❯ DevOpsDays Fortaleza 2025

CI em projetos open-source

Testando e entregando 85+ libs mensalmente

Emídio Neto

Quem sou eu

Maintainer OpenTelemetry Docs PT-BR

Approver OpenTelemetry Python SDK

Organizador do Cloud Native RN

Emídio Neto

Agenda

1. Por que testar múltiplas versões ?
2. Como o Tox ajuda (ou não) + GHA
3. Releases + Backport
4. Desafios e lições aprendidas
5. Q&A

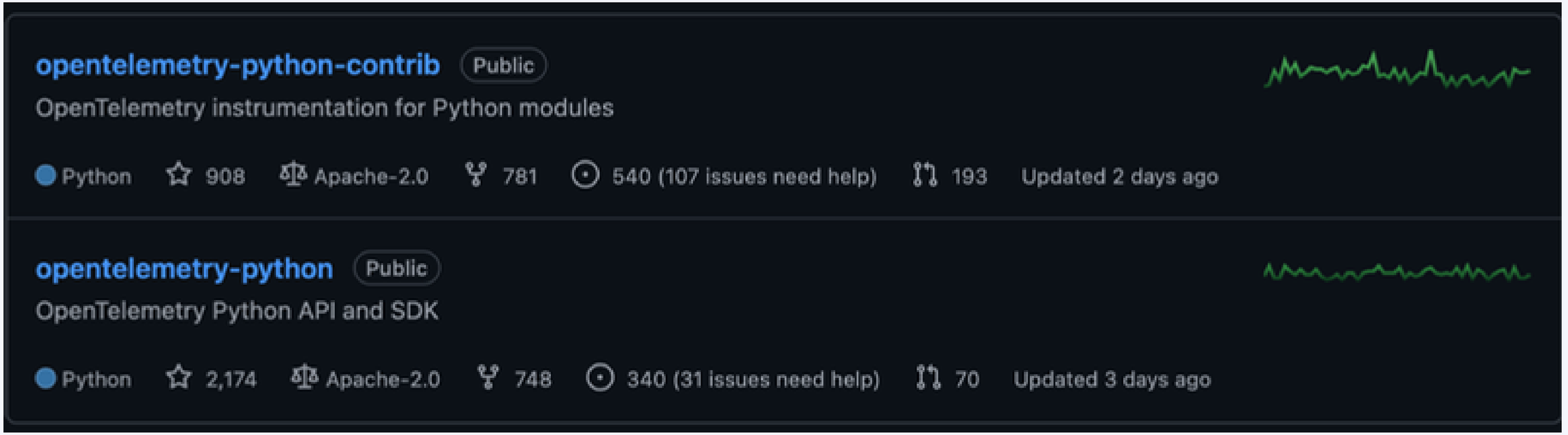
Por que testar múltiplas versões ?

