



# Continuous Integration



James M. Willenbring  
Sandia National Laboratories

David E. Bernholdt  
Oak Ridge National Laboratory

Mark C. Miller  
Lawrence Livermore National Laboratory

Developing a Testing and Continuous Integration Strategy for your  
Team tutorial @ ECP Annual Meeting, April 2021



See slide 2 for  
license details

# License, Citation and Acknowledgements

## License and Citation



- This work is licensed under a [Creative Commons Attribution 4.0 International License](#) (CC BY 4.0).
- **The requested citation the overall tutorial is: David E. Bernholdt, Patricia A. Grubel, and James M. Willenbring, Developing a Testing and Continuous Integration Strategy for your Team tutorial, in Exascale Computing Project Annual Meeting, online, 2021. DOI: [10.6084/m9.figshare.14376956](#)**
- Individual modules may be cited as *Speaker, Module Title*, in Better Scientific Software tutorial...

## Acknowledgements

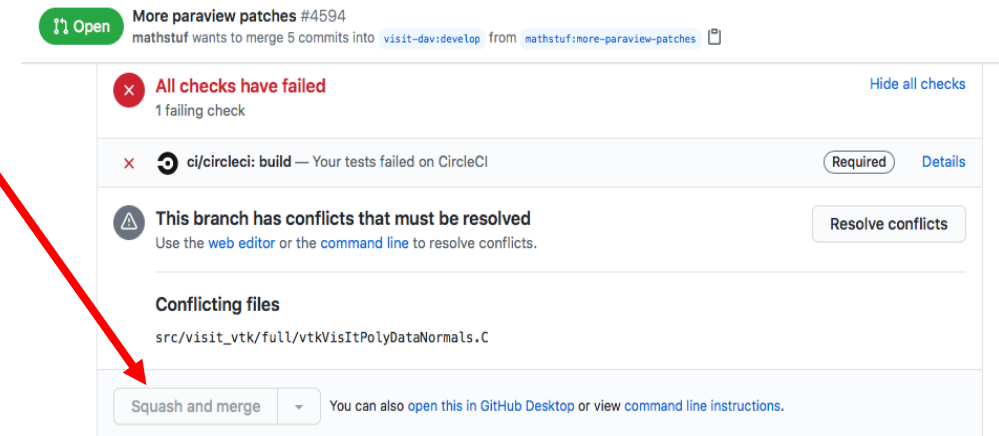
- Additional contributors include: David E. Bernholdt, Anshu Dubey, Rinku K. Gupta, Mike Heroux, Alicia Klinvex, Mark Miller, Jared O'Neal, Katherine Riley, David Rogers, Deborah Stevens, James Willenbring
- This work was supported by the U.S. Department of Energy Office of Science, Office of Advanced Scientific Computing Research (ASCR), and by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration.
- This work was performed in part at the Argonne National Laboratory, which is managed by UChicago Argonne, LLC for the U.S. Department of Energy under Contract No. DE-AC02-06CH11357.
- This work was performed in part at the Oak Ridge National Laboratory, which is managed by UT-Battelle, LLC for the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.
- This work was performed in part at the Lawrence Livermore National Laboratory, which is managed by Lawrence Livermore National Security, LLC for the U.S. Department of Energy under Contract No. DE-AC52-07NA27344.
- This work was performed in part at the Los Alamos National Laboratory, which is managed by Triad National Security, LLC for the U.S. Department of Energy under Contract No.89233218CNA000001
- This work was performed in part at Sandia National Laboratories. Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

# What is Continuous Integration (CI) *Testing*

- Testing
  - Focused, critical functionality (infrastructure), fast, independent, orthogonal, complete, ...
  - Existing test suites often require re-design/refactoring for CI
- Integration
  - Changes across key branches merged & tested to ensure the “whole” still works
    - Integration can take place at multiple levels
      - Individual project
      - Spack
      - E4S
  - Develop, develop, develop, merge, merge, merge, test, test, test...NO!
  - Develop, merge, test, develop, merge, test, develop, merge, test...YES!
- Continuous
  - Changes tested every commit and/or pull-request (like auto-correct)
- CI generally implies a lot of automation

