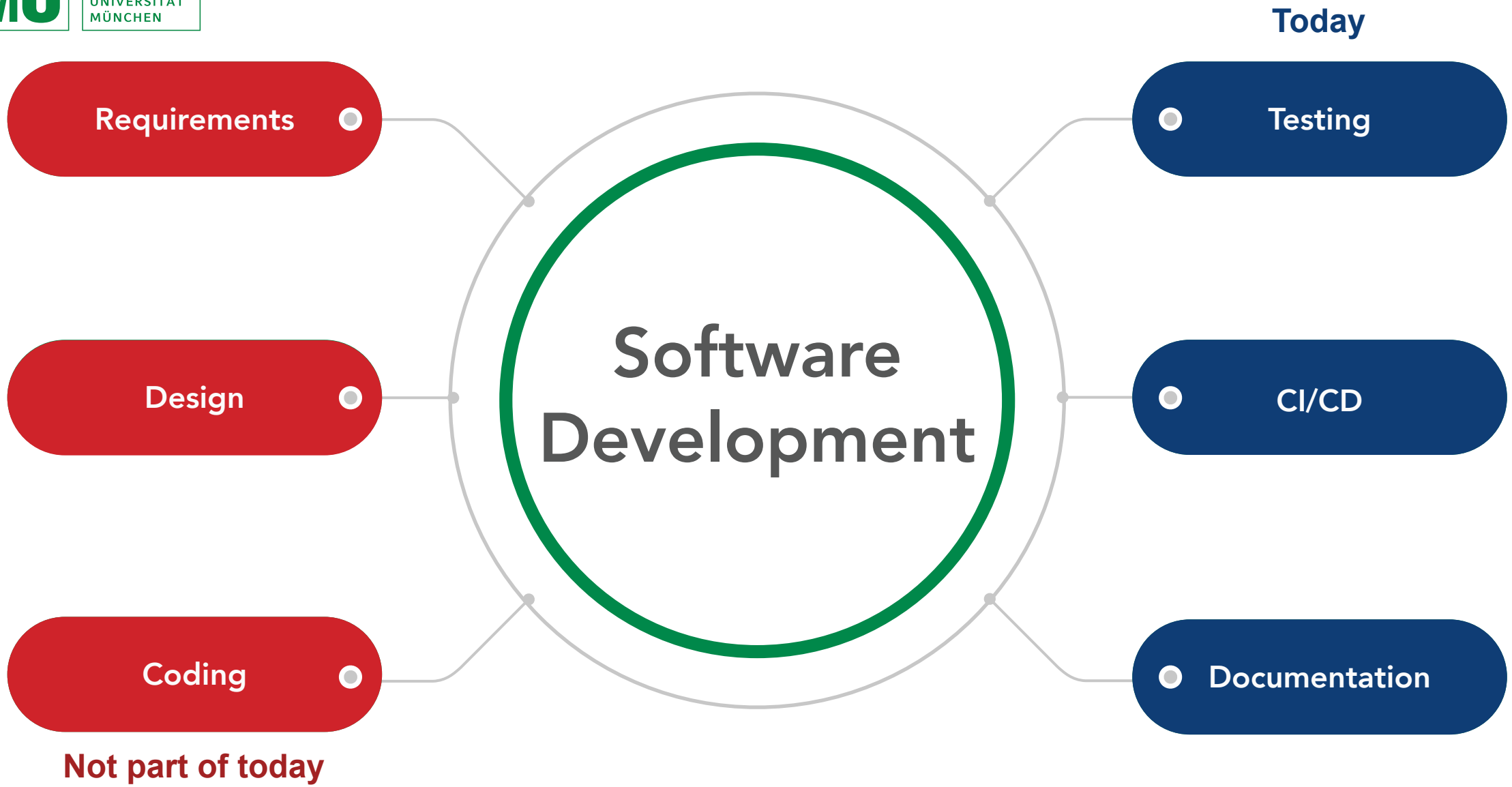
What is Software Development?

What is Software Development?

Not part of today

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

What is Software Development?

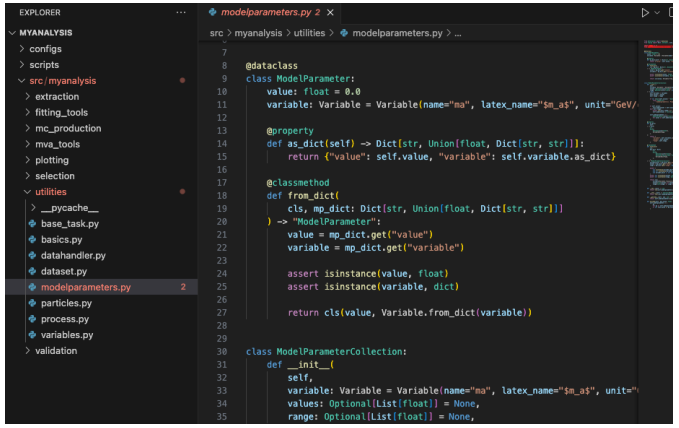


LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

What is Software Development?

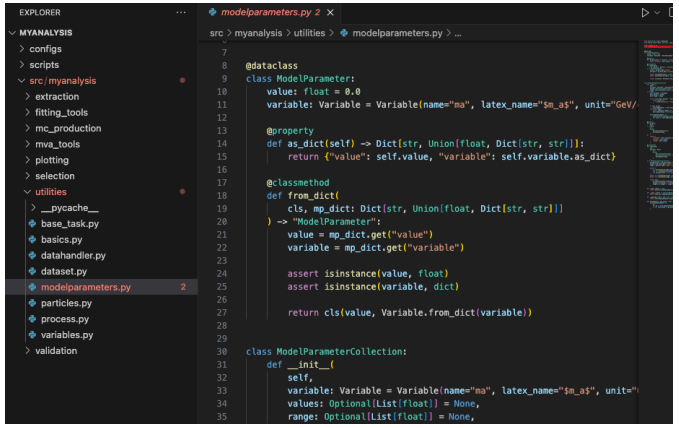
LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Does this sound familiar?

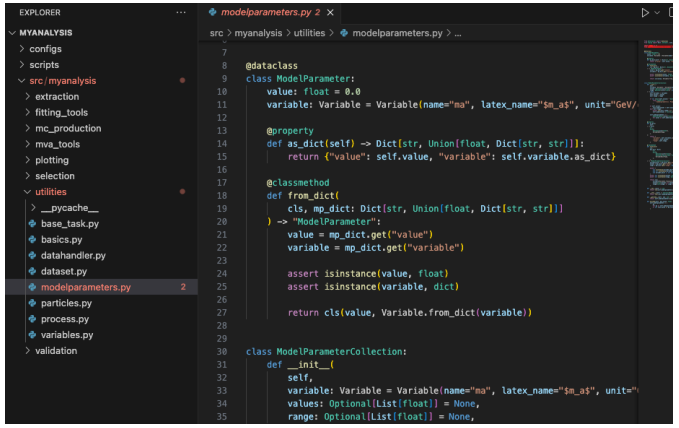


LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Does this sound familiar?



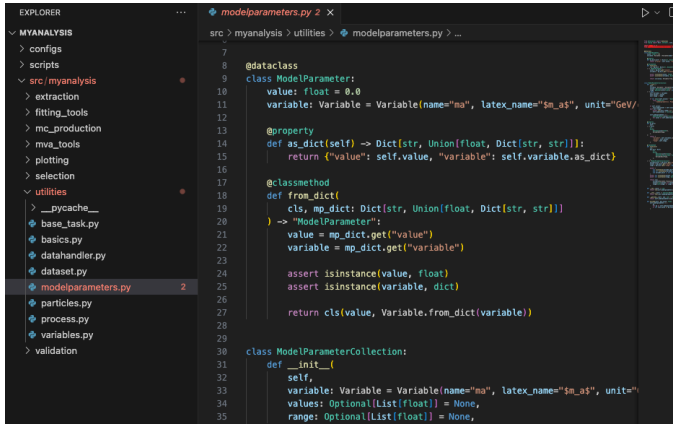
Does this sound familiar?



There is a new parameter in the MC files. Can you propagate it through your workflow?

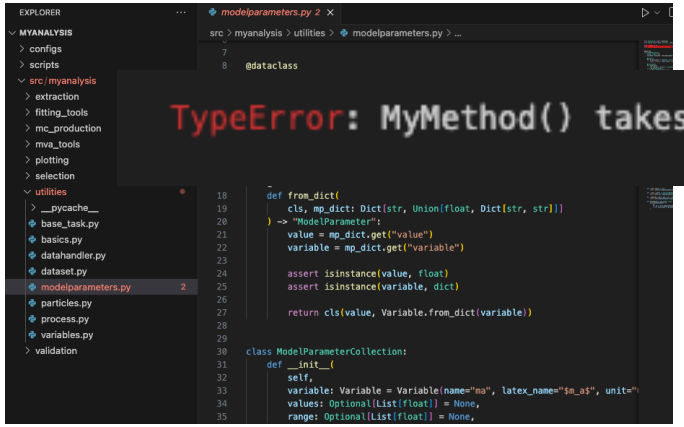
LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Does this sound familiar?



Sure! No problem. I just have to change the output of one method!

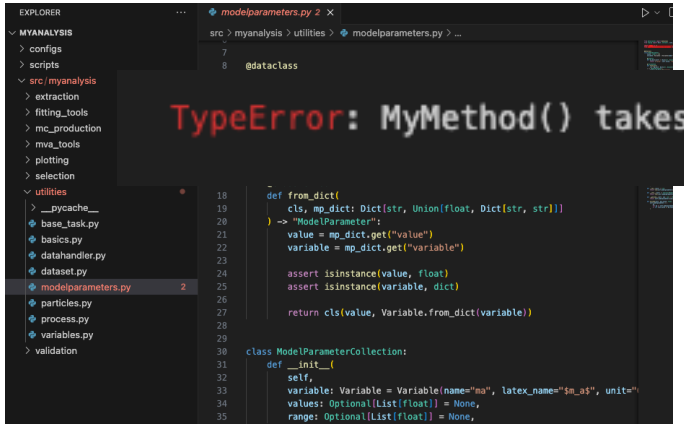
Does this sound familiar?




```
TypeError: MyMethod() takes 2 positional arguments but 3 were given
```

Sure! No problem. I just have to change the output of one method!

Does this sound familiar?




```
TypeError: MyMethod() takes 2 positional arguments but 3 were given
```

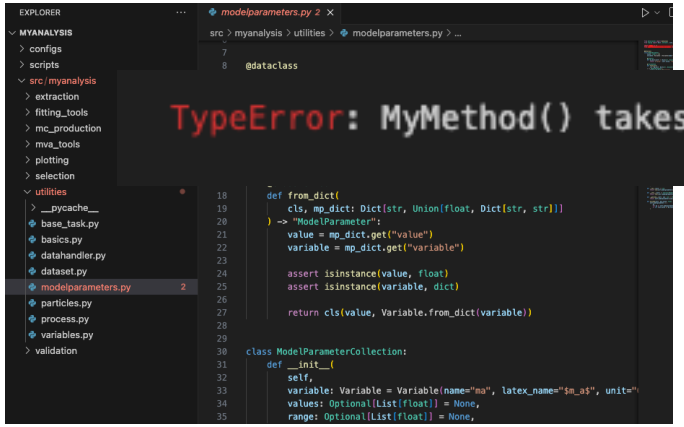
Sure! No problem. I just have to change th

```
IndexError: list index out of range
```

one method.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Does this sound familiar?




```
TypeError: MyMethod() takes 2 positional arguments but 3 were given
```

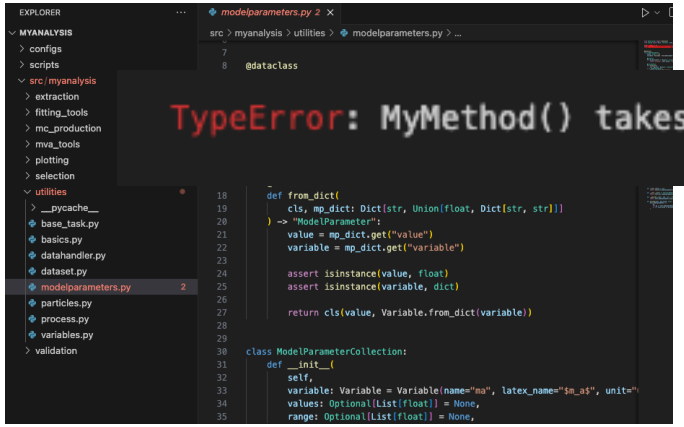
Sure! No problem. I just have to change th one method.

```
IndexError: list index out of range
```

```
TypeError: object of type 'float' has no len()
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Does this sound familiar?

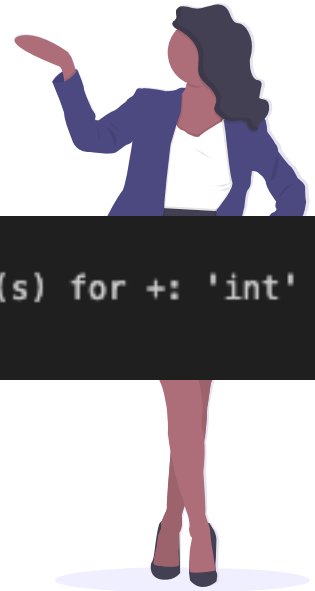



```
TypeError: MyMethod() takes 2 positional arguments but 3 were given
```

Sure! No problem. I just have to change th one method.

```
IndexError: list index out of range
```

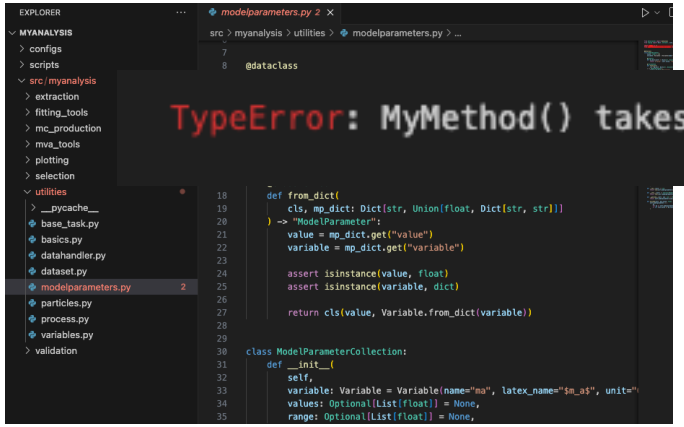
```
TypeError: object of type 'float' has no len()
```



```
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Does this sound familiar?



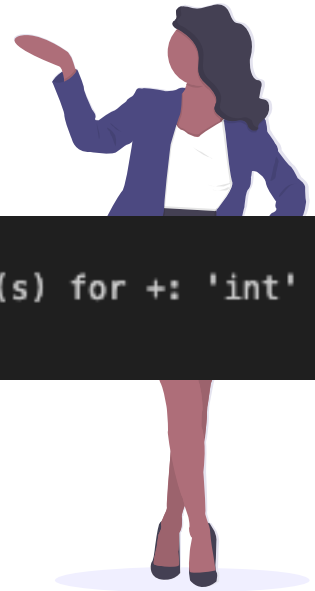

```
TypeError: MyMethod() takes 2 positional arguments but 3 were given
```