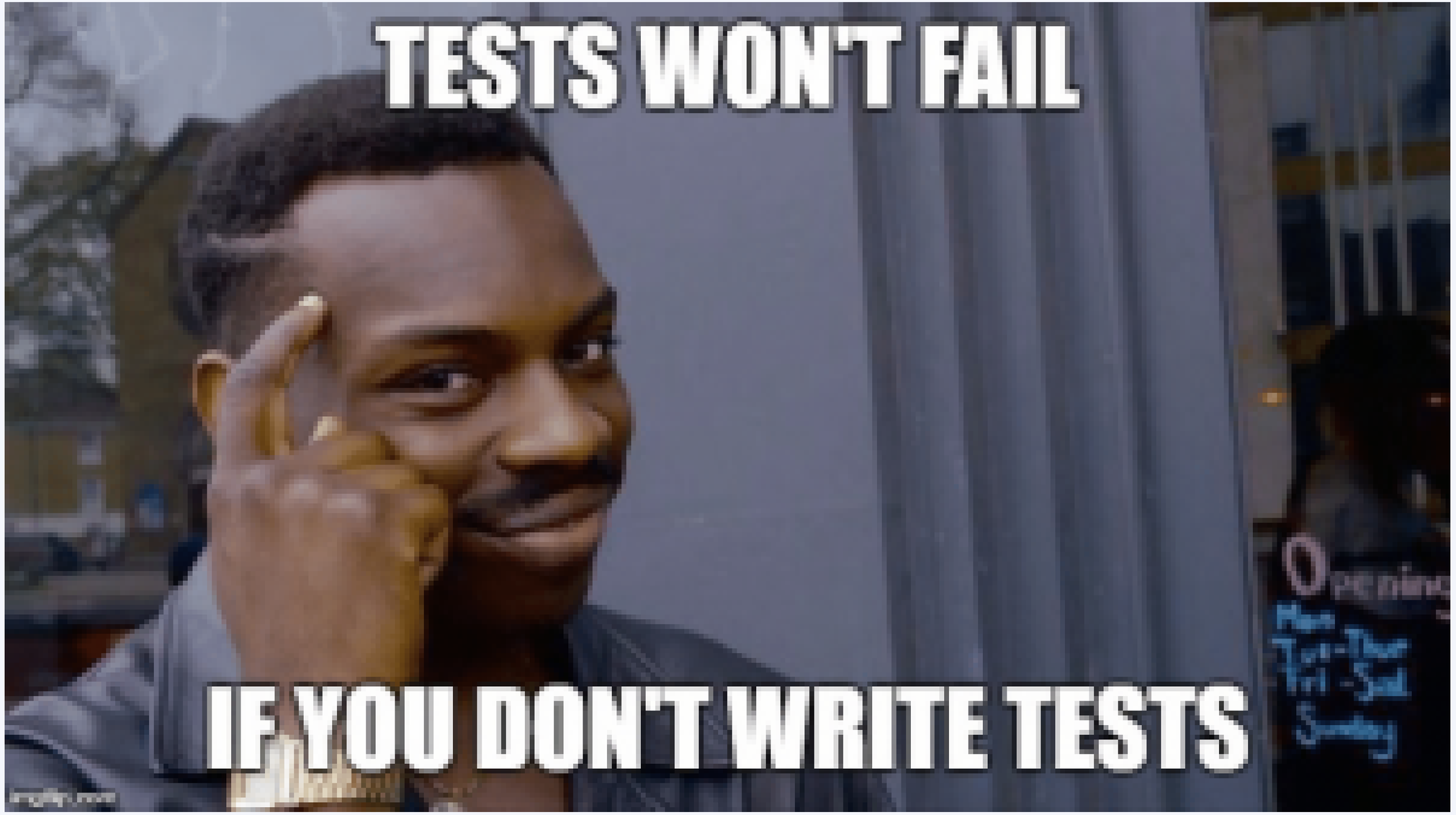
A integração contínua (CI) é o processo automatizado de build, teste e validação de cada alteração de código para garantir que o projeto permaneça estável e consistente ao longo do tempo.

Por que testar múltiplas versões ?

1. Garantir compatibilidade
2. Preparação para o futuro e detecção de depreciações
3. Requisitos específicos a depender do ambiente/cliente

2 monorepos
Repo principal e
repo de contrib (instrumentações específicas)




TESTS WON'T FAIL
IF YOU DON'T WRITE TESTS

PEP 585 – Built-in Generic Types (Python 3.9)

A partir do Python 3.9, você pode usar os tipos integrados diretamente (por exemplo, list[int]), o que pode quebrar o código que depende do estilo antigo.

snippet.py

```python
def double_numbers(nums: list[int]) -> list[int]:
    return [n * 2 for n in nums]
```

snippet

```
$ python3.8 snippet.py

Traceback (most recent call last):
  File "snippet.py", line 1, in <module>
    def double_numbers(nums: list[int]) -> list[int]:
TypeError: 'type' object is not subscriptable
```