# Automated Testing vs. Continuous Integration (CI) Testing

- **Automated Testing:** Software that automatically performs tests and reliably detects and reports anomalous behaviors/outcomes.
  - Examples: Auto-test, CTest/CDash, nightly testing, `make check`
  - Lives “next to” your development workflow
  - Potential issues: change attribution, timeliness of results, multiple branches of development
- **Continuous Integration (CI):** automated testing performed at high frequency and fine granularity aimed at *preventing* code changes from breaking key branches of development (e.g. *main*)
  - Example: Disabled/enabled “Merge Pull Request” button on GitHub
    - Can also be run post merge
  - Lives “within” your development workflow
  - Potential issues: extreme automation, test granularity, coverage, 3<sup>rd</sup>-party services/resources



# Examples...

## Automated Nightly Testing Dashboard Lives “next to” your development work


### Results of Visit Regression Test ( pascal,trunk,serial )

Test suite run started at 2020:07:09:22:49:46.  
(Click on table header to sort)

Index	Category	Test File	Status	Runtime (sec)
243	rendering	ospray.py	Unacceptable	5.0
273	simulation	batch.py	Unacceptable	38.0
24	databases	chgcarr.py	Succeeded With Skips	11.0
32	databases	exodus.py	Succeeded With Skips	14.0
66	databases	silos.py	Succeeded With Skips	50.0
67	databases	silos_altdriver.py	Succeeded With Skips	87.0
75	databases	xdmf.py	Succeeded With Skips	14.0
109	hybrid	merge_tree.py	Succeeded With Skips	11.0
136	meshtype	emptydomains.py	Succeeded With Skips	7.0
256	rendering	view.py	Succeeded With Skips	17.0
275	simulation	curve.py	Succeeded With Skips	8.0
281	simulation	life.py	Succeeded With Skips	8.0
296	simulation	zerocopy.py	Succeeded With Skips	32.0
0	databases	ANALYZE.py	Succeeded	10.0
1	databases	ANSYS.py	Succeeded	9.0
2	databases	CGNS.py	Succeeded	11.0
3	databases	Cale.py	Succeeded	6.0
4	databases	Chombo.py	Succeeded	7.0
5	databases	EnSight.py	Succeeded	9.0
6	databases	FITS.py	Succeeded	8.0
7	databases	Fluent.py	Succeeded	7.0
8	databases	GDAL.py	Succeeded	20.0
9	databases	NASTRAN.py	Succeeded	15.0


## CI Testing Lives embedded in your development work


Add more commits by pushing to the `exodus-patch-1` branch on `exodus/chromium-dashboard`.



✓ All checks have passed [Hide all checks](#)

2 successful checks


✓  **Lighthouse** — Passed. New Lighthouse score would be 100/100. [Details](#)

✓  **continuous-integration/travis-ci/pr** — The Travis CI build passed [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](#).