Sure! No problem. I just have to change th one method.

```
IndexError: list index out of range
```

```
TypeError: object of type 'float' has no len()
```

```
ZeroDivisionError: division by zero
```



```
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

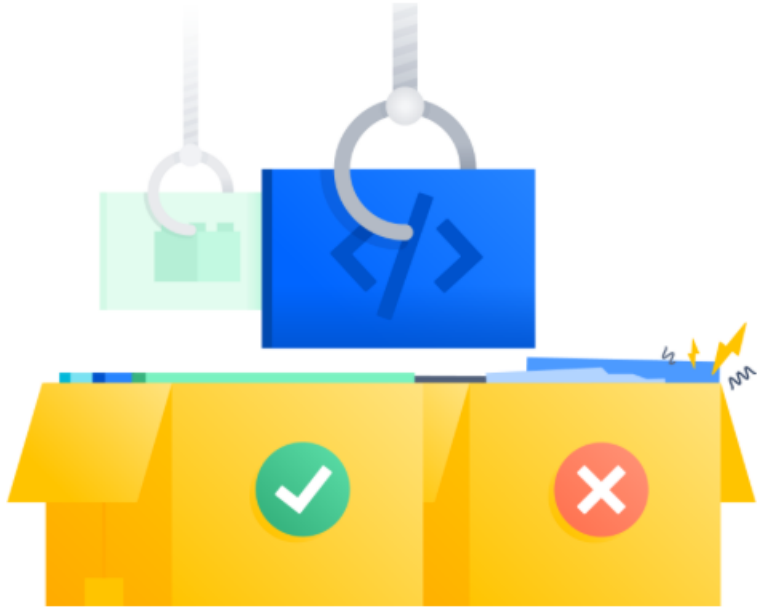
Testing is Common Practice

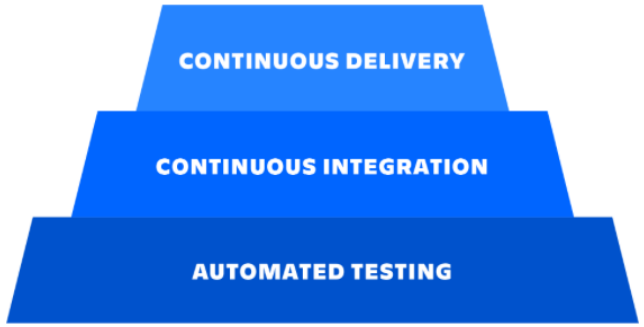
Testing code is as common as writing the code itself

- Before ~2000, companies had dedicated teams (much like Q&A) to make sure things worked manually

Automation in Industry

- Test-Driven Development
- Continuous Integration and Continuous Testing
- Artificial Intelligence in Testing





Testing is Common Practice

Testing code is as common as writing the code itself

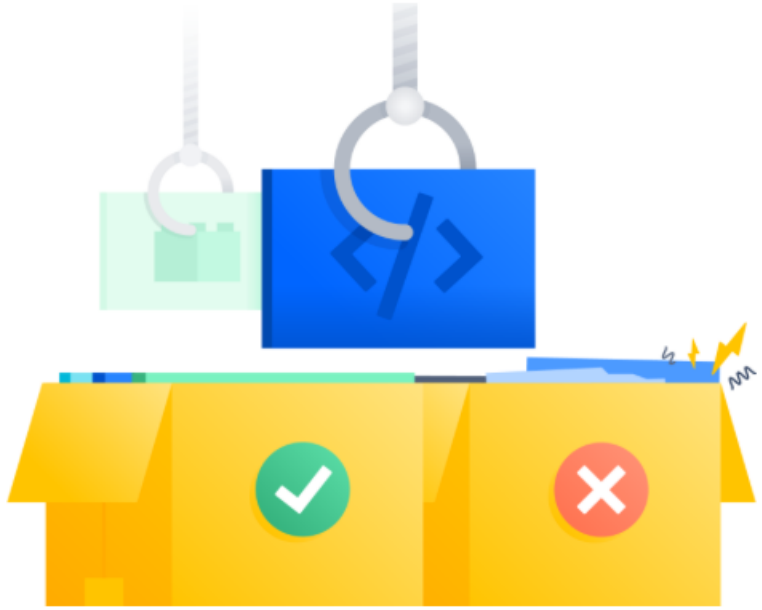
- Before ~2000, companies had dedicated teams (much like Q&A) to make sure things worked manually

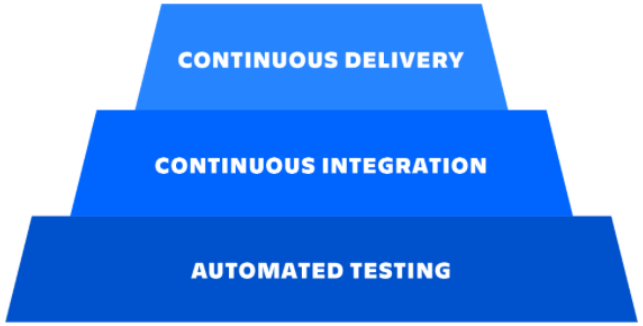
Automation in Industry

- Test-Driven Development
- Continuous Integration and Continuous Testing
- Artificial Intelligence in Testing

Belle II

- Over time, almost every core software package is at least functionally tested





LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Benefits of Testing Code

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Benefits of Testing Code

Time is Money!

- Complex code that grows over time
- Several weeks or months between connected projects
- Think of tests as an investment:
  - Each minute spent writing tests can save hours of frustration

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Benefits of Testing Code

Time is Money!

- Complex code that grows over time
- Several weeks or months between connected projects
- Think of tests as an investment:
  - Each minute spent writing tests can save hours of frustration

Collaboration

- Several people are working on the same project
- Multiple branches add to the main
  - There is a high chance of adding a commit that breaks everything

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Benefits of Testing Code

Time is Money!

- Complex code that grows over time
- Several weeks or months between connected projects
- Think of tests as an investment:
  - Each minute spent writing tests can save hours of frustration

Collaboration

- Several people are working on the same project
- Multiple branches add to the main
  - There is a high chance of adding a commit that breaks everything

Experience

- Learning to write tests is a valuable skill

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Benefits of Testing Code

Time is Money!

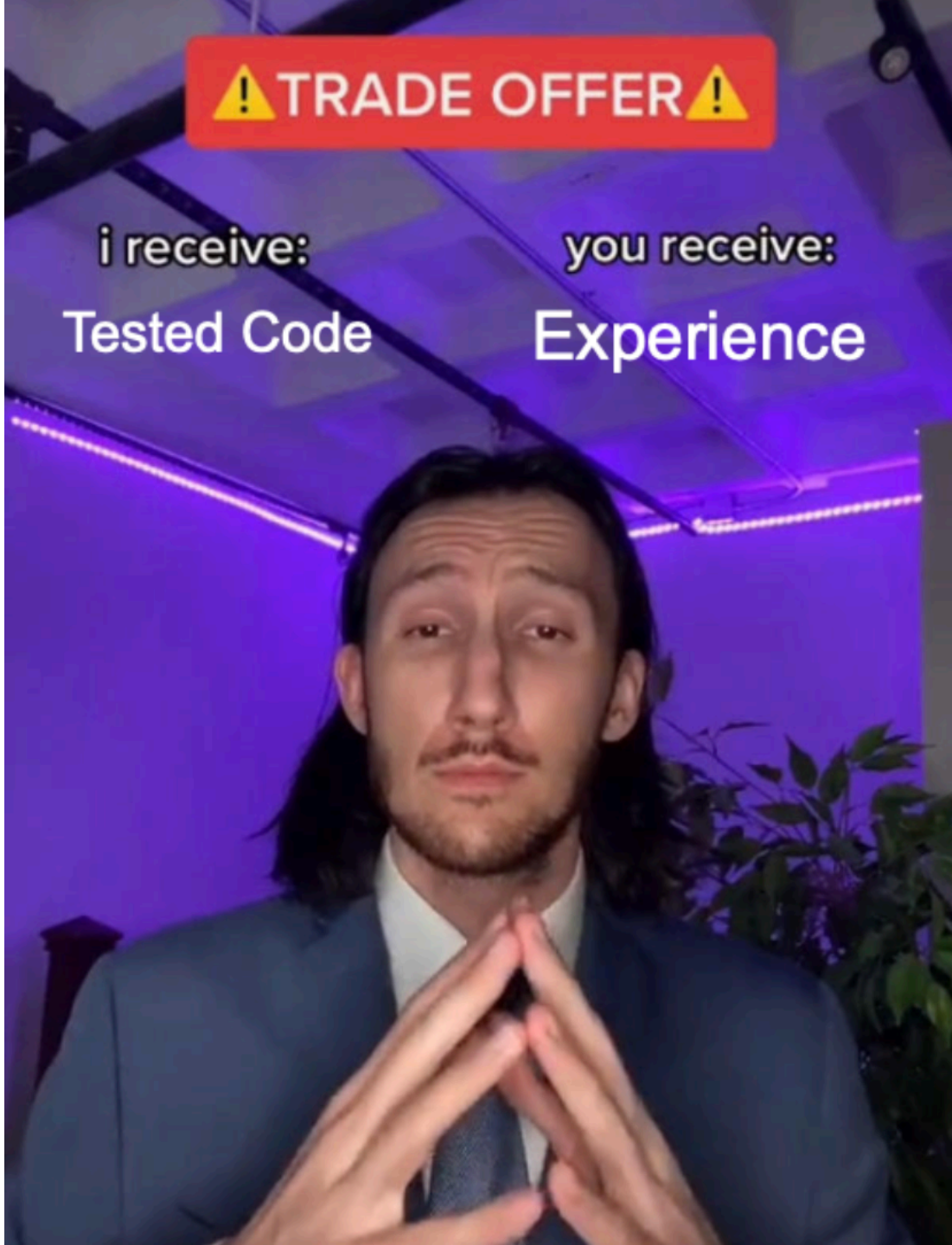
- Complex code that grows over time
- Several weeks or months between connected projects
- Think of tests as an investment:
  - Each minute spent writing tests can save hours of frustration

Collaboration

- Several people are working on the same project
- Multiple branches add to the main
  - There is a high chance of adding a commit that breaks everything

Experience

- Learning to write tests is a valuable skill



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

What is a “Test”?

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

What is a “Test”?

“Software testing is the act of checking whether software meets its intended objectives and satisfies expectations.”

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

What is a “Test”?

“Software testing is the act of checking whether software meets its intended objectives and satisfies expectations.”

“Does the software do what it is supposed to do and what it needs to do?”

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

What is a “Test”?

“Software testing is the act of checking whether software meets its intended objectives and satisfies expectations.”

“Does the software do what it is supposed to do and what it needs to do?”

Tests happen…

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

What is a “Test”?

“Software testing is the act of checking whether software meets its intended objectives and satisfies expectations.”

“Does the software do what it is supposed to do and what it needs to do?”

Tests happen…

- … automatic!

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

What is a “Test”?

“Software testing is the act of checking whether software meets its intended objectives and satisfies expectations.”

“Does the software do what it is supposed to do and what it needs to do?”

Tests happen…

- … automatic!
- … on different levels!
  - Unit, Integration, System

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

What is a “Test”?

“Software testing is the act of checking whether software meets its intended objectives and satisfies expectations.”

“Does the software do what it is supposed to do and what it needs to do?”

Tests happen…

- … automatic!
- … on different levels!
  - Unit, Integration, System
- … in different formats!
  - Static, dynamic, passive,…

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Hello World!

```python
import unittest

def add(a, b):
    return a + b

class TestMath(unittest.TestCase):
    def test_add(self):
        self.assertEqual(add(2, 3), 5)

if __name__ == "__main__":
    unittest.main()
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Hello World!

```python
import unittest

def add(a, b):
    return a + b

class TestMath(unittest.TestCase):
    def test_add(self):
        self.assertEqual(add(2, 3), 5)

if __name__ == "__main__":
    unittest.main()
```

```
❯ pytest simple.py
========== test session starts ==========
platform darwin -- Python 3.12.3, pytest-9.0.3, pluggy-1.6.0
rootdir: /Users/alexanderheidelbach/Local/BelleII_Academy
collected 1 item

simple.py .                                [100%]

========== 1 passed in 0.01s ==========
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Integration Test

```python
import tempfile
from pathlib import Path

def save_result(filename, value):
    Path(filename).write_text(str(value))

def load_result(filename):
    return int(Path(filename).read_text())

def test_save_and_load_result():
    with tempfile.NamedTemporaryFile() as tmp:
        save_result(tmp.name, 42)
        result = load_result(tmp.name)

    assert result == 42
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

System Test

```python
import sys

def main():
    name = sys.argv[1]
    print(f"Hello, {name}!")

if __name__ == "__main__":
    main()
```

```python
import subprocess

def test_app():
    result = subprocess.run(
        ["python", "app.py", "BelleII"],
        capture_output=True,
        text=True
    )

    assert result.stdout.strip() == "Hello, BelleII!"
    assert result.returncode == 0
```

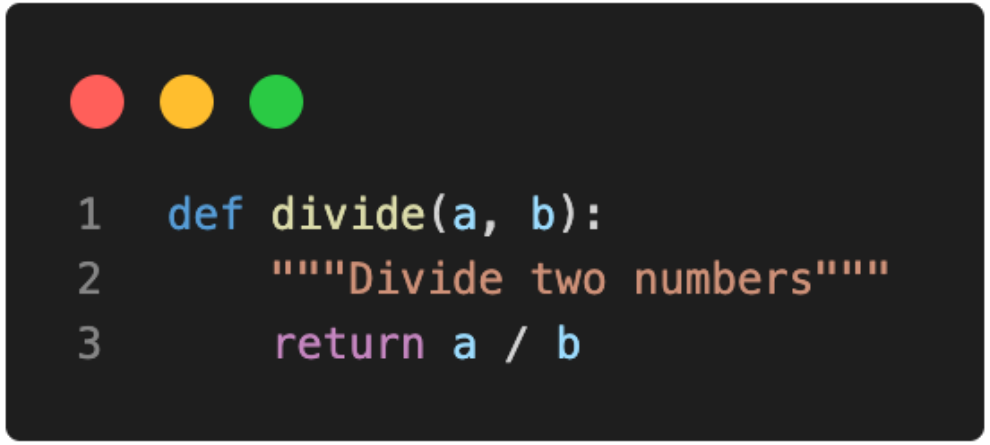
LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Test Formats

```
1  def divide(a, b):
2      """Divide two numbers"""
3      return a / b
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Test Formats



```
1  def divide(a, b):
2      """Divide two numbers"""
3      return a / b
```

Static Testing is like proofreading
But there are also tools to check structure, syntax

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Test Formats

Dynamic Testing executes the code

```python
class TestDivide(unittest.TestCase):

    def test_divide_success(self):
        self.assertEqual(divide(6, 2), 3)
```

```python
def divide(a, b):
    """Divide two numbers"""
    return a / b
```

Static Testing is like proofreading
But there are also tools to check structure, syntax

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Test Formats

```
1  def divide(a, b):
2      """Divide two numbers"""
3      return a / b
```

Static Testing is like proofreading
But there are also tools to check structure, syntax

Dynamic Testing executes the code

```
7  class TestDivide(unittest.TestCase):
8
9      def test_divide_success(self):
10         self.assertEqual(divide(6, 2), 3)
11
12     def test_divide_by_zero(self):
13         with self.assertRaises(ZeroDivisionError):
14             divide(1, 0)
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Test Formats

```python
def divide(a, b):
    """Divide two numbers"""
    return a / b
```

Static Testing is like proofreading
But there are also tools to check structure, syntax

Dynamic Testing executes the code

```python
class TestDivide(unittest.TestCase):

    def test_divide_success(self):
        self.assertEqual(divide(6, 2), 3)

    def test_divide_by_zero(self):
        with self.assertRaises(ZeroDivisionError):
            divide(1, 0)
```

```python
result = divide(10, 2)
print(f"Result: {result}")
```

Passive Testing observes the operation

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Tests in basf2 - Unittest

Code

In C++, a popular framework is the so-called Google Test framework

```cpp
#include <gtest/gtest.h>

TEST(HitOrdererTest, OrderNoHits)
{
  HitOrderer orderer;

  std::vector<KDTHit> hits = {};
  auto result = orderer.orderHits(2.0, 1.0, hits);

  EXPECT_TRUE(result.empty());
}
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Tests in basf2 - Integration

Code

```python
def run_simulation(output_file):
    """Main function to be executed if this script"""
    set_random_seed(12345)
    main = create_path()

    # specify number of events to be generated
    main.add_module('EventInfoSetter', expList=[0], evtNumList=[5], runList=[1])
    main.add_module('Progress')
    main.add_module('ParticleGun',
                    pdgCodes=[211],
                    nTracks=1,
                    momentumGeneration='fixed',
                    momentumParams=[1.618034],
                    phiGeneration='fixed',
                    phiParams=[27.182818],
                    thetaGeneration='fixed',
                    thetaParams=[62.83185])
    add_simulation(main, bkgfiles=None)
    main.add_module('RootOutput', outputFileName=output_file)
    process(main)
```

```python
def run_tracking(input_file, mode):
    set_random_seed(12345)
    main = create_path()
    main.add_module('RootInput', inputFileName=input_file)

    main.add_module('Gearbox')
    main.add_module('Geometry')
    add_tracking_reconstruction(
        main,
        pruneTracks=False,
        svd_standalone_mode=mode)
    process(main)
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Tests in basf2 - Integration

Code

```python
def run_simulation(output_file):
    """Main function to be executed if this script"""
    set_random_seed(12345)
    main = create_path()

    # specify number of events to be generated
    main.add_module('EventInfoSetter', expList=[0], evtNumList=[5], runList=[1])
    main.add_module('Progress')
    main.add_module('ParticleGun',
                    pdgCodes=[211],
                    nTracks=1,
                    momentumGeneration='fixed',
                    momentumParams=[1.618034],
                    phiGeneration='fixed',
                    phiParams=[27.182818],
                    thetaGeneration='fixed',
                    thetaParams=[62.83185])
    add_simulation(main, bkgfiles=None)
    main.add_module('RootOutput', outputFileName=output_file)
    process(main)
```

```python
def run_tracking(input_file, mode):
    set_random_seed(12345)
    main = create_path()
    main.add_module('RootInput', inputFileName=input_file)

    main.add_module('Gearbox')
    main.add_module('Geometry')
    add_tracking_reconstruction(
        main,
        pruneTracks=False,
        svd_standalone_mode=mode)
    process(main)
```

```python
if __name__ == "__main__":

    with b2tu.clean_working_directory():
        dst_file = 'dst.root'
        b2tu.run_in_subprocess(target=run_simulation, output_file=dst_file)
        for mode in [
            'VXDTF2',
            'SVDHough',
            'VXDTF2_and_SVDHough',
            'SVDHough_and_VXDTF2'
            ]:
            b2tu.run_in_subprocess(target=run_tracking, input_file=dst_file, mode=mode)

```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Tests in basf2 - Passive Testing

Code

```python
import b2test_utils
from basf2 import set_random_seed, create_path, process

# make logging more reproducible by replacing some strings
b2test_utils.configure_logging_for_tests()
set_random_seed("1337")
testinput = [b2test_utils.require_file('analysis/tests/mdst.root')]

fsps = ['pi-:all', 'gamma:all']

# a new ParticleLoader for each fsp
testpath = create_path()
testpath.add_module('RootInput', inputFileNames=testinput)
testpath.add_module('ParticleLoader', decayStrings=fsps)
testpath.add_module('ParticleSelector', decayString='gamma:all', cut='isFromECL')

# Variables created by event kinematics module
event_kinematics = [
    "missingMomentumOfEvent",
    ...
]

testpath.add_module('EventKinematics', particleLists=fsps)
# Print the variables to log
testpath.add_module('ParticlePrinter', listName='', fullPrint=False,
                    variables=event_kinematics)
process(testpath, 1)