“

PEP 604 – Allow writing union types as X | Y

No Python 3.9, embora você pudesse usar tipos integrados (como list[int]) graças ao PEP 585, os tipos union ainda precisam da sintaxe “typing.Union” se você quiser compatibilidade com versões anteriores (<3.10).

snippet

```python
def get_items(data: list[int] | None) -> list[str] | None:
    if data is None:
        return None
    return [str(x) for x in data]
```

snippet

```
$ python3.9 snippet.py
Traceback (most recent call last):
  File "snippet.py", line 1, in <module>
    def get_items(data: list[int] | None) -> list[str] | None:
TypeError: unsupported operand type(s) for |: 'types.GenericAlias' and 'NoneType'
```

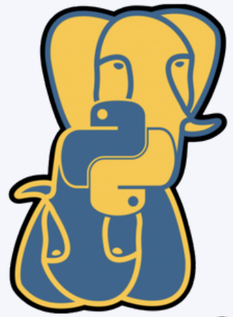
Breaking changes que afetam o serviço em runtime não são legais

Uma boa prática é rodar testes para todas versões Python que sua biblioteca suporta

Matriz de Testes

Static Type Checkers

Pacotes populares fazem isso



psycopg2

opentelemetry-python Public

18 Python Packages

+

opentelemetry-python-contrib Public

69 Python Packages

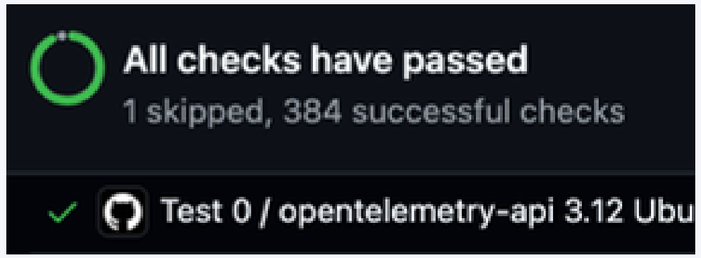
opentelemetry-python Public

~300+ Jobs na CI

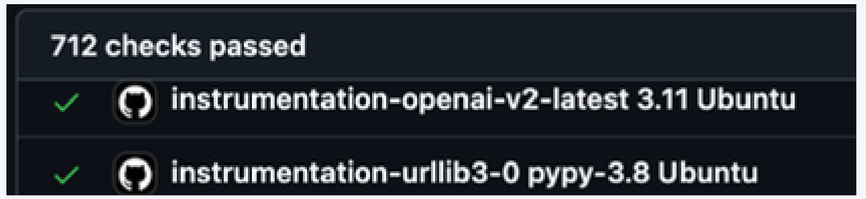
+

opentelemetry-python-contrib Public

~700+ Jobs na CI



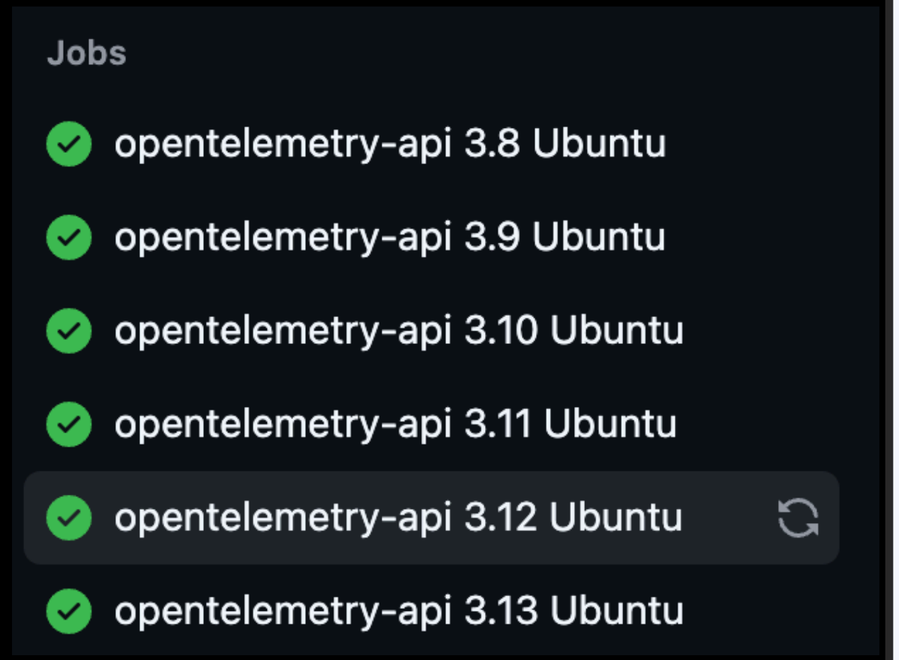
+ 1000 Jobs



Testes

Considere a seguinte matriz de testes:

package: [”opentelemetry-api”]
OS: [”Ubuntu”]
python: [”3.8", “3.9”, “3.10”, “3.11”, “3.12”, “3.13”]