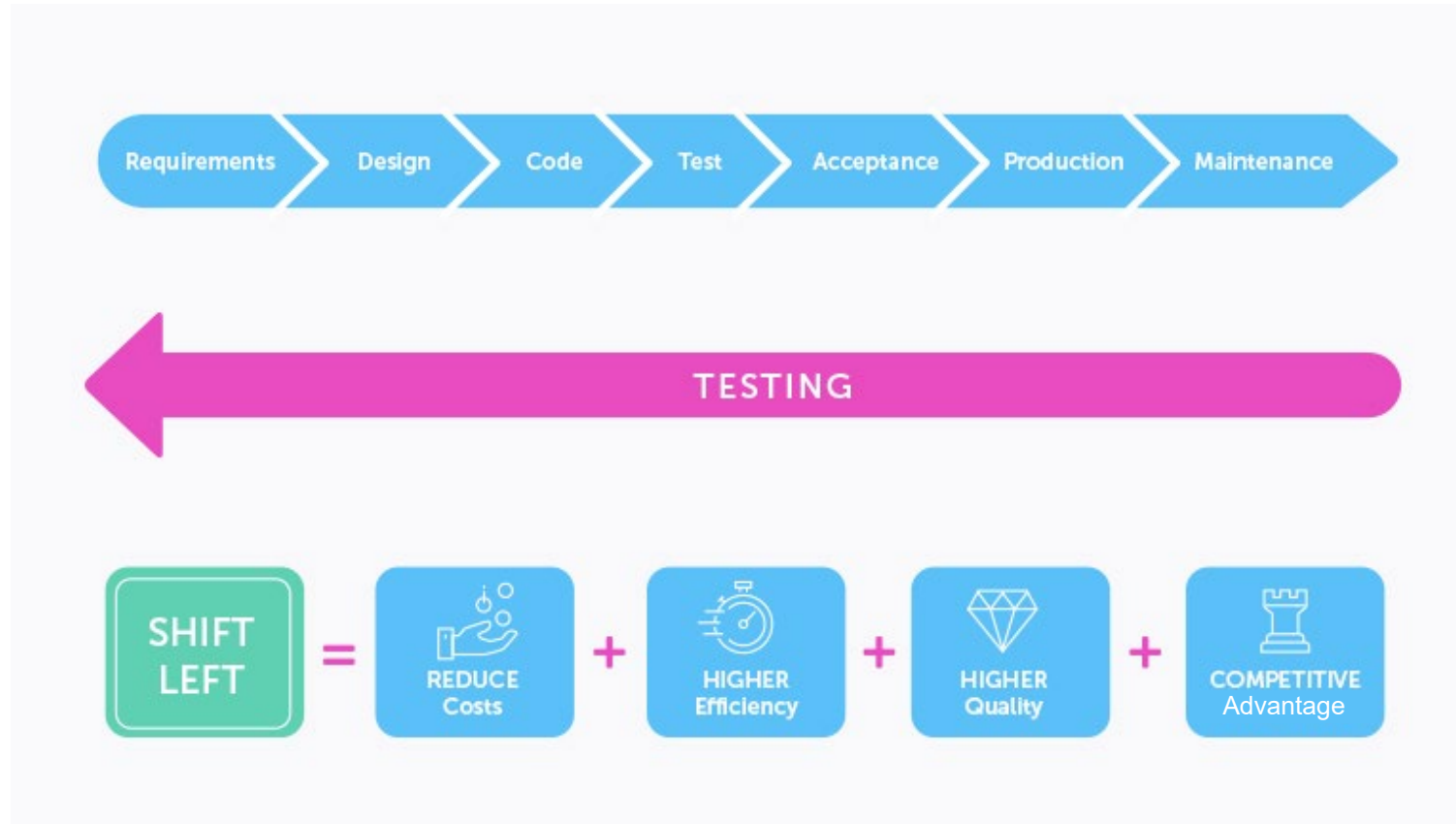


Write

Preview

AA B i “ < > ⌂ ⋮ ⋮ ⋮ ↶ @ 📌

# CI Testing is one part of the “Shift Left” movement in DevOps



# What can make CI Difficult

## Common situations

- Just getting started
  - Many technologies/choices; often in the "cloud"
  - Solution: start small, simple, build up
- Developing suitable tests
  - Many project's existing tests not suitable for CI
  - CI testing is a balance of thoroughness and responsiveness
  - Solution: Simplify/refactor and/or sub-setting test suite
- Ensuring sufficient coverage
  - Some changes to code never get tested – CI can provide a false sense of security
  - Solution: tools to measure it, enforce always increasing