```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

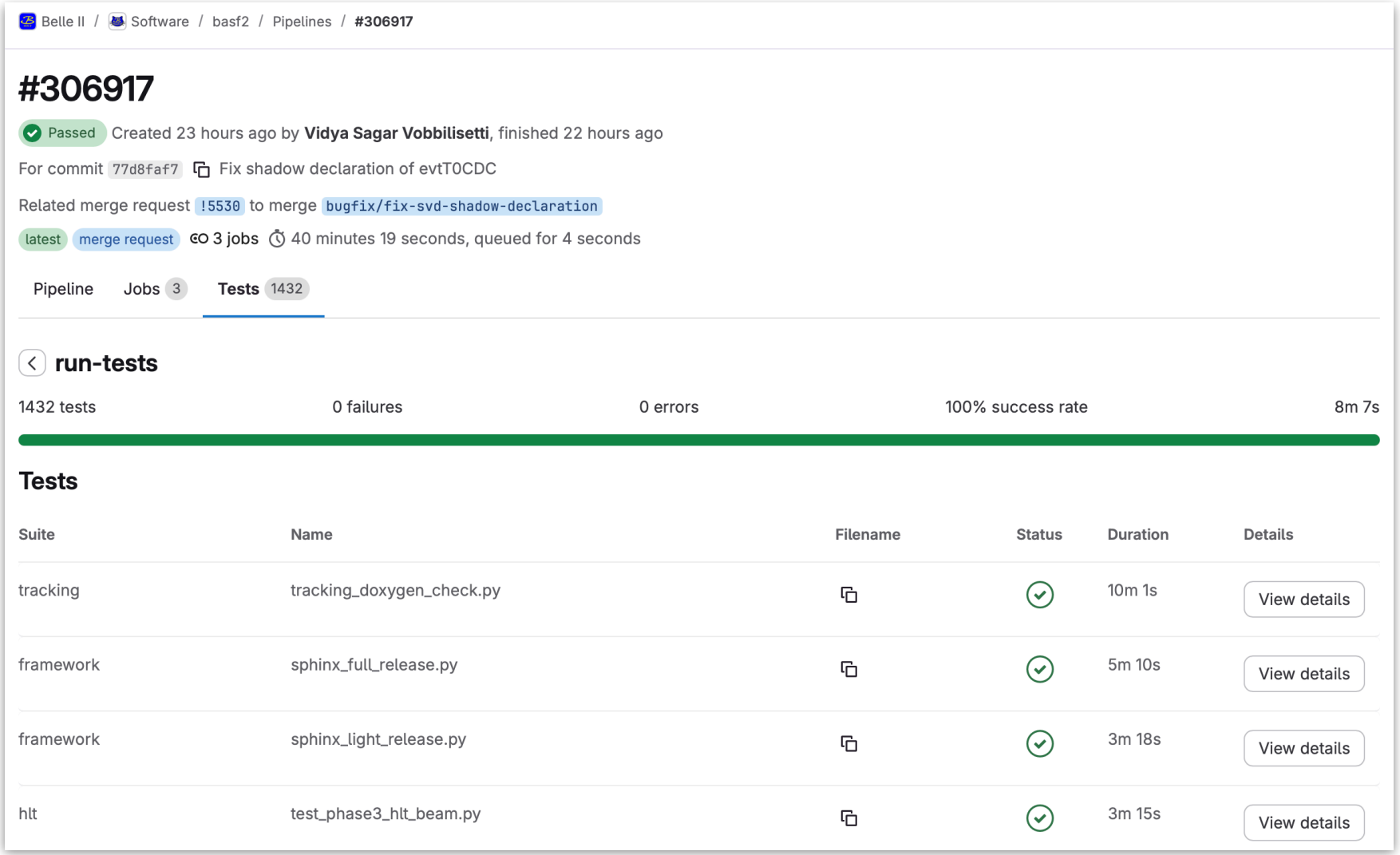
Tests in basf2 - Passive Testing

Code

```python
import b2test_utils
from basf2 import set_random_seed, create_path, process

# make logging more reproducible by replacing some strings
b2test_utils.configure_logging_for_tests()
set_random_seed("1337")
testinput = [b2test_utils.require_file('analysis/tests/mdst.root')]

fsps = ['pi-:all', 'gamma:all']

# a new ParticleLoader for each fsp
testpath = create_path()
testpath.add_module('RootInput', inputFileNames=testinput)
testpath.add_module('ParticleLoader', decayStrings=fsps)
testpath.add_module('ParticleSelector', decayString='gamma:all', cut='isFromECL')

# Variables created by event kinematics module
event_kinematics = [
    "missingMomentumOfEvent",
    ...
]

testpath.add_module('EventKinematics', particleLists=fsps)
# Print the variables to log
testpath.add_module('ParticlePrinter', listName='', fullPrint=False,
                    variables=event_kinematics)
process(testpath, 1)

```

```
[INFO] Steering file: analysis/tests/event_kinematics_variables.py
[INFO] The random number seed is set to "1337"
[INFO] Starting event processing, random seed is set to '1337'
[INFO] Added file ${BELLE2_SOFTWARE_DIR}/analysis/tests/mdst.root
[INFO] ParticleLoader's Summary of Actions:
[INFO]  o) creating (anti-)ParticleList with name: pi-:all (pi+:all)
[INFO]     -> MDST source: Tracks
[INFO]  o) creating ParticleList with name: gamma:all
[INFO]     -> MDST source: ECLClusters and KLMClusters
[INFO] ParticleSelector: gamma:all
[INFO]     -> With cuts  : isFromECL
[INFO] Conditions Database: found working metadata provider
       provider = ${BELLE2_CONDB_METADATA}
[INFO] Conditions data: configured globaltags
       (highest priority first) are release-05-01-03, Legacy_CollisionAxisCMS
[INFO]      o) missingMomentumOfEvent = 1.09551

```

Tests in basf2 - Summary



Belle II / Software / basf2 / Pipelines / #306917

#306917

Passed Created 23 hours ago by **Vidya Sagar Vobbilisetti**, finished 22 hours ago

For commit `77d8faf7` Fix shadow declaration of evtT0CDC

Related merge request `!5530` to merge `bugfix/fix-svd-shadow-declaration`

latest merge request 3 jobs 40 minutes 19 seconds, queued for 4 seconds

Pipeline Jobs 3 **Tests** 1432

run-tests

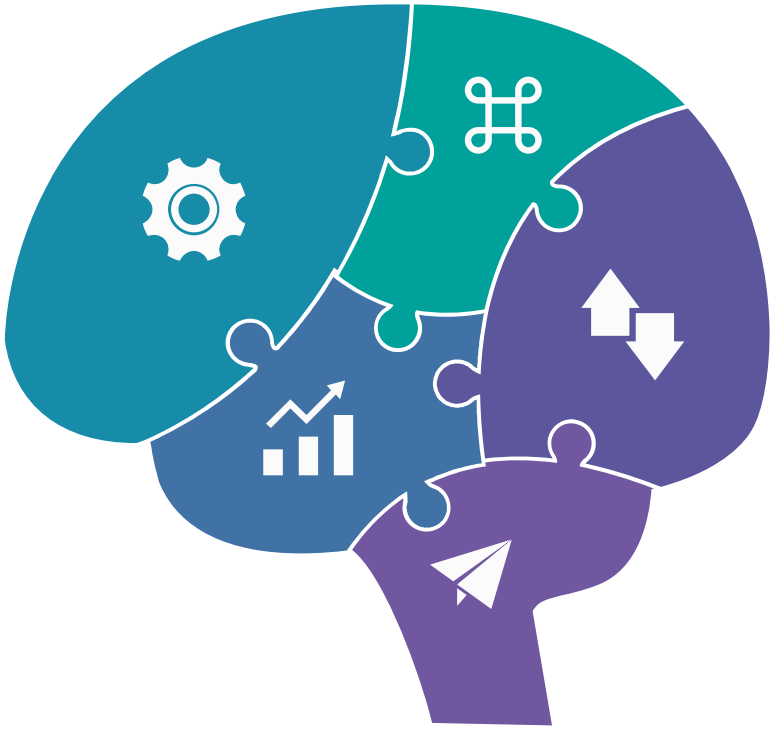
1432 tests | 0 failures | 0 errors | 100% success rate | 8m 7s

Tests

Suite	Name	Filename	Status	Duration	Details
tracking	tracking_doxygen_check.py		✓	10m 1s	View details
framework	sphinx_full_release.py		✓	5m 10s	View details
framework	sphinx_light_release.py		✓	3m 18s	View details
hlt	test_phase3_hlt_beam.py		✓	3m 15s	View details

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

A Few Remarks

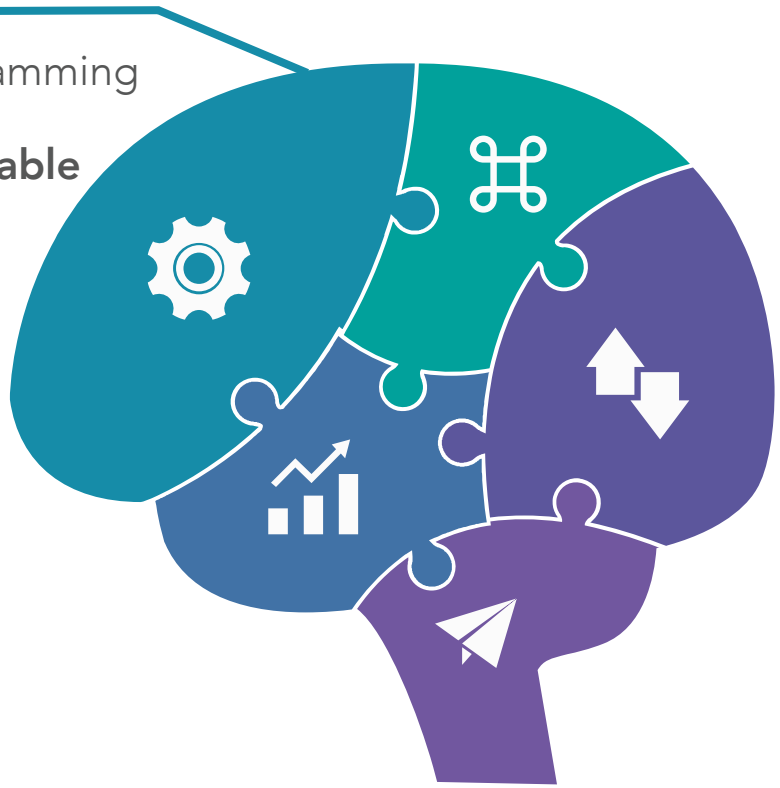


LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

A Few Remarks

Functional Programming

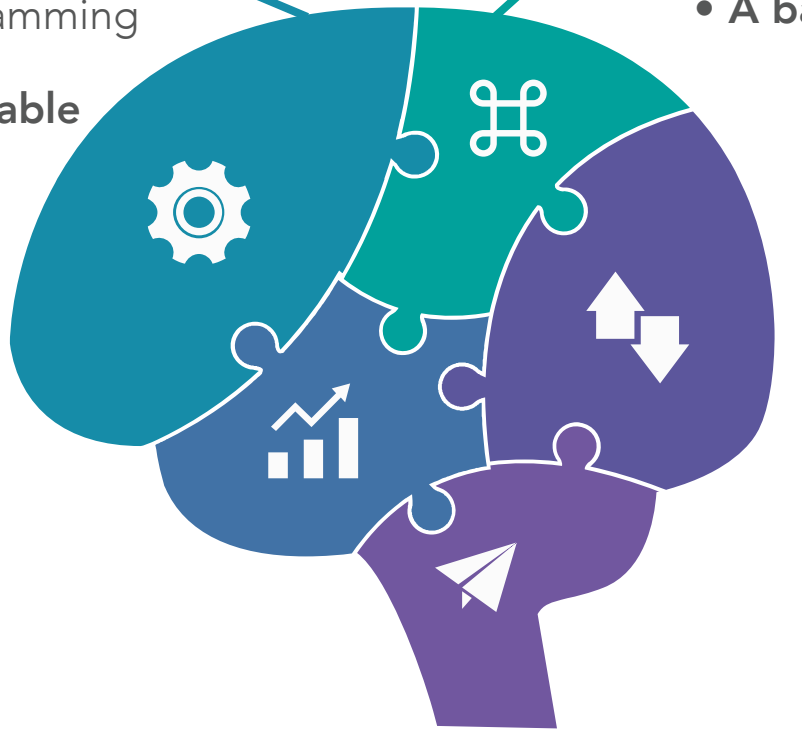
- **Object-orientated** and **functional** programming
- Test **small** and independent **units**
- Tests: **independent**, **isolated**, and **repeatable**



A Few Remarks

Functional Programming

- **Object-orientated** and **functional** programming
- Test **small** and independent **units**
- Tests: **independent**, **isolated**, and **repeatable**



"Good" Tests

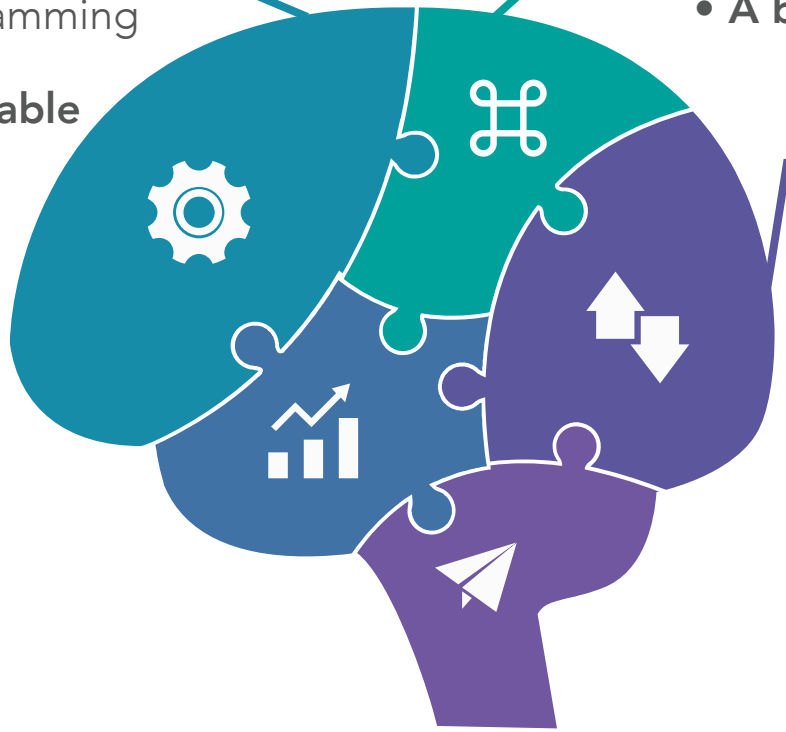
- Test depicts the **desired functionality**
- **A bad test can be more harmful than no test**

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

A Few Remarks

Functional Programming

- **Object-orientated** and **functional** programming
- Test **small** and independent **units**
- Tests: **independent**, **isolated**, and **repeatable**



"Good" Tests

- Test depicts the **desired functionality**
- **A bad test can be more harmful than no test**

Bottom Up and Top Down

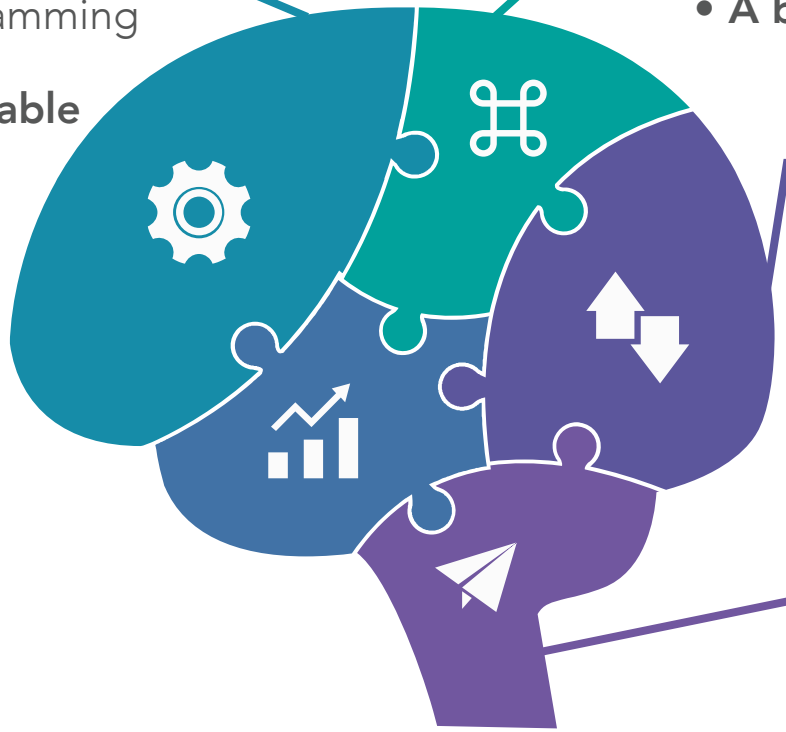
- Two approaches:
  - Write a unit and **test for its functionality**
  - Write a **test first** with your desired behaviour and then the function that passes the test

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

A Few Remarks

Functional Programming

- **Object-orientated** and **functional** programming
- Test **small** and independent **units**
- Tests: **independent**, **isolated**, and **repeatable**



"Good" Tests

- Test depicts the **desired functionality**
- **A bad test can be more harmful than no test**

Bottom Up and Top Down

- Two approaches:
  - Write a unit and **test for its functionality**
  - Write a **test first** with your desired behaviour and then the function that passes the test

Start Early

- Write tests **early** and **continuously** throughout development
- Let it become an **unconscious habit** while you're writing code

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

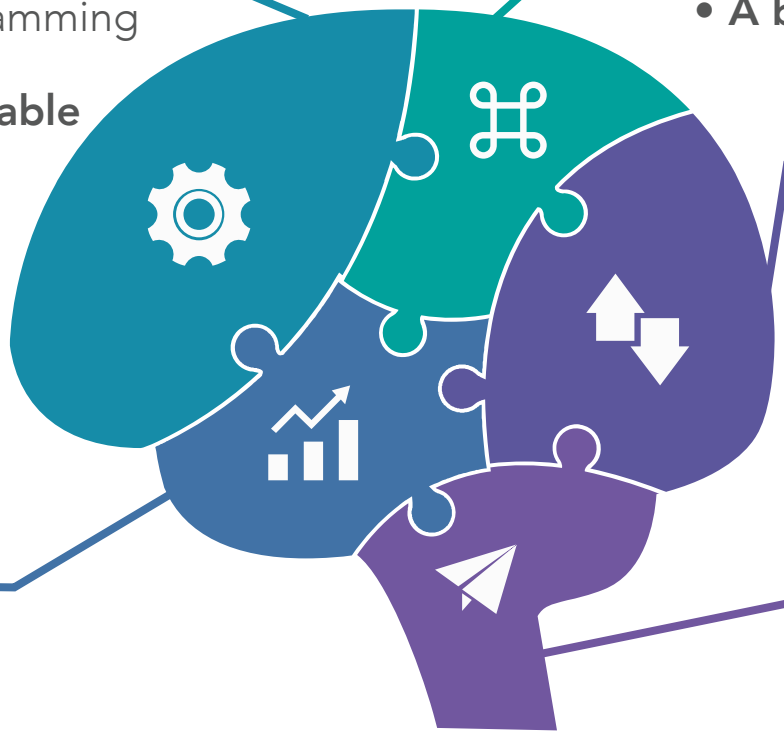
A Few Remarks

Functional Programming

- **Object-orientated** and **functional** programming
- Test **small** and independent **units**
- Tests: **independent**, **isolated**, and **repeatable**

"Good" Tests

- Test depicts the **desired functionality**
- **A bad test can be more harmful than no test**



Bottom Up and Top Down

- Two approaches:
  - Write a unit and **test for its functionality**
  - Write a **test first** with your desired behaviour and then the function that passes the test

Modern Tools

- **Ready-to-use libraries** for testing
- **AI** tools to write tests
- **Coverage** tools
- Integrate testing in an **automated workflow** (e.g. GitLab CI/CD)

Start Early

- Write tests **early** and **continuously** throughout development
- Let it become an **unconscious habit** while you're writing code

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

What is Software Development?