Para uma biblioteca parece OK

mas e pra 80?

Como o Tox ajuda (ou não) + GHA

“

Tox automatiza e padroniza testes em Python.
Como um orquestrador

https://tox.wiki

x

6 versões Python
+ pypy3 + lint

```
[tox]
isolated_build = True
skipsdist = True
skip_missing_interpreters = True
envlist =
    ; Environments are organized by individual package, allowing
    ; for specifying supported Python versions per package.

    py3{8,9,10,11,12,13}-test-opentelemetry-api
    pypy3-test-opentelemetry-api
    lint-opentelemetry-api

    py3{8,9,10,11,12,13}-test-opentelemetry-proto-protobuf5
    pypy3-test-opentelemetry-proto-protobuf5
    lint-opentelemetry-proto-protobuf5

    py3{8,9,10,11,12,13}-test-opentelemetry-sdk
    pypy3-test-opentelemetry-sdk
    lint-opentelemetry-sdk
    benchmark-opentelemetry-sdk

    py3{8,9,10,11,12,13}-test-opentelemetry-semantic-conventions
    pypy3-test-opentelemetry-semantic-conventions
    lint-opentelemetry-semantic-conventions
```

```
[testenv]
deps =
  lint: -r dev-requirements.txt
  coverage: pytest
  coverage: pytest-cov

  mypy,mypyinstalled: -r {toxinidir}/mypy-requirements.txt

  api: -r {toxinidir}/opentelemetry-api/test-requirements.txt

  sdk: -r {toxinidir}/opentelemetry-sdk/test-requirements.txt
  benchmark-opentelemetry-sdk: -r {toxinidir}/opentelemetry-sdk/benchmark-requirements.txt

  semantic-conventions: -r {toxinidir}/opentelemetry-semantic-conventions/test-requirements.txt
```