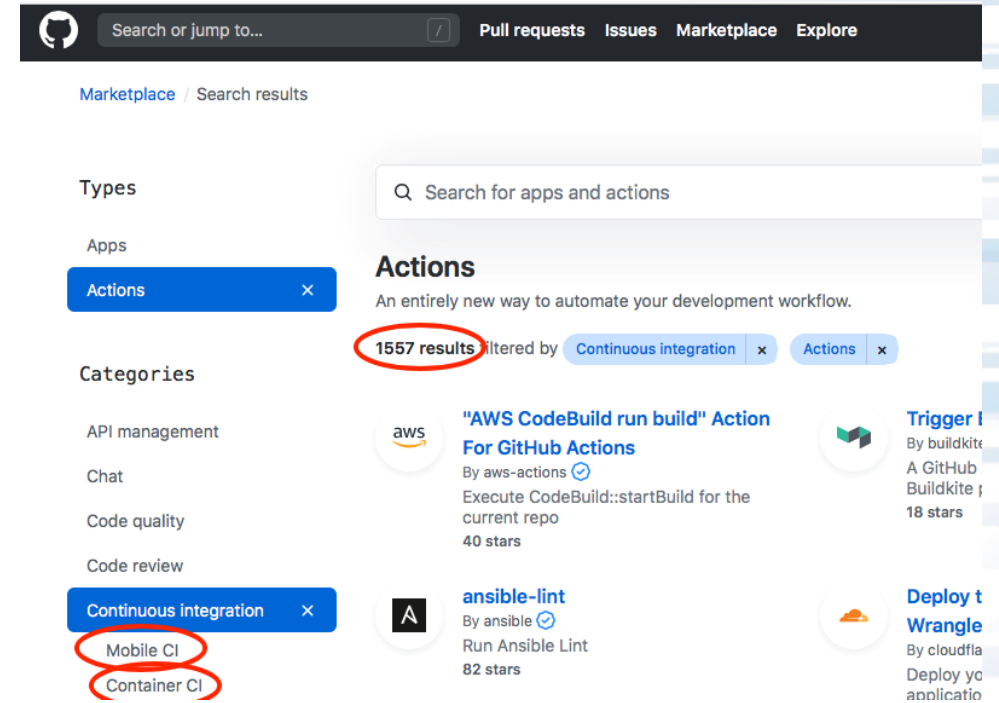
## Advanced situations

- Defining failure for *many* configurations / inconsistent failures
  - Bit-for-bit (exact) match vs. fuzzy match
  - Solution: absolute/relative tolerances → AI/ML
- Numerous 3<sup>rd</sup> party libraries (TPLs)
  - Compiling takes too long
  - Solution: cache pre-built TPLs, containers
- Performance testing
  - Avoid time-, space-, scaling-performance degradation
  - Solution: Perf. instrumentation and *scheduled* testing



# CI Resources (Where do jobs run?)

- Free Cloud Resources (many free on GitHub, BitBucket, GitLab, etc.)
  - Travis-CI, Circle-CI, AppVeyor, Azure Pipelines,...
  - All launch a VM (Linux variants, Windows and OSX)
    - Constrained in time/size, hardware (e.g. GPU type/count)
    - Not a complete solution for many HPC/scientific codes, but a useful starting point.
- Site-local Resources
  - Group, department, institution, computing facility
  - Examples: Bamboo @ LLNL, Jenkins @ ANL, Travis+CDash @ NERSC, etc.
  - ECP Program: GitLab-CI @ ANL, LANL, LLNL, NERSC, ORNL, SNL
- Create your own by setting up resources/services

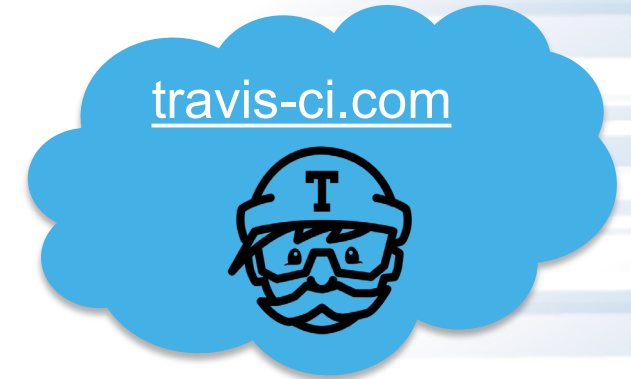




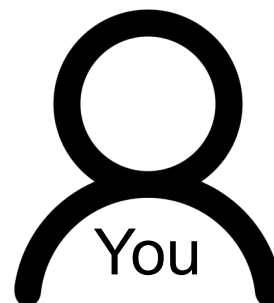
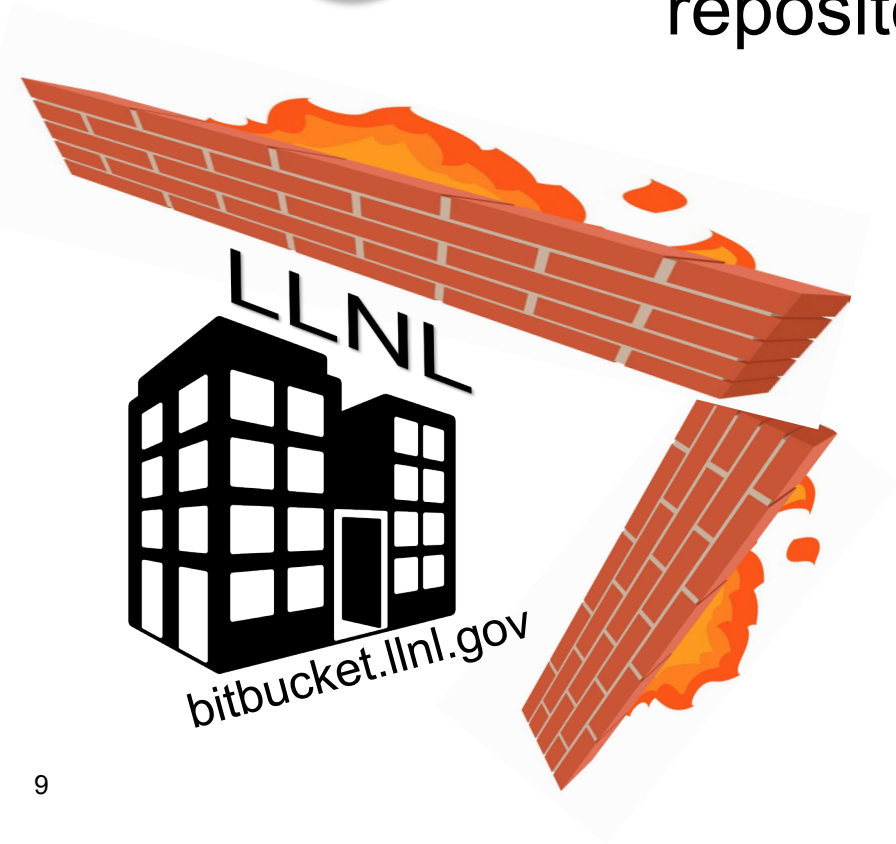
# Examples...