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

CI/CD

Continuous Integration (CI)

- Frequently merge code into a shared branch
- Each change is automatically built and tested

Continuous Delivery (CD)

- Software is built, tested, and prepared for release continuously
- Changes are delivered in small, frequent increments
- The system is always ready to be released

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

CI/CD

Continuous Integration (CI)

- Frequently merge code into a shared branch
- Each change is automatically built and tested

Continuous Delivery (CD)

- Software is built, tested, and prepared for release continuously
- Changes are delivered in small, frequent increments
- The system is always ready to be released

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

CI/CD

Continuous Integration (CI)

- Frequently merge code into a shared branch
- Each change is automatically built and tested

Continuous Delivery (CD)

- Software is built, tested, and prepared for release continuously
- Changes are delivered in small, frequent increments
- The system is always ready to be released

Automate everything that ensures software quality

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Pipelines

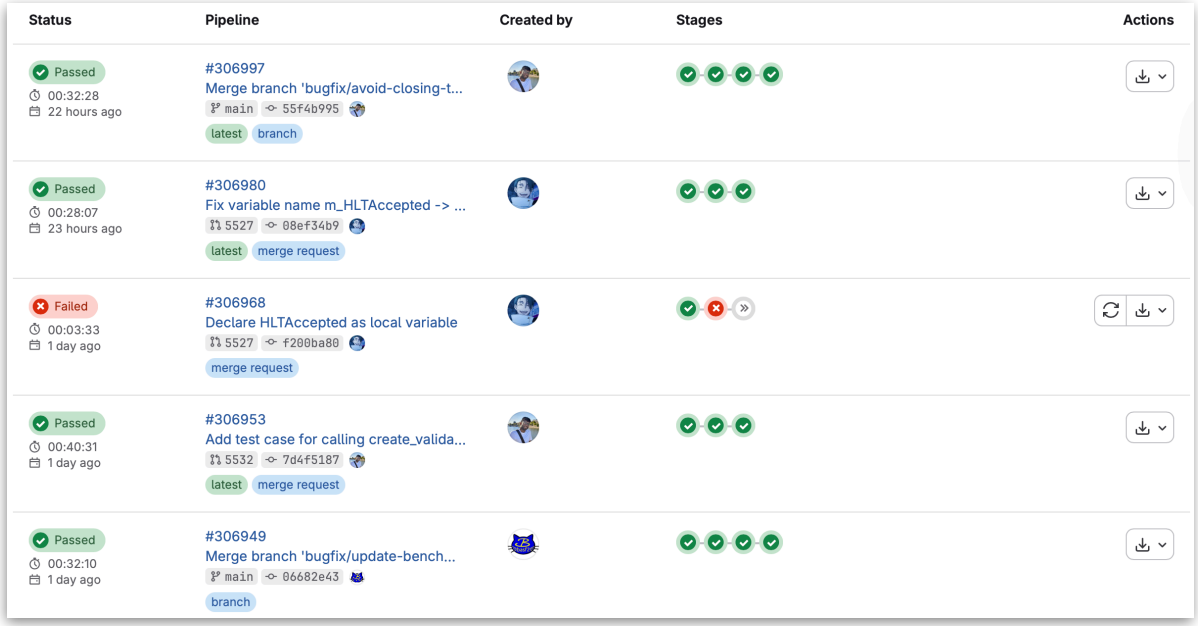
Framework to automatically start the CI(/CD) process

Pipeline is split into

- Stages - Collection of jobs
- Jobs - Single task

Example: Test stage

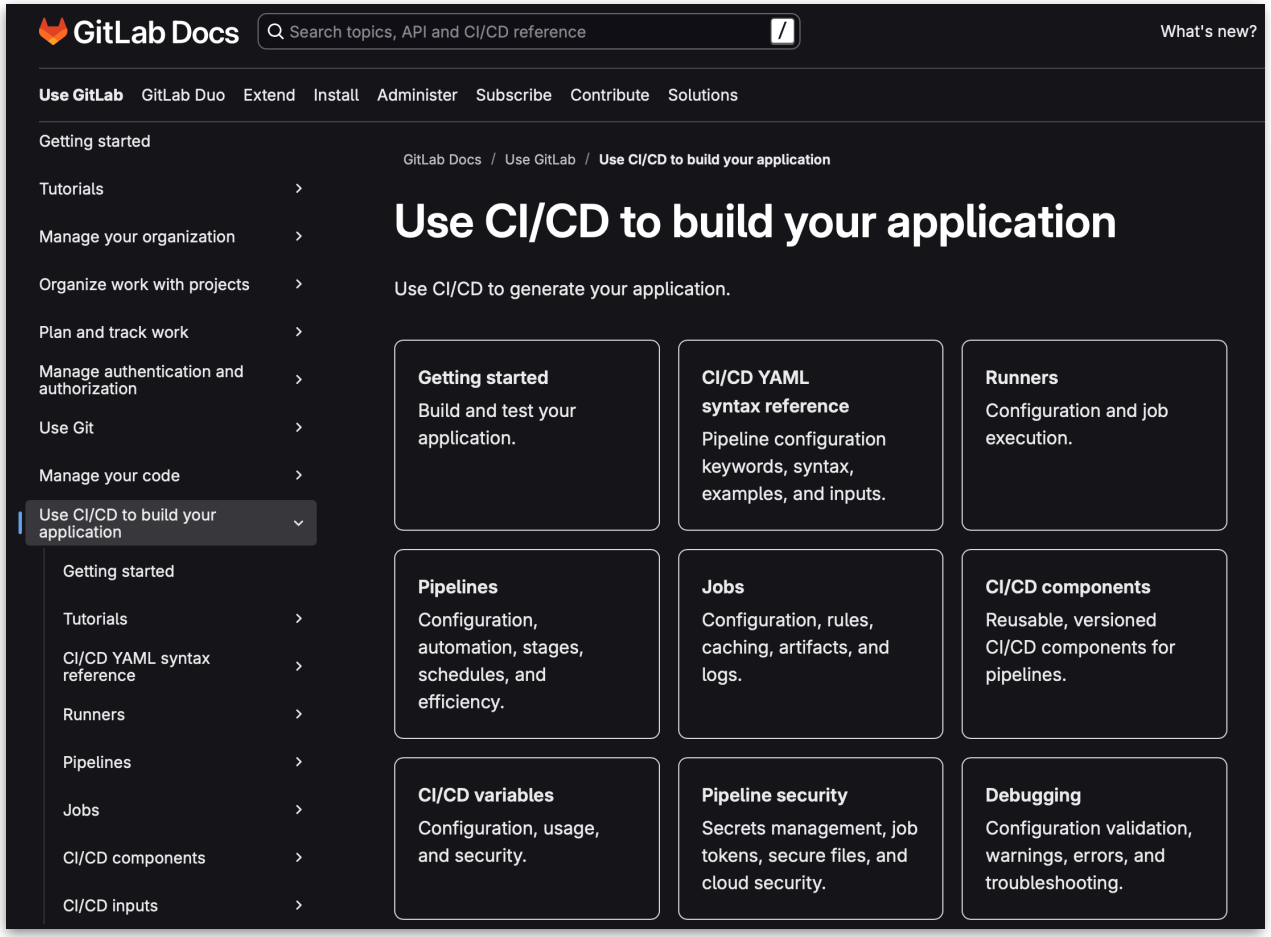
- Prepare the environment
- Execute the tests
- Upload the test results

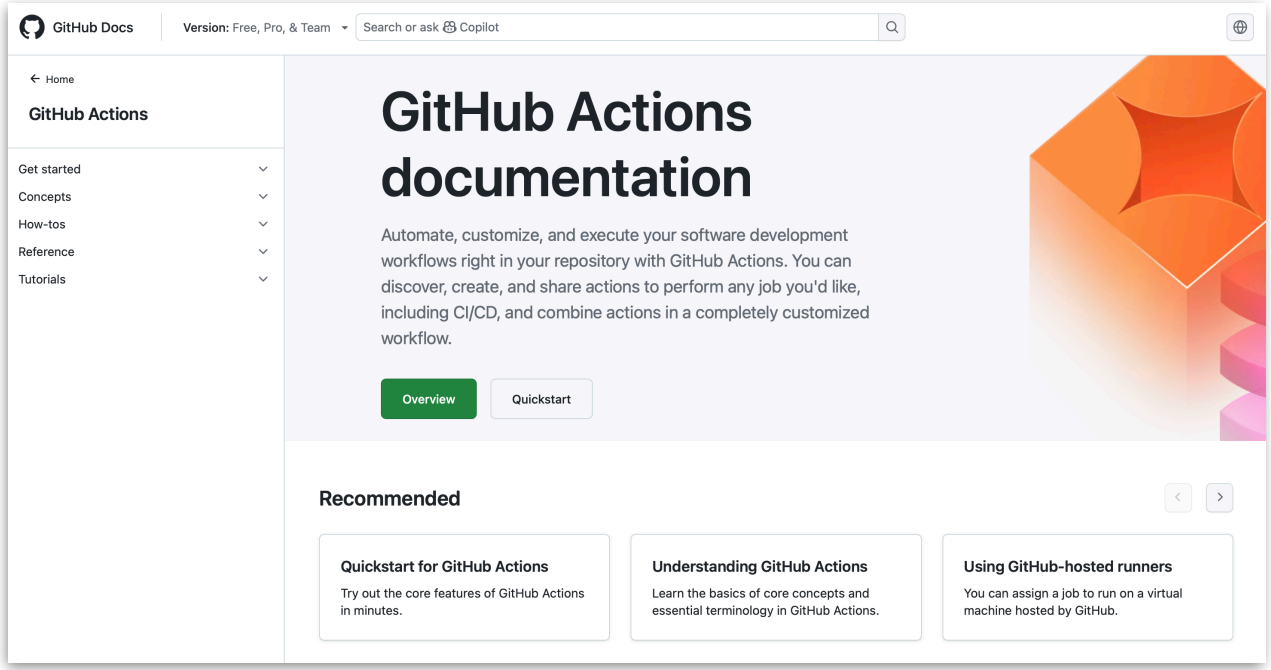


Code → Build → Test → Documentation → Integrate

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Automation

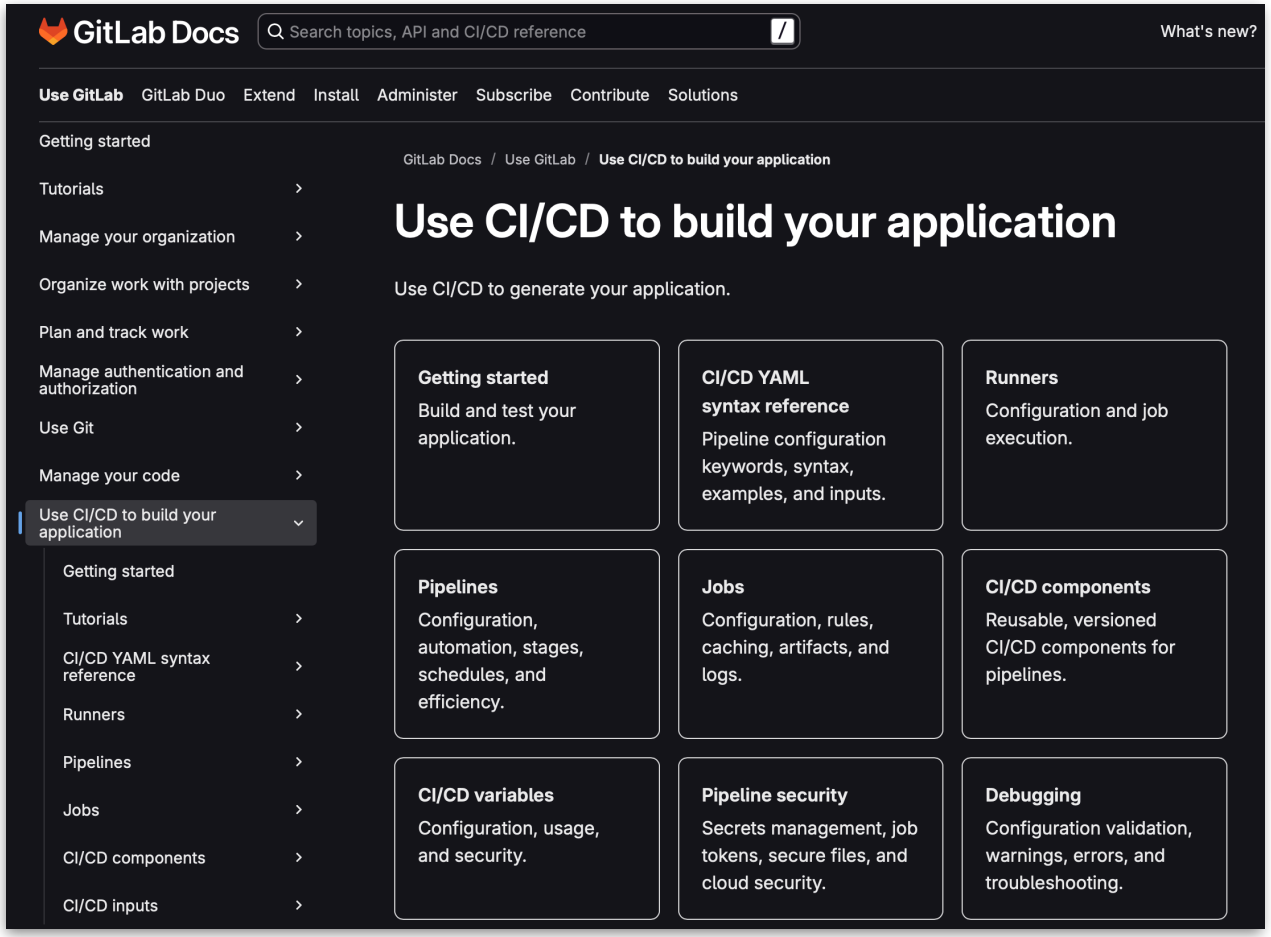


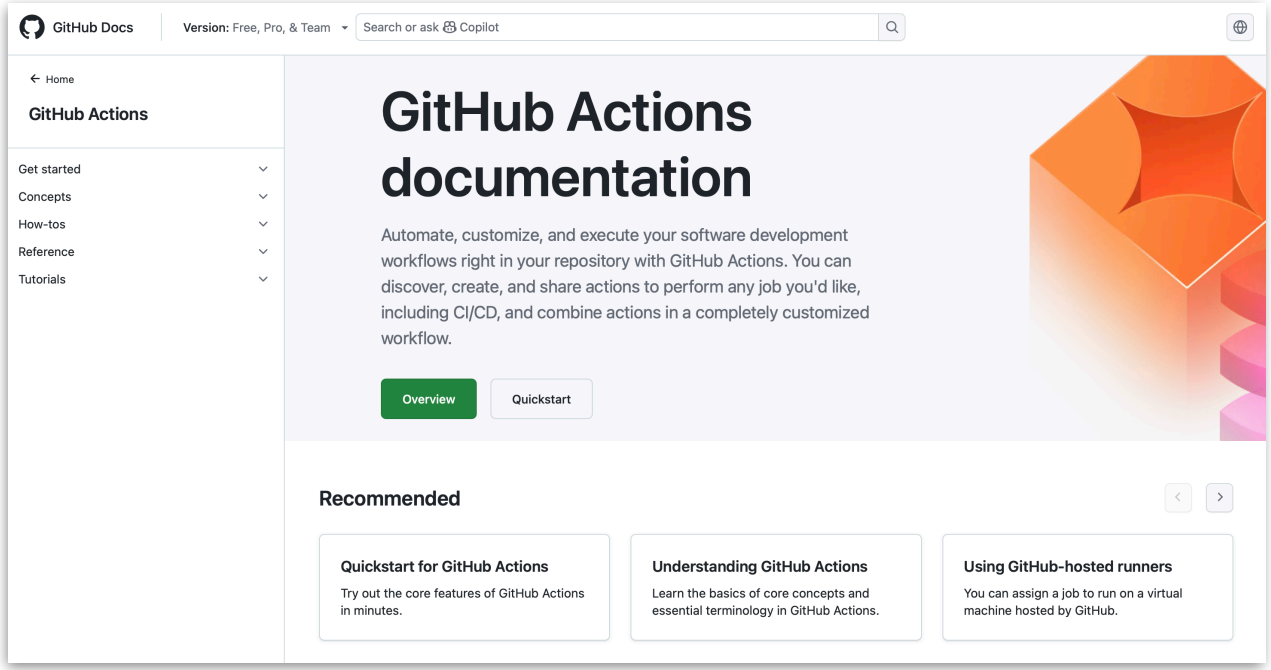


Automation

We will focus on GitLab

But the ideas stay the same





LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Hello World!

```yaml
stages:
  - build
  - test

build_job:
  stage: build
  script:
    - echo "Building the project"

test_job:
  stage: test
  script:
    - echo "Running tests"
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Hello World!

Defines the order of execution
Pipelines run stage by stage

```yaml
stages:
  - build
  - test

build_job:
  stage: build
  script:
    - echo "Building the project"

test_job:
  stage: test
  script:
    - echo "Running tests"
```

Hello World!

Defines the order of execution
Pipelines run stage by stage

Arbitrary names you choose
Each job = one task

```yaml
stages:
  - build
  - test

build_job:
  stage: build
  script:
    - echo "Building the project"

test_job:
  stage: test
  script:
    - echo "Running tests"
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Hello World!