Para cada pacote, instalar as dependências necessárias pro teste

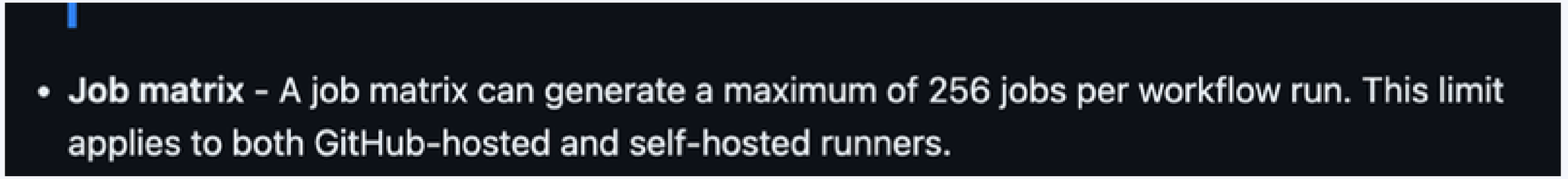
```
commands =
  test-opentelemetry-api: pytest {toxinidir}/opentelemetry-api/tests {posargs}
  lint-opentelemetry-api: pylint {toxinidir}/opentelemetry-api

  test-opentelemetry-sdk: pytest {toxinidir}/opentelemetry-sdk/tests {posargs}
  lint-opentelemetry-sdk: pylint {toxinidir}/opentelemetry-sdk
  benchmark-opentelemetry-sdk: pytest {toxinidir}/opentelemetry-sdk/benchmarks --ben

  test-opentelemetry-proto-protobuf5: pytest {toxinidir}/opentelemetry-proto/tests {
  lint-opentelemetry-proto-protobuf5: pylint {toxinidir}/opentelemetry-proto

  test-opentelemetry-semantic-conventions: pytest {toxinidir}/opentelemetry-semantic
  lint-opentelemetry-semantic-conventions: pylint {toxinidir}/opentelemetry-semantic
```

Para cada pacote, rodar os testes, lint ou benchmark

Só criar uma matriz dos 80 pacotes

...

Até funcionou por um tempo, mas...



Jinja2 Templates
+ Automação em Python
=
Gerar os workflows

```
33    jobs:
34      {%- for job_data in job_datas %}
35
36      {{ job_data.name }}:
37        name: {{ job_data.ui_name }}
38        runs-on: {{ job_data.os }}
39        timeout-minutes: 30
40        steps:
41          - name: Checkout repo @ SHA - ${% raw %}{{ github.sha }}{% endraw %}
42            uses: actions/checkout@v4
43
44          - name: Set up Python {{ job_data.python_version }}
45            uses: actions/setup-python@v5
46            with:
47              python-version: "{{ job_data.python_version }}"
48
49          - name: Install tox
50            run: pip install tox-uv
51          {%- if job_data.os == "windows-latest" %}
52
53          - name: Configure git to support long filenames
54            run: git config --system core.longpaths true
55          {%- endif %}
56
57          - name: Run tests
58            run: tox -e {{ job_data.tox_env }} -- -ra
59      {%- endfor %}
```

```yaml
  py39-test-instrumentation-openai-v2-oldest_ubuntu-latest:
    name: instrumentation-openai-v2-oldest 3.9 Ubuntu
    runs-on: ubuntu-latest
    timeout-minutes: 30
    steps:
      - name: Checkout repo @ SHA - ${{ github.sha }}
        uses: actions/checkout@v4

      - name: Set up Python 3.9
        uses: actions/setup-python@v5
        with:
          python-version: "3.9"

      - name: Install tox
        run: pip install tox-uv

      - name: Run tests
        run: tox -e py39-test-instrumentation-openai-v2-oldest -- -ra
```

```yaml
  generate-workflows:
    name: generate-workflows
    runs-on: ubuntu-latest
    timeout-minutes: 30
    if: |
      !contains(github.event.pull_request.labels.*.name, 'Skip generate-workflows')
      && github.event.pull_request.user.login != 'otelbot[bot]' && github.event_name == 'pull_request'
    steps:
      - name: Checkout repo @ SHA - ${{ github.sha }}
        uses: actions/checkout@v4

      - name: Set up Python 3.11
        uses: actions/setup-python@v5
        with:
          python-version: "3.11"

      - name: Install tox
        run: pip install tox-uv

      - name: Run tests
        run: tox -e generate-workflows

      - name: Check github workflows are up to date
        run: git diff --exit-code || (echo 'Generated workflows are out of date, run "tox -e generate-workflows" and commit the changes in this PR.' && exit 1)
```

Também verifica se está tudo Up-to-date

Releases + Backport

Opentelemetry Python → PEP 440

- Release: X.Y.Z → e.g., **1.37.0**
- Pre-releases: aN (alpha), bN (beta), rcN (release candidate) e.g., **0.58b0**
- Dev releases: .dev → e.g., **1.37.0.dev (fica apenas no git)**