Your code repository

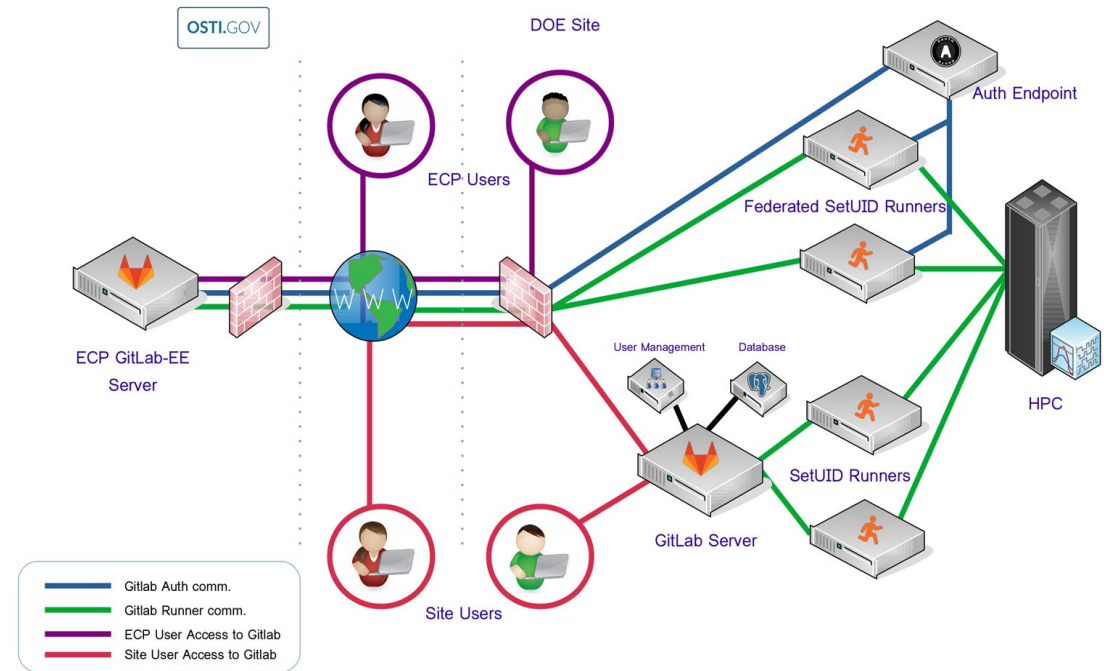


Your CI Resources



# ECP CI Resources

- ECP investing in GitLab for complex-wide CI
- Complex-wide Federation
  - Many hurdles still to overcome
- More about ECP CI at the Annual Meeting
  - **Benefitting From ECP CI**
    - Wednesday 2:30-3:30 Eastern
  - **ECP CI Startup Tutorial**
    - Friday 2:30-6 Eastern
- Documentation and on-boarding help
  - <https://ecp-ci.gitlab.io>