Defines the order of execution
Pipelines run stage by stage

Arbitrary names you choose
Each job = one task

Assigns a job to a stage
Controls when it runs

```yaml
stages:
  - build
  - test

build_job:
  stage: build
  script:
    - echo "Building the project"

test_job:
  stage: test
  script:
    - echo "Running tests"
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Hello World!

Defines the order of execution
Pipelines run stage by stage

Arbitrary names you choose
Each job = one task

Assigns a job to a stage
Controls when it runs

The actual commands executed
Runs in a shell on the runner

```yaml
stages:
  - build
  - test

build_job:
  stage: build
  script:
    - echo "Building the project"

test_job:
  stage: test
  script:
    - echo "Running tests"
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Some More Features

```yaml
image: python:3.11

stages:
  - build
  - test

before_script:
  - python --version

build_job:
  stage: build
  script:
    - mkdir build
    - echo "Build output" > build/result.txt
  artifacts:
    paths:
      - build/

test_job:
  stage: test
  script:
    - cat build/result.txt
    - python -m unittest discover
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Some More Features

Defines the container used to run the jobs

```yaml
image: python:3.11

stages:
  - build
  - test

before_script:
  - python --version

build_job:
  stage: build
  script:
    - mkdir build
    - echo "Build output" > build/result.txt
  artifacts:
    paths:
      - build/

test_job:
  stage: test
  script:
    - cat build/result.txt
    - python -m unittest discover
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Some More Features

Defines the container used to run the jobs

Runs setup commands before each job

```yaml
image: python:3.11

stages:
  - build
  - test

before_script:
  - python --version

build_job:
  stage: build
  script:
    - mkdir build
    - echo "Build output" > build/result.txt
  artifacts:
    paths:
      - build/

test_job:
  stage: test
  script:
    - cat build/result.txt
    - python -m unittest discover
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Some More Features

Defines the container used to run the jobs

Runs setup commands before each job

Stores files from one job so later jobs can use them

```yaml
image: python:3.11

stages:
  - build
  - test

before_script:
  - python --version

build_job:
  stage: build
  script:
    - mkdir build
    - echo "Build output" > build/result.txt
  artifacts:
    paths:
      - build/

test_job:
  stage: test
  script:
    - cat build/result.txt
    - python -m unittest discover
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Even More Features

```yaml
image: python:3.11

stages:
  - build
  - test

variables:
  OUTPUT_FILE: "build/result.txt"

before_script:
  - python --version

build_job:
  stage: build
  script:
    - mkdir -p build
    - echo "Build output" > $OUTPUT_FILE
  artifacts:
    paths:
      - build/

test_job:
  stage: test
  needs: [build_job]
  script:
    - cat $OUTPUT_FILE
    - python -m unittest discover

lint_job:
  stage: test
  script:
    - pip install flake8
    - flake8 .
  rules:
    - if: '$CI_PIPELINE_SOURCE == "merge_request_event"'
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Even More Features

Define reusable values
Avoid hardcoding paths

```yaml
image: python:3.11

stages:
  - build
  - test

variables:
  OUTPUT_FILE: "build/result.txt"

before_script:
  - python --version

build_job:
  stage: build
  script:
    - mkdir -p build
    - echo "Build output" > $OUTPUT_FILE
  artifacts:
    paths:
      - build/

test_job:
  stage: test
  needs: [build_job]
  script:
    - cat $OUTPUT_FILE
    - python -m unittest discover

lint_job:
  stage: test
  script:
    - pip install flake8
    - flake8 .
  rules:
    - if: '$CI_PIPELINE_SOURCE == "merge_request_event"'
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Even More Features

Define reusable values
Avoid hardcoding paths

Controlls order of the jobs

```yaml
image: python:3.11

stages:
  - build
  - test

variables:
  OUTPUT_FILE: "build/result.txt"

before_script:
  - python --version

build_job:
  stage: build
  script:
    - mkdir -p build
    - echo "Build output" > $OUTPUT_FILE
  artifacts:
    paths:
      - build/

test_job:
  stage: test
  needs: [build_job]
  script:
    - cat $OUTPUT_FILE
    - python -m unittest discover

lint_job:
  stage: test
  script:
    - pip install flake8
    - flake8 .
  rules:
    - if: '$CI_PIPELINE_SOURCE == "merge_request_event"'
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Even More Features

Define reusable values
Avoid hardcoding paths

Controlls order of the jobs

Control when jobs run
More: `workflows`

```yaml
image: python:3.11

stages:
  - build
  - test

variables:
  OUTPUT_FILE: "build/result.txt"

before_script:
  - python --version

build_job:
  stage: build
  script:
    - mkdir -p build
    - echo "Build output" > $OUTPUT_FILE
  artifacts:
    paths:
      - build/

test_job:
  stage: test
  needs: [build_job]
  script:
    - cat $OUTPUT_FILE
    - python -m unittest discover

lint_job:
  stage: test
  script:
    - pip install flake8
    - flake8 .
  rules:
    - if: '$CI_PIPELINE_SOURCE == "merge_request_event"'
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

One Real Example

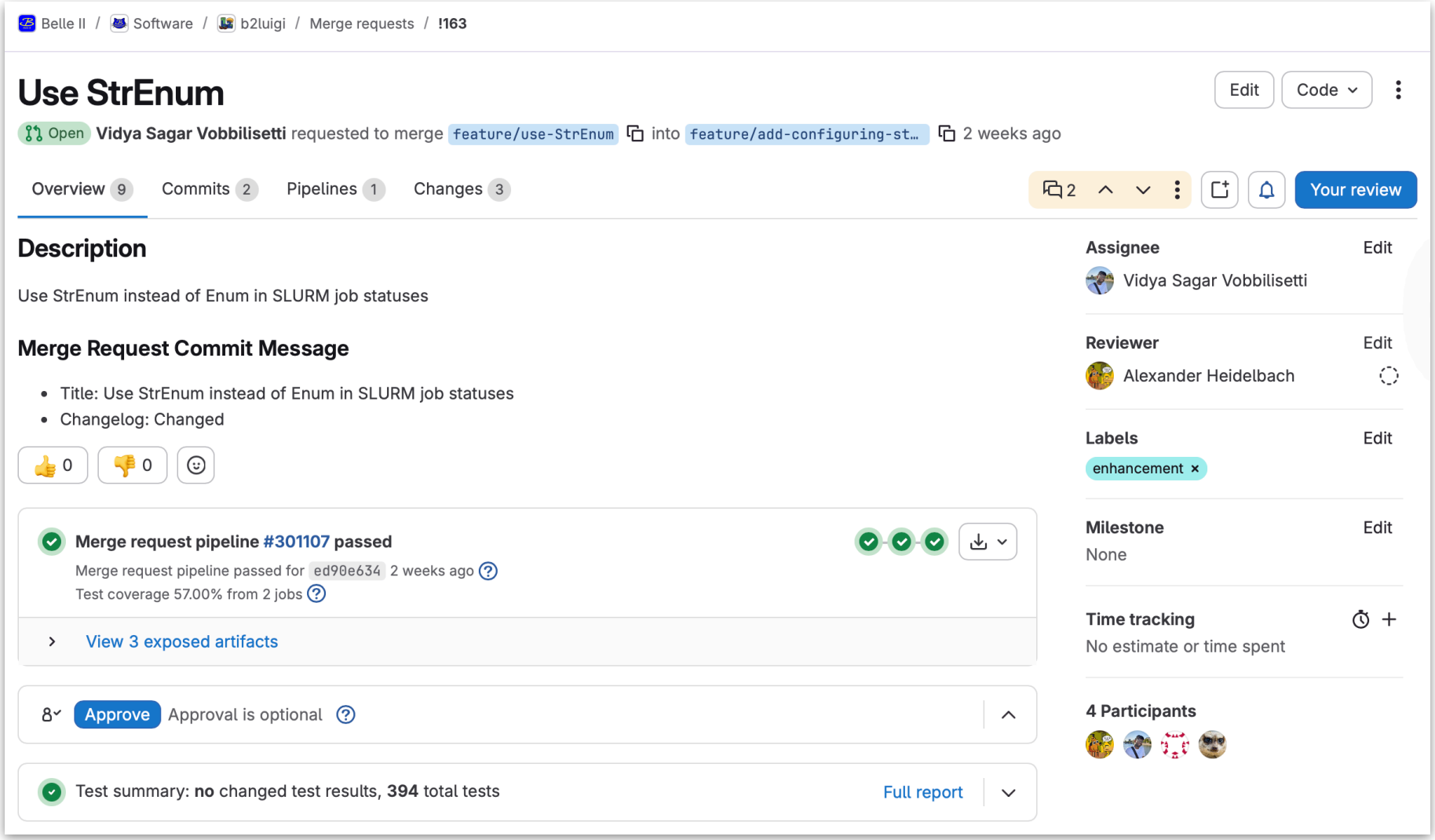
```yaml
stages:
  - pre_commit
  - test
  - docs
  - publish

variables:
  FLIT_ROOT_INSTALL: "1"
  PYTHON_VERSION: "3.8.8"
  ARTIFACTS_TEST_PATH: "junit/test-results.xml"
  ARTIFACTS_DOCS_PATH: "docs-build"
  ARTIFACTS_COV_PATH: "coverage/"
  GITLAB_BELLE2_TOOLS_B2SETUP: "/cvmfs/belle.cern.ch/tools/b2setup"
  FLIT_INDEX_URL: "https://upload.pypi.org/legacy/"
  FLIT_USERNAME: __token__
  BASF2_RELEASE_VERSION: "light-2409-toyger"

workflow:
  rules:
    - if: $CI_PIPELINE_SOURCE == 'merge_request_event'
    - if: $CI_PIPELINE_SOURCE == 'push' && ($CI_COMMIT_BRANCH == 'main') && $CI_COMMIT_TITLE =~ /Merge branch.*/
    - if: $CI_COMMIT_TAG
    - if: $GITLAB_FORCE_RUNNING_PIPELINE == 'yes'

default:
  image: belle2/belle2-base-el9:latest
  tags:
    - gitlab-runner12
  before_script:
    # Preliminary operations with Git
    - git config user.email "something@something.com"
    - git config user.name "someone"
    - git config --global --add safe.directory $(dirname ${GITLAB_BELLE2_TOOLS_B2SETUP})
    # Setup of basf2
    - source ${GITLAB_BELLE2_TOOLS_B2SETUP}
    - echo ${BASF2_RELEASE_VERSION}
    - b2venv ${BASF2_RELEASE_VERSION}
    - source venv/bin/activate
    - pip install flit
  interruptible: true  # All the jobs can be interrupted by newer pipelines

pre_commit:
  stage: pre_commit
  script:
    - pip install pre-commit
    - pre-commit run --all-files
```

```yaml
test:
  stage: test
  script:
    - flit install -s --deps=develop
    - echo "Running tests"
    - pytest -v --cov b2luigi --junitxml=$ARTIFACTS_TEST_PATH --cov-report=xml --cov-report=html --cov-report=term tests
    - mv htmlcov $ARTIFACTS_COV_PATH
  coverage: '/(?i)total.*? (100(?:\.0+)?\%|[1-9]?\d(?:\.\d+)?\%)$/'
  parallel:
    matrix:
      - BASF2_RELEASE_VERSION: [$BASF2_PYTHON38_RELEASE, $BASF2_PYTHON311_RELEASE]
  artifacts:
    reports:
      junit: $ARTIFACTS_TEST_PATH
      coverage_report:
        coverage_format: cobertura
        path: coverage.xml
    paths:
      - junit/test-results.xml
      - coverage/
    expose_as: 'test result and coverage'

docs:
  stage: docs
  needs:
    - job: test
      artifacts: false
  script:
    - flit install -s --deps=develop
    - sphinx-build docs/ $ARTIFACTS_DOCS_PATH
  artifacts:
    paths:
      - docs-build/
    expire_in: 1 week
    expose_as: 'sphinx documentation'
    when: on_success

publish:
  stage: publish
  needs:
    - job: test
      artifacts: false
    - job: docs
      artifacts: false
  script:
    - pip install setuptools wheel twine
    - flit publish
  only:
    - tags
```

```yaml
prepare_release:
  stage: publish
  rules:
  - if: '$CI_COMMIT_TAG =~ /^v?\d+\.\d+\.\d+$/'
  script:
    - yum install -y jq
    - 'curl --header "JOB-TOKEN: ${CI_JOB_TOKEN}"
      "$CI_API_V4_URL/projects/$CI_PROJECT_ID/repository/
       changelog?version=$CI_COMMIT_TAG" | jq -r .notes > release_notes.md'
  artifacts:
    paths:
    - release_notes.md

publish_release:
  stage: publish
  image: registry.gitlab.com/gitlab-org/release-cli:latest
  before_script: []
  needs:
    - job: prepare_release
      artifacts: true
  rules:
  - if: '$CI_COMMIT_TAG =~ /^v?\d+\.\d+\.\d+$/'
  script:
    - echo "Creating release"
  release:
    name: 'Release $CI_COMMIT_TAG'
    description: release_notes.md
    tag_name: '$CI_COMMIT_TAG'
    ref: '$CI_COMMIT_SHA'
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

How does this look in practice?



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

How does this look in practice?

Merge request pipeline #301107 passed

Merge request pipeline passed for `ed90e634` 2 weeks ago

Test coverage 57.00% from 2 jobs

Collapse

Artifact	Job
test result and coverage	test: [$BASF2_PYTHON311_RELEASE]
sphinx documentation	docs
test result and coverage	test: [$BASF2_PYTHON38_RELEASE]

How does this look in practice?

passed **Job #1210489** in pipeline **#301107** for `ed90e634` from `refs/merge-requests/163/head` by  **Vidya Sagar Vobbilisetti** 2 weeks ago

Artifacts

Download artifacts archive

Name	Size
docs-build	

How does this look in practice?

passed **Job #1210489** in pipeline **#301107** for `ed90e634` from `refs/merge-requests/163/head` by **Vidya Sagar Vobbilisetti** 2 weeks ago

Belle II / Software / b2luigi / Jobs / #1210489 / **Artifacts**

development.html	27.8 KiB
examples.html	21 KiB
faq.html	45.9 KiB
features.html	24.3 KiB
genindex.html	65.9 KiB
index.html	36.2 KiB
installation.html	22.4 KiB

How does this look in practice?

passed **Job #1210489** in pipeline **#301107** for `ed90e634` from `refs/merge-requests/163/head` by **Vidya Sagar Vobbilisetti** 2 weeks ago

Belle II / Software / b2luigi / Jobs / #1210489 / **Artifacts**

development.html 27.8 KiB

21 KiB

passed **Job #1210489** in pipeline **#301107** for `ed90e634` from `refs/merge-requests/163/head` by **Vidya Sagar Vobbilisetti** 2 weeks ago

Artifacts

You are being redirected away from GitLab

This page is hosted on GitLab pages but contains user-generated content and may contain malicious code. Do not accept unless you trust the author and source.

http://belle2.pages.desy.de/-/software/b2luigi/-/jobs/1210489/artifacts/docs-build/index.html

installation.html 22.4 KiB

How does this look in practice?

Merge request pipeline #301107 passed

Merge request pipeline passed for `ed90e634` 2 weeks ago

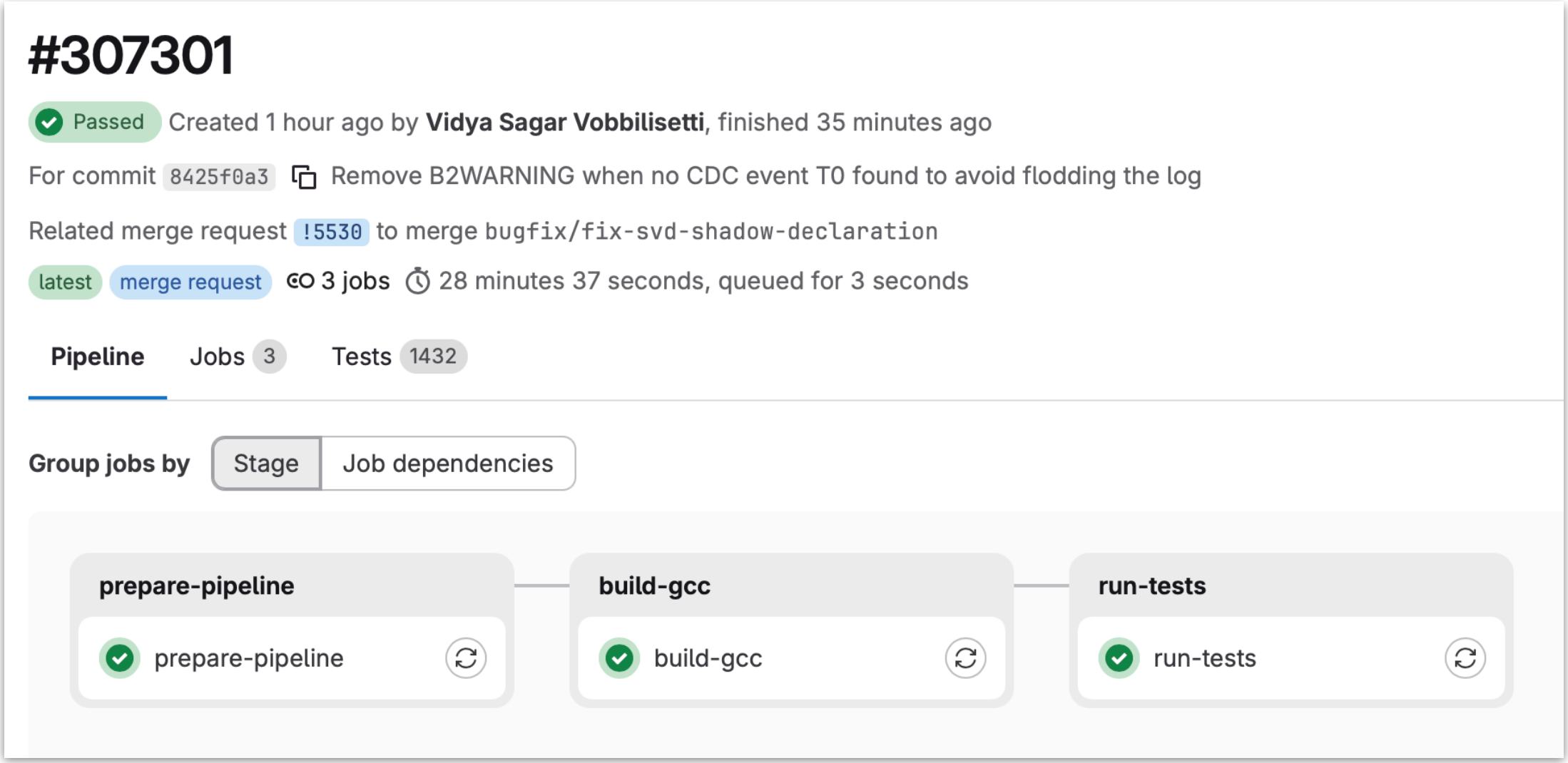
Test coverage 57.00% from 2 jobs

Collapse

Artifact	Job
test result and coverage	test: [$BASF2_PYTHON311_RELEASE]
sphinx documentation	docs
test result and coverage	test: [$BASF2_PYTHON38_RELEASE]

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

How does this look in practice?



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

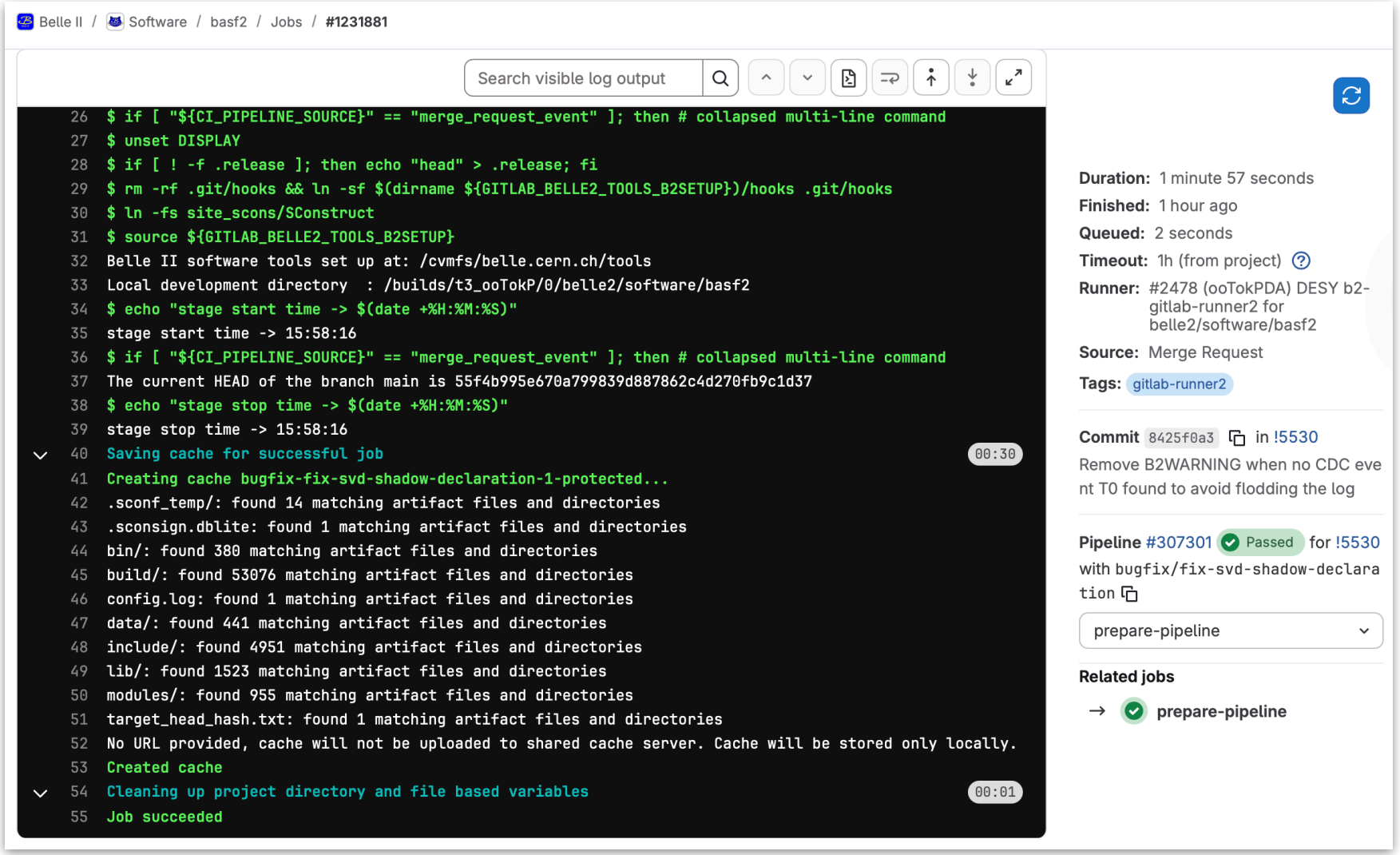
How does this look in practice?

Pipeline | **Jobs** 3 | Tests 1432

Status	Job	Stage	Coverage
Passed 00:24:17 37 minutes ago	#1231883: run-tests refs/merge-requests/553... 8425f0a3 gitlab-runner2	run-tests	
Passed 00:02:21 1 hour ago	#1231882: build-gcc refs/merge-requests/553... 8425f0a3 gitlab-runner2	build-gcc	
Passed 00:01:57 1 hour ago	#1231881: prepare-pipeline refs/merge-requests/553... 8425f0a3 gitlab-runner2	prepare-pipeline	

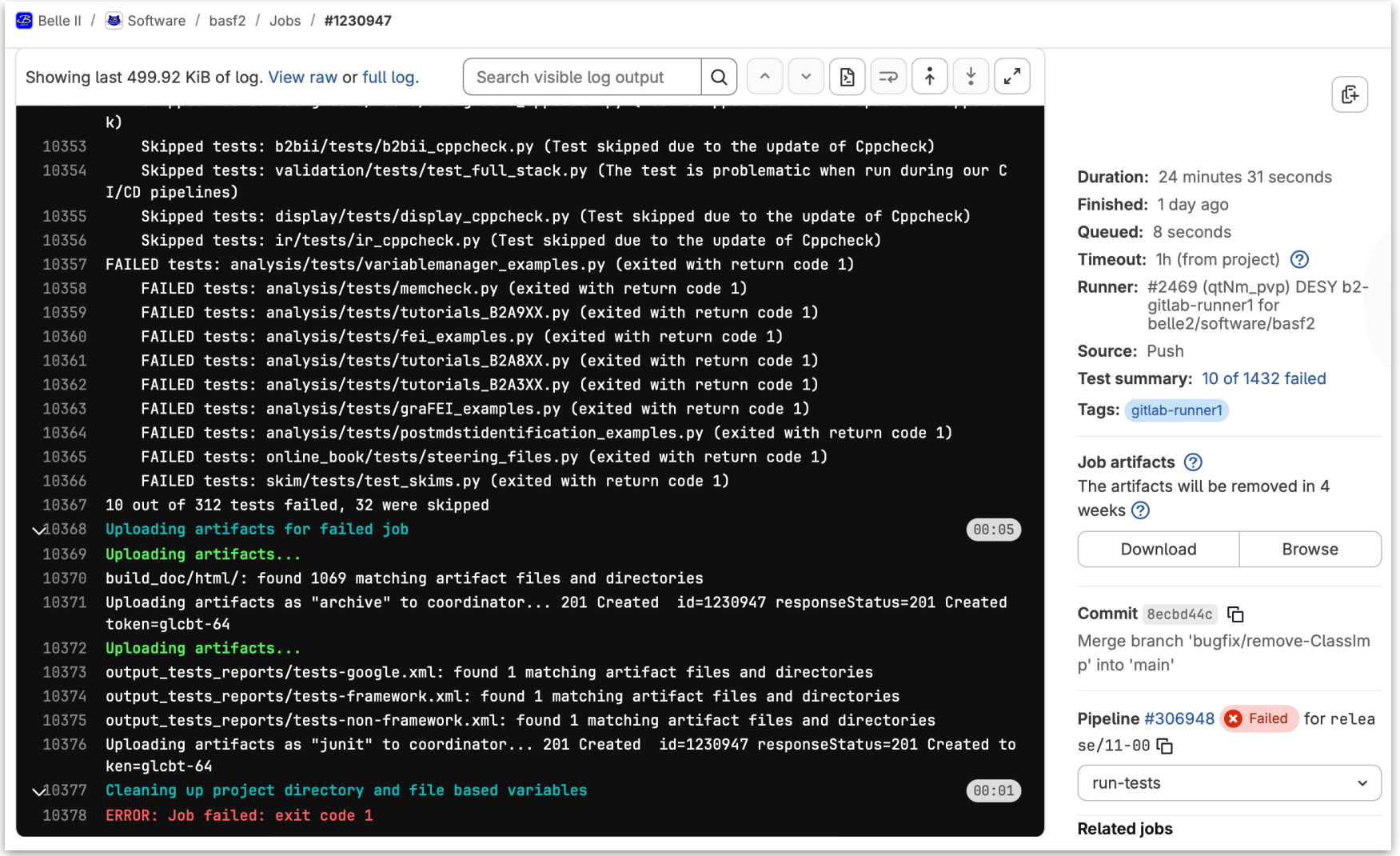
LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

How does this look in practice?



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

How does this look in practice?



How does this look in practice?

Belle II / Software / basf2 / Pipelines / **#307301**

#307301

Passed Created 1 hour ago by **Vidya Sagar Vobbilisetti**, finished 40 minutes ago

For commit `8425f0a3` Remove B2WARNING when no CDC event T0 found to avoid flodding the log

Related merge request `!5530` to merge `bugfix/fix-svd-shadow-declaration`

latest merge request 3 jobs 28 minutes 37 seconds, queued for 3 seconds

Pipeline | Jobs 3 | **Tests** 1432

Summary

1432 tests | 0 failures | 0 errors | 100% success rate | 8m 10s

Jobs

Job	Execution time	Failed	Errors	Skipped	Passed	Total
run-tests	8m 10s	0	0	37	1395	1432

How does this look in practice?

Pipeline | Jobs 3 | **Tests 1432**

run-tests

1432 tests | 0 failures | 0 errors | 100% success rate | 8m 10s

Tests

Suite	Name	Filename	Status	Duration	Details
tracking	tracking_doxygen_check.py		✓	9m 48s	View details
framework	sphinx_full_release.py		✓	5m 15s	View details
framework	sphinx_light_release.py		✓	3m 24s	View details
hlt	test_phase3_hlt_beam.py		✓	3m 15s	View details
daq/hbasf2	test_histogram.py		✓	3m 7s	View details
analysis	code-quality-doxygen.py		✓	3m 5s	View details

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

What is Software Development?

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Why Documentation?

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

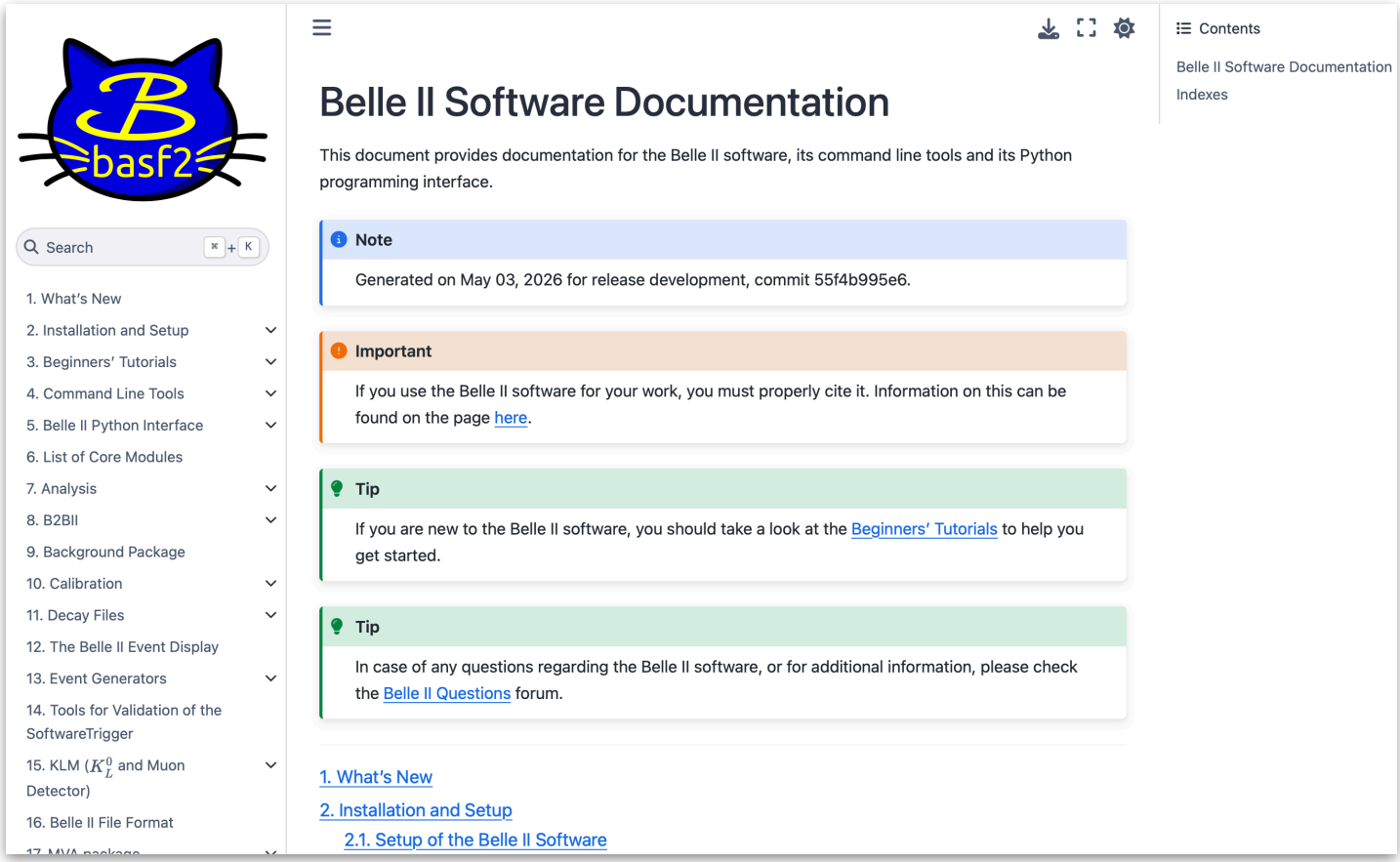
Why Documentation?