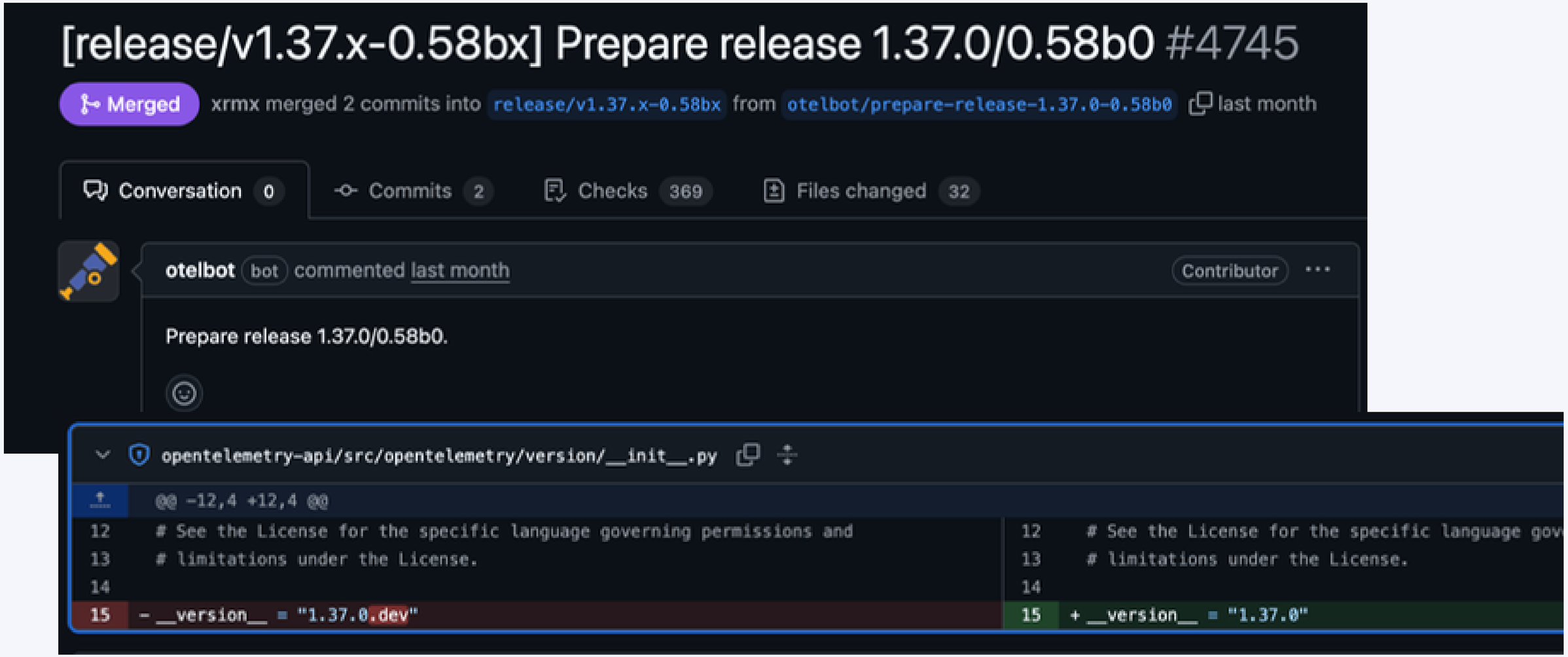
e.g., opentelemetry-api → 1.34.0, 1.34.1 (patch release),

opentelemetry-instrumentation → 0.58b0, 0.58b1 (patch)

Automação via Github Actions cria uma branch de release por repositório: **e.g., release/v1.37.x-0.58bx**