# Getting started with CI

- What *configuration* is most important?
  - Examples: gcc, icc, xlc? MPI-2 or MPI-3? Python 2, 3 or 2 & 3?
- What *functionality* is most important?
  - Examples: vanilla numerical kernels? OpenMP kernels? GPU kernels? All of these?
- Good candidates...
  - A “hello world” example for your project
  - At a minimum, even just building the code can be a place to start!
  - Once you’ve got the basics working, its easy to build up from there

# Getting started with CI:

## Setting up CI

Service	Interface	
Travis	repo YAML file [& repo scripts]	/.travis.yml in root of repo
GitLab	Web page configurator + repo YAML file [& repo scripts]	/.gitlab-ci.yml in root of repo
Bamboo	Web page configurator + repo scripts	
GitHub Actions	Repo YAML file	.github/workflows/<test_name>.yml
.		

Keywords defined by service provider's YAML docs

## Example .travis.yml file (also doing coverage analysis)

markcmiller86 /

hello-numerical-world-atpesc-2020

forked from betterscientificsoftware/hello-numerical-world-atpesc-2020

<> Code

Pull requests

Actions

Projects

Wiki

Security

main

hello-numerical-world-atpesc-2020 / .travis.yml

markcmiller86

Create .travis.yml ✓

Latest commit ad0c96b 2 h

1 contributor

5 lines (3 sloc) | 49 Bytes

Raw

B

1 language: c++

2

3 compiler: gcc

4

5 script: make check

} Specify environment

} Commands to run test

# Getting started with Travis CI:

10 lines (7 sloc) | 166 Bytes

```
1  language: c++
2
3  compiler:
4    - gcc
5
6  script:
7    - make CXXFLAGS=--coverage LDFLAGS="--coverage -lm" check_all
8
9  after_success:
10    - bash <(curl -s https://codecov.io/bash)
```

# travis-ci.com

# codecov.io

Travis CI

[Dashboard](#)[Changelog](#)[Documentation](#)[Help](#)

Search all repositories

markcmiller86 / hello-numerical-world

Current

Branches

Build History

Pull Requests

My Repositories

Running (1/2)

spack/spack

# 47315

Duration: 14 sec

mfem/mfem

# 8441

Duration: 1 hr 38 min 44 sec

Finished: 2 hours ago

markcmiller86/hello-numerical-world

# 7

Duration: 19 sec

Finished: 3 hours ago

beterscientificsoftware/Trust

# 2

Duration: 26 sec

Finished: 20 hours ago

LLNL/MACSIo

# 152

Duration: 1 min 24 sec

Finished: 2 days ago

beterscientificsoftware/bssw

# 83

Duration: 32 sec

Finished: 13 days ago

spack/spack-tutorial

# 125

Duration: 1 min 17 sec

Finished: 26 days ago

LLNL/lor

Duration: -

LLNL/FASTMath4

Duration: -

beterscientificsoftware/bssw

Duration: -

main

fix error threshold

Commit 26d69cd

Compare d24c2f3...26d69cd

Branch main

Mark C. Miller

Compiler: gcc C++

AMD64

Job log

View config

1 Worker information

6

7 Build system information

158

159

160 \$ git clone --depth=50 --branch=main https://github.com/markcmiller86/hello-numerical-world

170

171 \$ export TRAVIS\_COMPILER=gcc

172 \$ export CXX=\${CXX:-g++}

173 \$ export CXX\_FOR\_BUILD=\${CXX\_FOR\_BUILD:-g++}

174 \$ export CC=\${CC:-gcc}

175 \$ export CC\_FOR\_BUILD=\${CC\_FOR\_BUILD:-gcc}

176 \$ gcc --version

177 gcc (Ubuntu 5.4.0-6ubuntu1-16.04.11) 5.4.0 20160609

178 Copyright (C) 2015 Free Software Foundation, Inc.

179 This is free software; see the source for copying conditions. There is NO

180 warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

181

182 \$ make CXXFLAGS=-coverage LDFLAGS=-coverage -lm check

183 g++ -c --coverage heat.C -o heat.o

184 g++ -c --coverage utils.C -o utils.o

185 g++ -c --coverage args.C -o args.o

186 g++ -c --coverage exact.C -o exact.o

187 g++ -c --coverage ftcs.C -o ftcs.o

188 g++ -c --coverage upwind15.C -o upwind15.o

189 g++ -c --coverage crankn.C -o crankn.o

190 g++ -o heat heat.o utils.o args.o exact.o ftcs.o upwind15.o crankn.o --coverage -lm -lm

ghmarkcmiller86hello-numerical-worldDocsSupportBlog

fix error threshold

markcmiller86

3 hours ago

CI Passed

26d69cd

main

d24c2f3

51.60%

Files

Coverage

Double.H

65.63%

args.C

82.05%

crankn.C

0.00%

exact.C

0.00%

ftcs.C

100.00%

heat.C

73.81%

upwind15.C

0.00%

utils.C

49.35%

Project Totals (8 files)