```
If it is not documented, it does not exist
```

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Why Documentation?

If it is not documented, it does not exist



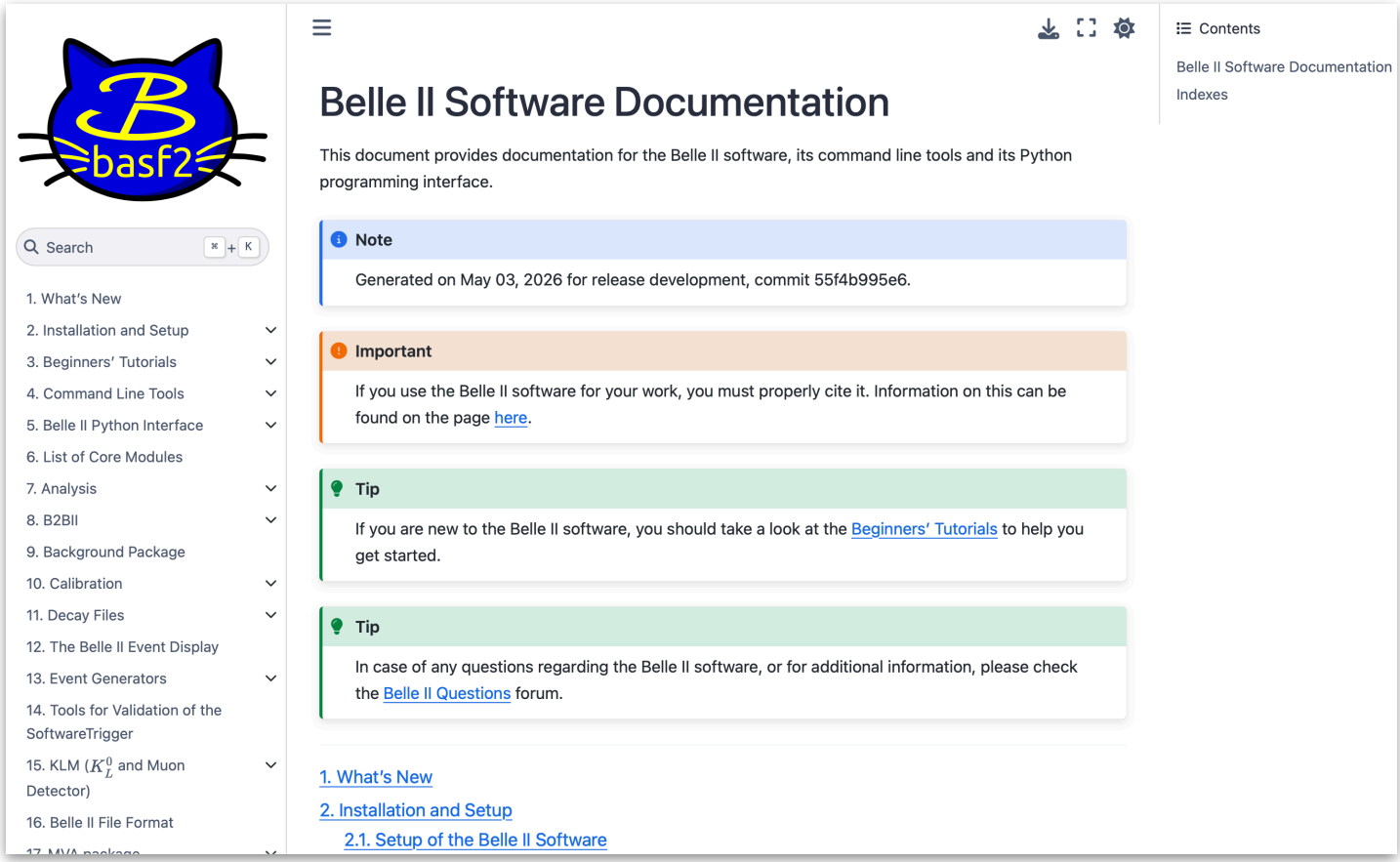
LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Why Documentation?

`If it is not documented, it does not exist`

What is documented?

- Code
- Examples
- Structures
- Educational Kits



Why Documentation?

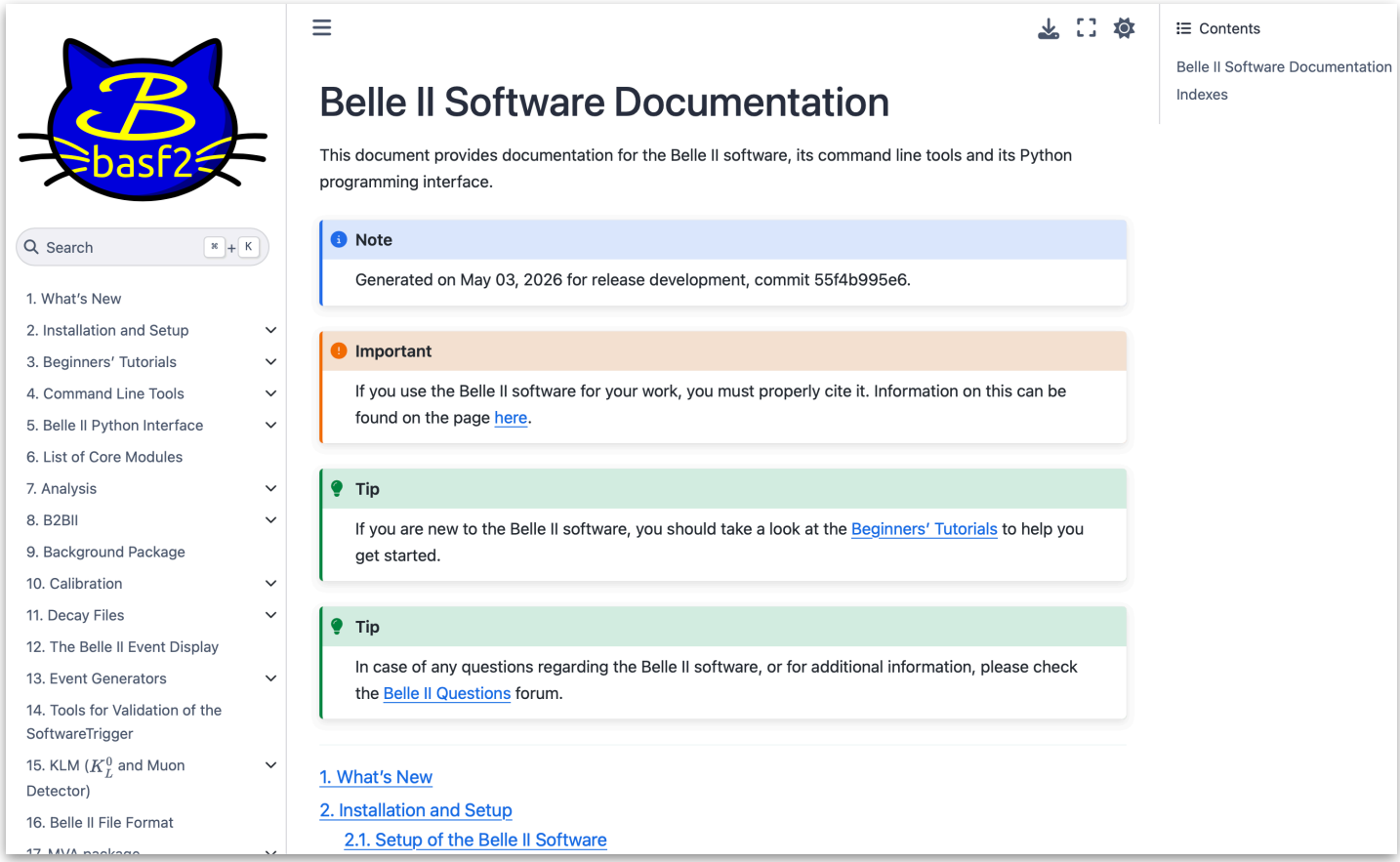
`If it is not documented, it does not exist`

What is documented?

- Code
- Examples
- Structures
- Educational Kits

What should be documented?

- What it does
- How to use it
- How it works



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Sphinx

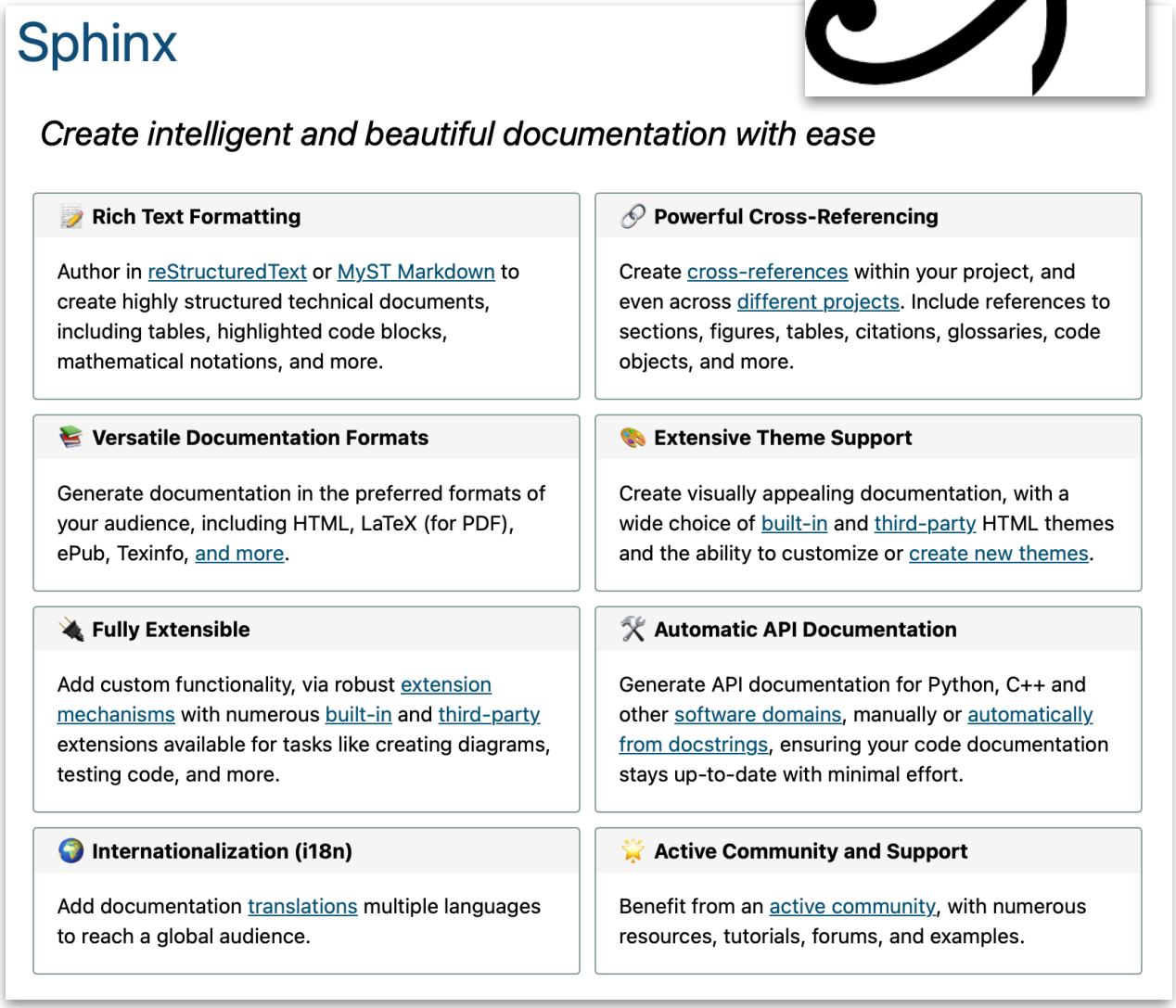
A tool that turns text + code into structured documentation websites

How does Sphinx work?

- Inline or separate files in `reStructuredText` or `Markdown` are collected and compiled into an output

Why Sphinx?

- Easy to learn (once setup is understood)
- Automatic API documentation
- Cross-referencing is easy
- Many different extensions



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

basf2 - API Documentation

RootInput

Reads objects/arrays from one or more .root files saved by the RootOutput module and makes them available through the DataStore. Files do not necessarily have to be local, http:// and root:// (for files in xrootd) URLs are supported as well.

Package:

framework

Library:

librootio.so

Parameters:

- **branchNames** *(list(str), default=[])*

  Names of event durability branches to be read. Empty means all branches. (EventMetaData is always read)

- **branchNamesPersistent** *(list(str), default=[])*

  Names of persistent durability branches to be read. Empty means all branches. (FileMetaData is always read)

- **cacheSize** *(int, default=0)*

  file cache size in Mbytes. If negative, use root default

basf2 - API Documentation

RootInput

Reads objects/arrays from one or more .root files saved by the RootOutput module and makes them available through the DataStore. Files do not necessarily have to be local, http:// and root:// (for files in xrootd) URLs are supported as well.

Package:

framework

Library:

librootio.so

Parameters:

- **branchNames** *(list(str), default=[])*

  Names of event durability branches to be read. Empty means all branches. (EventMetaData is always read)

- **branchNamesPersistent** *(list(str), default=[])*

  Names of persistent durability branches to be read. Empty means all branches. (FileMetaData is always read)

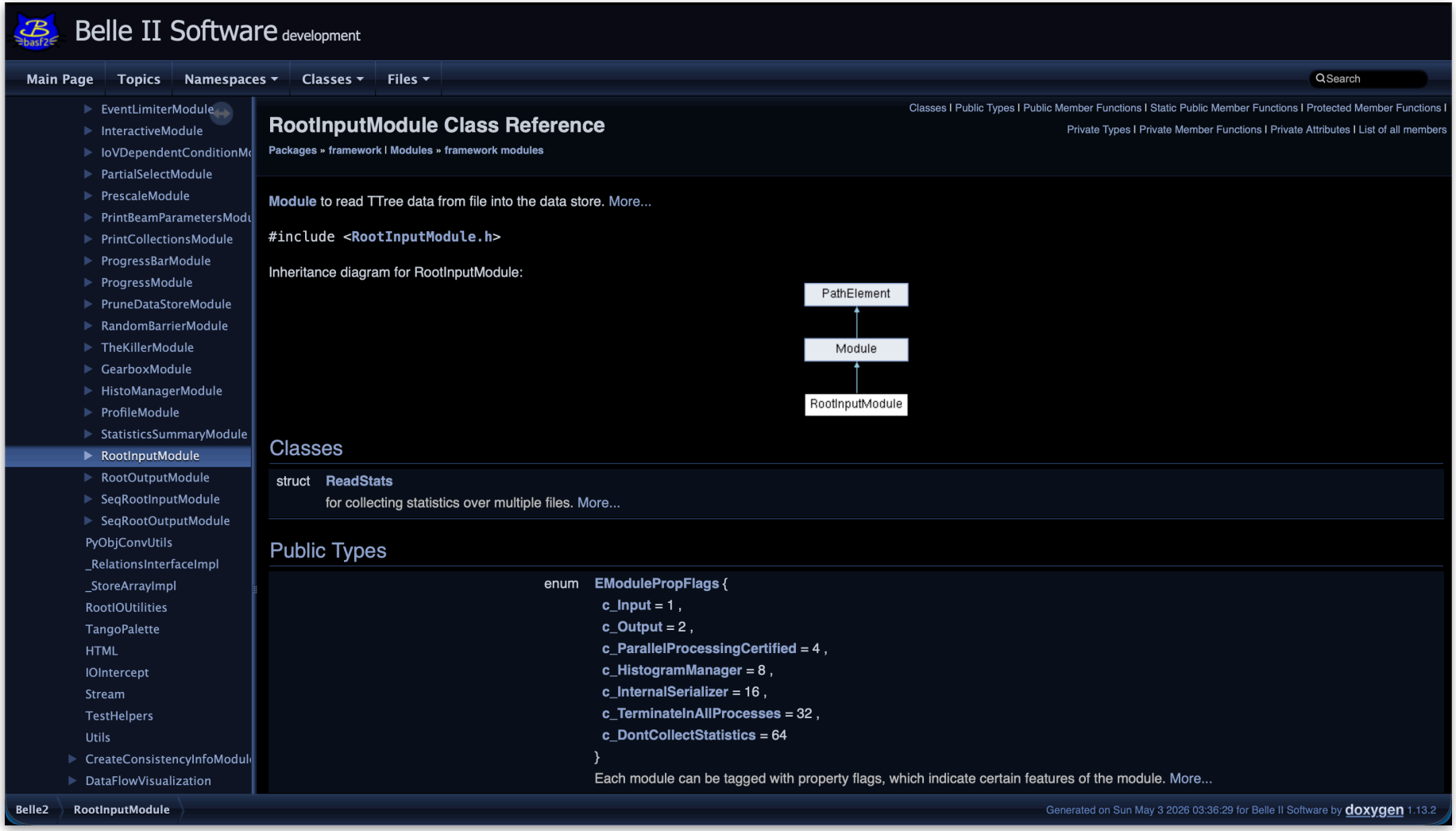
- **cacheSize** *(int, default=0)*

  file cache size in Mbytes. If negative, use root default

```
39 RootInputModule::RootInputModule() : Module(), m_nextEntry(0), m_lastPersistentEntry(-1), m_tree(nullptr), m_persistent(nullptr)
40 {
41   //Set module properties
42   setDescription("Reads objects/arrays from one or more .root files saved by the RootOutput
43      module and makes them available through the DataStore. Files do not necessarily have
44      to be local, http:// and root:// (for files in xrootd) URLs are supported as well.");
45   setPropertyFlags(c_Input);
46
47   //Parameter definition
48   vector<string> emptyvector;
49   addParam("cacheSize", m_cacheSize,
50            "file cache size in Mbytes. If negative, use root default", 0);
51   addParam("inputFileName", m_inputFileName,
52            "Input file name. For multiple files, use inputFileNames or wildcards instead.
53             Can be overridden using the -i argument to basf2.",
54            string(""));
```

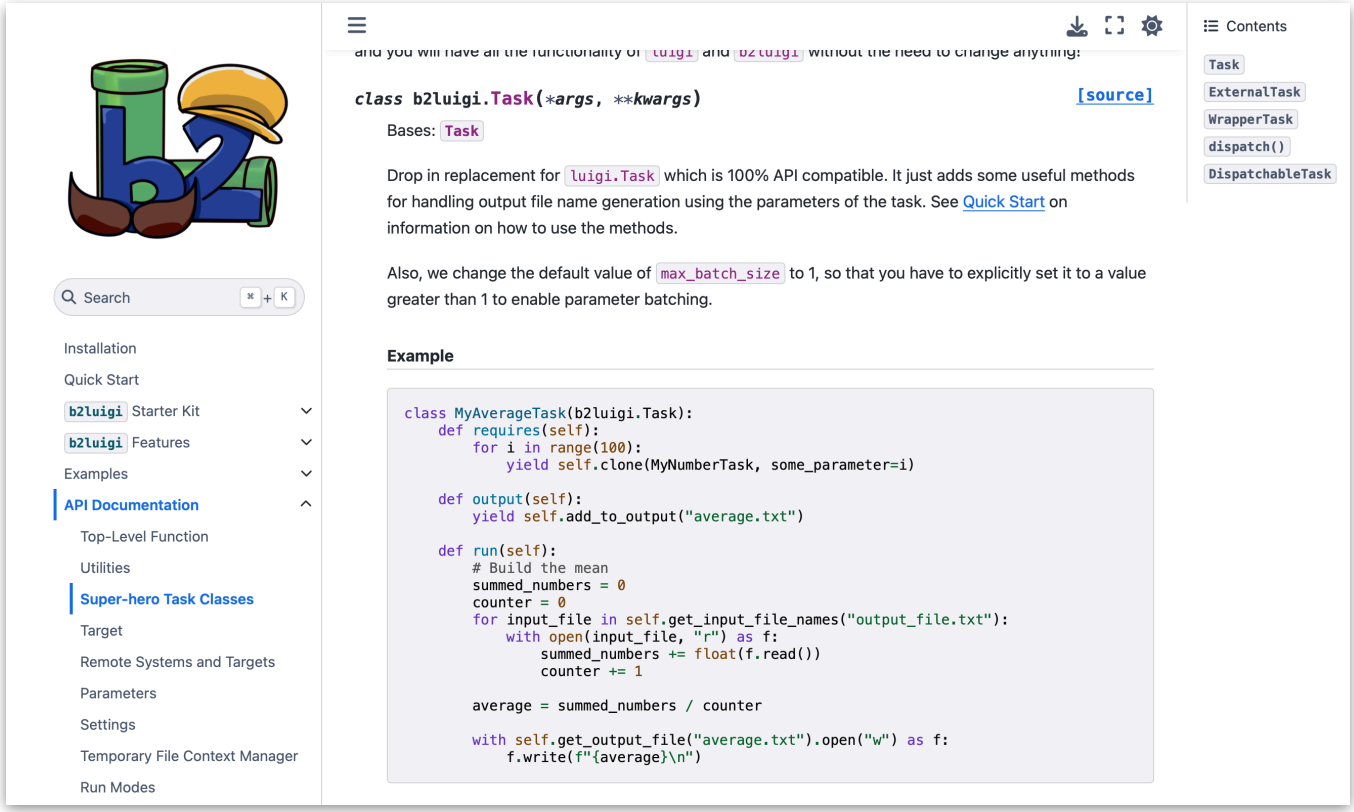
LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Doxygen



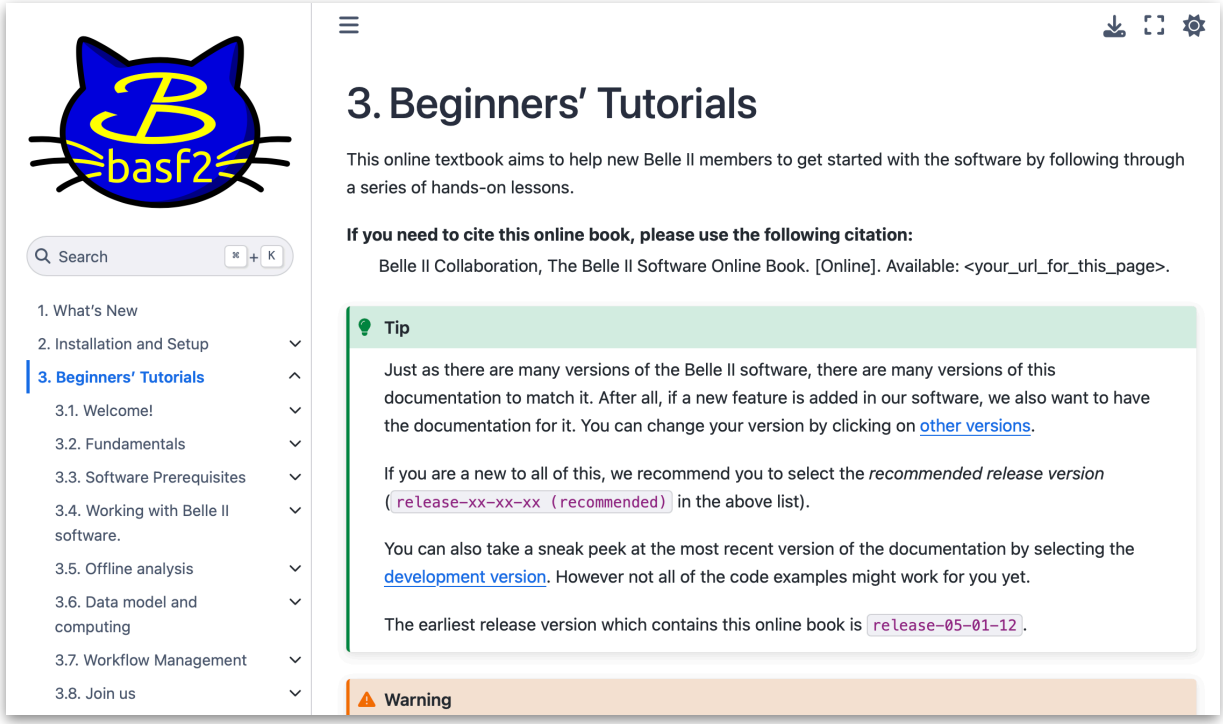
LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

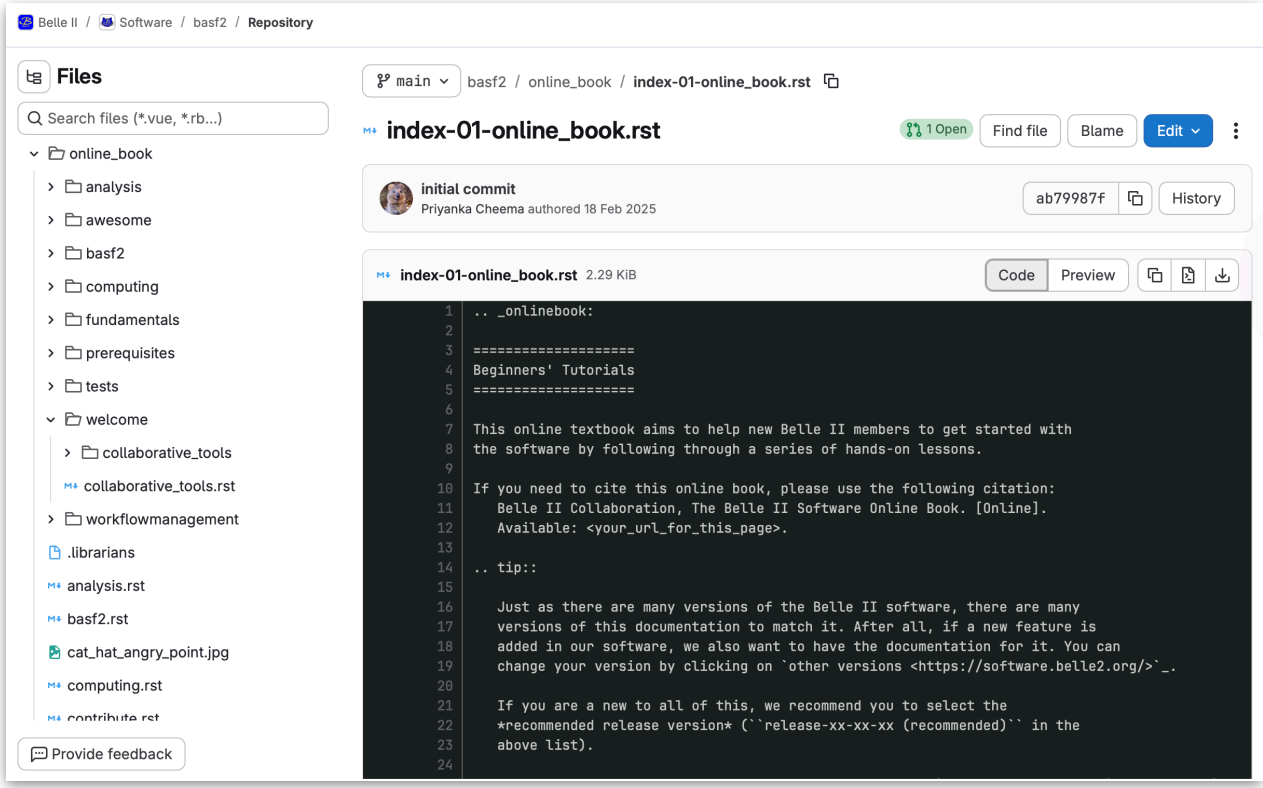
b2luigi - API documentation



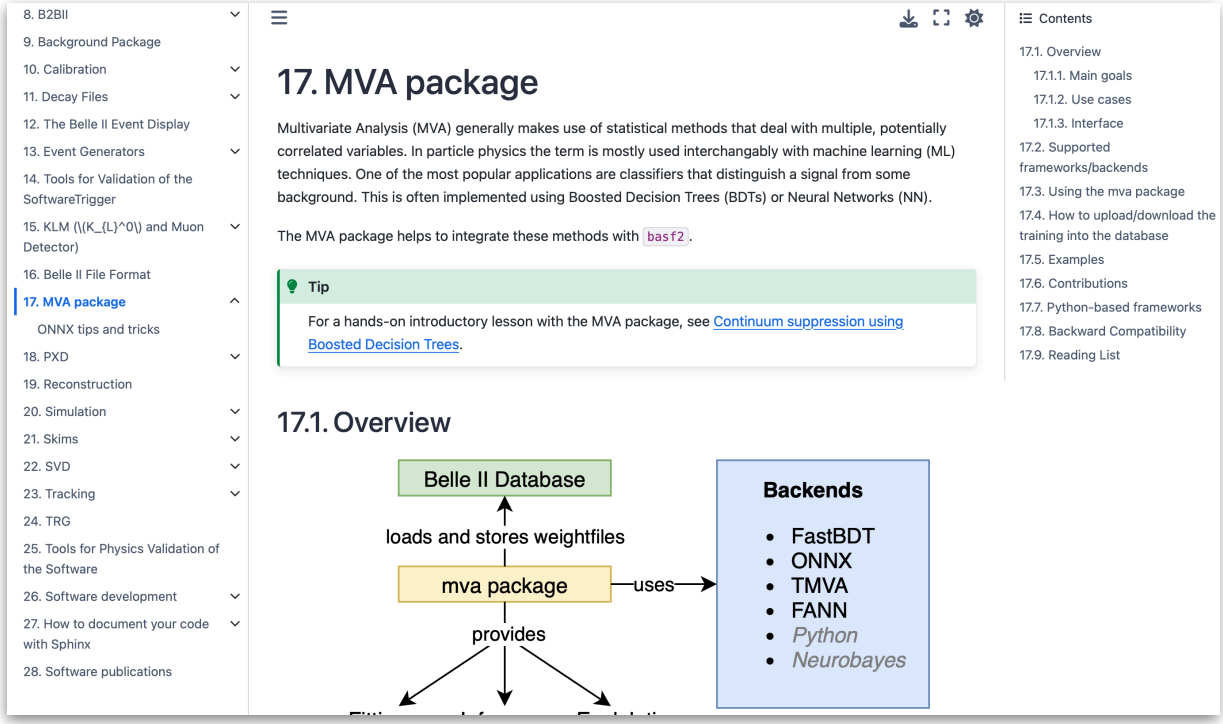