51.60%

CALECULATINGECT

14

# GitHub Actions – results of workflow test runs

## Workflows

All workflows

- (TEST) Pyomo Windows Tests ...
- (WIP) Pyomo Windows Test (P...
- (WIP) Pyomo Windows Test (P...
- (WIP) Pyomo Windows Tests (...)
- (WIP) Windows Pip Cmd Pyom...
- GitHub Branch CI
- GitHub CI
- Pyomo Release Distribution Cr...
- Python package
- Ubuntu Pyomo Single Python ...
- Ubuntu Pyomo Workflow (Slim,...

## GitHub CI

Showing runs from all workflows named GitHub CI

event:push workflow:"GitHub CI"

461 workflow run results

Event Status Branch Actor

✓ Merge pull request #1902 from jsiirola/fix-unittest-rc GitHub CI #121: Commit 901b487 pushed by blnicho	main	20 hours ago 1h 3m 55s	...
✓ Merge pull request #1901 from mrmundt/remove-six GitHub CI #117: Commit a101b6d pushed by mrmundt	main	2 days ago 1h 3m 12s	...
✓ Merge pull request #1896 from jsiirola/abstract-disa... GitHub CI #112: Commit 1f9dd19 pushed by jsiirola	main	2 days ago 1h 5m 39s	...
✗ Merge pull request #1898 from mrmundt/py-unittest GitHub CI #109: Commit 1beb848 pushed by michaelbynum	main	3 days ago 1h 3m 33s	...
✓ Merge pull request #1893 from jsiirola/config-enum GitHub CI #105: Commit 9aa1186 pushed by jsiirola	main	3 days ago 1h 11m 3s	...



# GitHub Pull Request Status Indicators

Filters ▾

🔍

is:pr is:open

🏷️ Labels 26

📌 Milestones 3

New pull request

🔗 17 Open ✓ 1,018 Closed

Author ▾Label ▾Projects ▾Milestones ▾Reviews ▾Assignee ▾Sort ▾

🔗 Allow string args to external fuctions ✖

#1904 opened 19 hours ago by eslickj • Review required

❗ 1💬 1

🔗 Port Cloning ✓

#1899 opened 3 days ago by michaelbynum • Approved

❗ 1💬 3

🔗 Remove pyomo.solvers.tests.core ✓

#1897 opened 3 days ago by mrmundt • Approved

💬 2

🔗 [WIP] Model partition package ●

#1895 opened 4 days ago by rahuljoglekar47 • Review required

💬 1

# GitHub Actions – Key Elements of Defining Tests

```
1  name: GitHub CI
2
3  on:
4    push:
5      branches:
6        - main
7    pull_request:
8      branches:
9        - main
10 workflow_dispatch:
11   inputs:
12     git-ref:
13       description: Git Hash (Optional)
14       required: false
15
```

← Name

← Trigger

# GitHub Actions – Key Elements of Defining Tests

```
35 jobs:
36   build:
37     name: ${ matrix.TARGET }/${ matrix.python }${ matrix.other }
38     runs-on: ${ matrix.os }
39     timeout-minutes: 90
40     strategy:
41       fail-fast: false
42     matrix:
43       os: [ubuntu-18.04, macos-latest, windows-latest]
44       python: [3.6, 3.7, 3.8, 3.9, pypy3]
45       other: [""]
46       category: ["nightly"]
47       # Ubuntu-18.04 should be replaced with ubuntu-latest once PyNumero
48       # build error is resolved:
49       # https://github.com/Pyomo/pyomo/issues/1710
50
51     include:
52     - os: ubuntu-18.04
53       TARGET: linux
54       PYENV: pip
55
56     - os: macos-latest
57       TARGET: osx
58       PYENV: pip
```

← Job setup

# GitHub Actions – Key Elements of Defining Tests

```
111 steps:
112 - name: Checkout Pyomo source
113   uses: actions/checkout@v2
114
115 - name: Configure job parameters
116   run: |
117     JOB="${{matrix.TARGET}}/${{matrix.python}}${{matrix.other}}"
118     echo "GHA_JOBNAME=$JOB" | sed 's|/|_|g' >> $GITHUB_ENV
119     if test -z "${{matrix.other}}"; then
120       echo "GHA_JOBGROUP=${{matrix.TARGET}}" >> $GITHUB_ENV
121     else
122       echo "GHA_JOBGROUP=other" >> $GITHUB_ENV
123     fi
124     # Note: pandas 1.0.3 causes gams 29.1.0 import to fail in python 3.8
125     PYTHON_PACKAGES="${PYTHON_REQUIRED_PKGS}"
126     if test -z "${{matrix.slim}}"; then
127       PYTHON_PACKAGES="$PYTHON_PACKAGES ${PYTHON_BASE_PKGS}"
128     fi
129     if [[ ${{matrix.python}} != pypy* && ! "${{matrix.slim}}" ]]; then
130       # NumPy and derivatives either don't build under pypy, or if
131       # they do, the builds take forever.
132       PYTHON_PACKAGES="$PYTHON_PACKAGES ${PYTHON_NUMPY_PKGS}"
133     fi
134     PYTHON_PACKAGES="$PYTHON_PACKAGES ${matrix.PACKAGES}"
135     echo "PYTHON_PACKAGES=$PYTHON_PACKAGES" \
136       | tr '\n' ' ' | sed 's/ \+/ /g' >> $GITHUB_ENV
```

← Job steps

# GitHub Actions - Release Automation

✓ Merge pull request #1807 from Pyomo/finalize-release Pyomo Release Distribution Creation #9

Summary

- Jobs
- ✓ manylinux/wheel\_creation
  - ✓ generic\_tarball
  - ✓ osxpy2.7/wheel\_creation
  - ✓ osxpy3.5/wheel\_creation
  - ✓ osxpy3.6/wheel\_creation
  - ✓ osxpy3.7/wheel\_creation
  - ✓ osxpy3.8/wheel\_creation
  - ✓ osxpy3.9/wheel\_creation
  - ✓ winpy3.6/wheel\_creation
  - ✓ winpy3.7/wheel\_creation
  - ✓ winpy3.8/wheel\_creation
  - ✓ winpy3.9/wheel\_creation

Triggered via push 2 months ago

jsirola pushed 46e336e 5.7.3

Status

Success

Total duration

49m 30s

Artifacts

4

release\_wheel\_creation.yml

on: push

Matrix: generictarball

✓ 1 job completed

Show all jobs

Matrix: manylinux

✓ 1 job completed

Show all jobs

Matrix: osx





✓ 6 jobs completed

Show all jobs

Matrix: windows

✓ 4 jobs completed

Show all jobs

Artifacts	
Produced during runtime	
Name	Size
 generictarball	2.11 MB
 manylinux-wheels	117 MB
 osx-wheels	28.9 MB
 win-wheels	18.5 MB

# Summary

- The purpose of Continuous Integration Testing is to identify problems early
  - Catch things that would “break the build” or adversely impact other developers
  - Need to provide sufficient confidence, but run quickly – balance varies by project
- CI testing should complement (not replace) more extensive automated “nightly” testing
  - Use scheduled testing for more and more detailed tests, more configurations and platforms, performance testing, etc.
- Many options for where to execute CI tests
  - Free services are a good (easy) place to start
  - But may not be sufficient in the long run (especially large HPC/scientific codes)
- Start simple to get automation working, then build out what you need
  - Focus initially on key software configurations and aspects of the code
  - Make sure your testing expands to cover new code

# Agenda

Time (EDT)	Module	Topic	Speaker
2:30-2:35pm	00	Introduction	David E. Bernholdt, ORNL
2:35pm-2:40pm	01	Motivation and Overview	Patricia A. Grubel, LANL
2:40pm-3:00pm	02	Software Testing 1	Patricia A. Grubel, LANL
3:00pm-3:25pm	03	Software Testing 2	David E. Bernholdt, ORNL
3:25pm-3:55pm	04	Continuous Integration	James M. Willenbring, SNL
3:55pm-4:00pm	05	Summary	James M. Willenbring, SNL