```python
class Task(luigi.Task):
    """
    Drop in replacement for ``luigi.Task`` which is 100% API compatible.
    It just adds some useful methods for handling output file name generation using
    the parameters of the task.
    See :ref:`quick-start-label` on information on how to use the methods.

    Also, we change the default value of ``max_batch_size`` to 1, so that you have
    to explicitly set it to a value greater than 1 to enable parameter batching.

    Example:

        .. code-block:: python

          class MyAverageTask(b2luigi.Task):
              def requires(self):
                  for i in range(100):
                      yield self.clone(MyNumberTask, some_parameter=i)

              def output(self):
                  yield self.add_to_output("average.txt")

              def run(self):
                  # Build the mean
                  summed_numbers = 0
                  counter = 0
                  for input_file in self.get_input_file_names("output_file.txt"):
                      with open(input_file, "r") as f:
                          summed_numbers += float(f.read())
                          counter += 1

                  average = summed_numbers / counter

                  with self.get_output_file("average.txt").open("w") as f:
                      f.write(f"{average}\\n")
    """

```

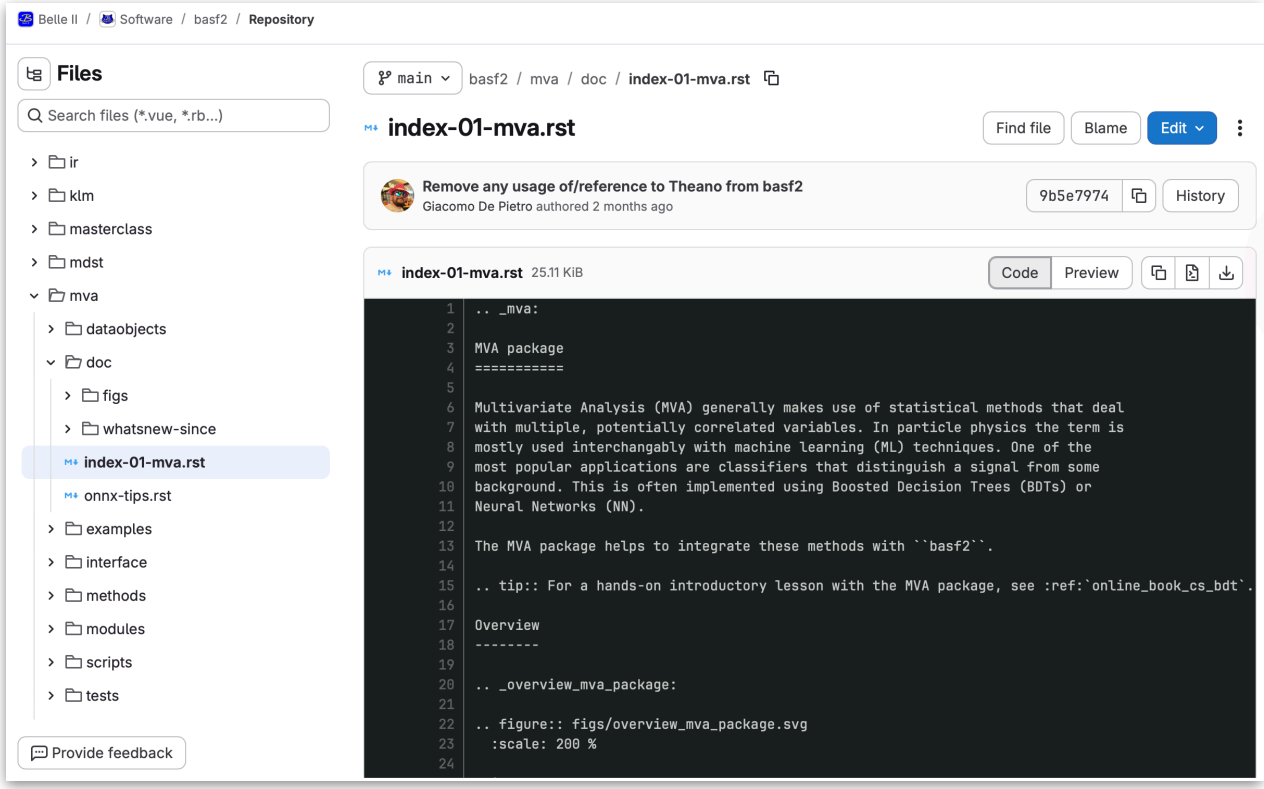
basf2 - Online Book



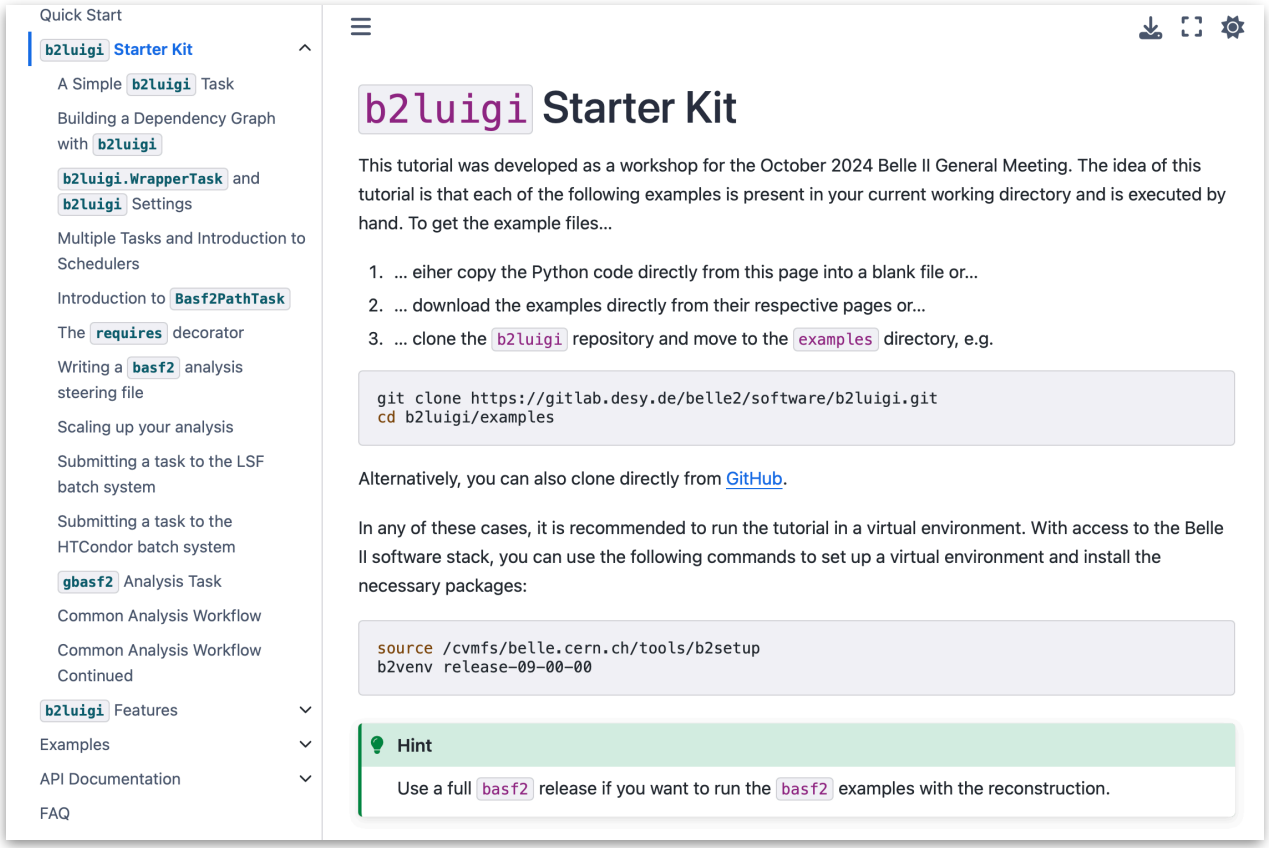


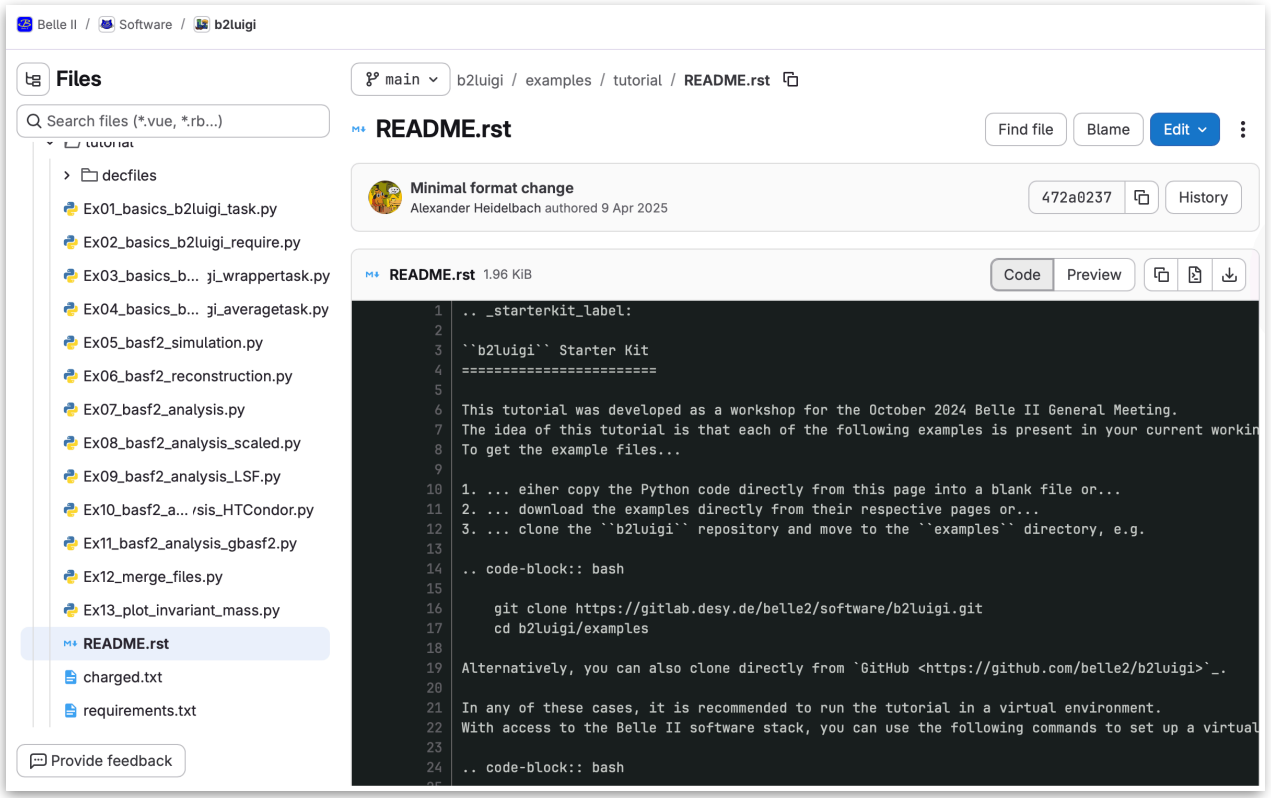
basf2 - Packages





Bonus: b2luigi - Starter Kit





Bonus: b2luigi - Starter Kit

Multiple Tasks and Introduction to Schedulers

Hint

This example demonstrates how a single task can require multiple tasks to complete. This is useful if the output of multiple tasks is needed to compute the output of the following task. Additionally, we introduce the core concepts of schedulers.

`b2luigi.WrapperTask` is not the only way to require multiple tasks. Also normal tasks can require multiple tasks. This is useful if the wrapper task uses the inputs of the required tasks to compute its output.

For this, we define the first and second task in the same way as in the previous example.

```python
import b2luigi


class MyTask(b2luigi.Task):
    parameter = b2luigi.IntParameter()

    def run(self):
        with open(self.get_output_file_name("output.txt"), "w") as f:
            f.write(f"{self.parameter}")

    def output(self):
        yield self.add_to_output("output.txt")


class MyOtherTask(b2luigi.Task):
    parameter = b2luigi.IntParameter()

    def requires(self):
        return MyTask(parameter=self.parameter)
```

```python
"""
.. _exercise04_label:

Multiple Tasks and Introduction to Schedulers
=============================================

.. hint::
    This example demonstrates how a single task can require multiple tasks to complete.
    This is useful if the output of multiple tasks is needed to compute the output of the following task.
    Additionally, we introduce the core concepts of schedulers.

:class:`b2luigi.WrapperTask` is not the only way to require multiple tasks.
Also normal tasks can require multiple tasks.
This is useful if the wrapper task uses the inputs of the required tasks to compute its output.

For this, we define the first and second task in the same way as in the previous example.
"""
import b2luigi


class MyTask(b2luigi.Task):
    parameter = b2luigi.IntParameter()

    def run(self):
        with open(self.get_output_file_name("output.txt"), "w") as f:
            f.write(f"{self.parameter}")

    def output(self):
        yield self.add_to_output("output.txt")


class MyOtherTask(b2luigi.Task):
    parameter = b2luigi.IntParameter()

    def requires(self):
        return MyTask(parameter=self.parameter)
```


HELP
WANTED
NEEDED