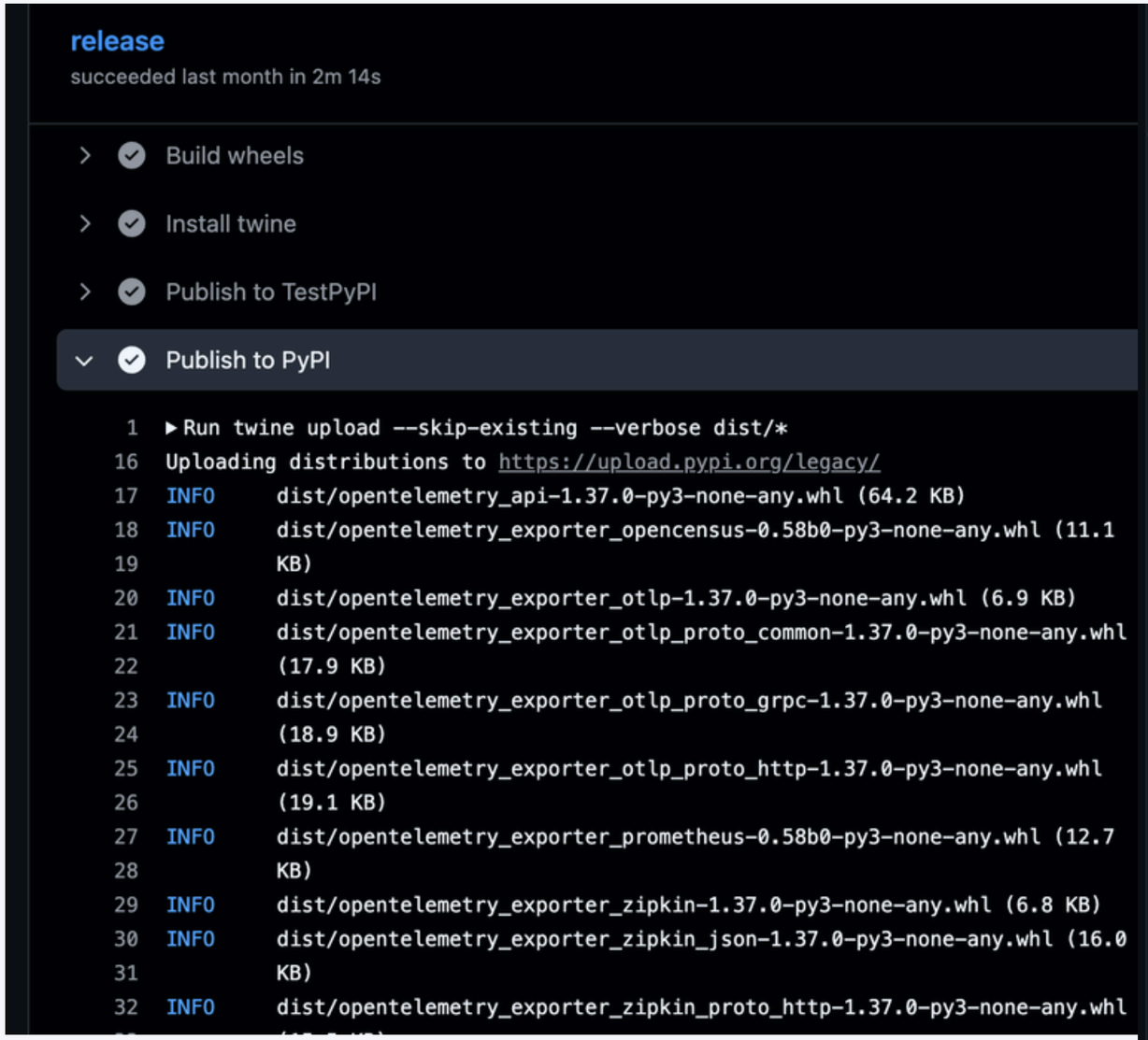
2 PRs por repositório (automação via Github Actions)

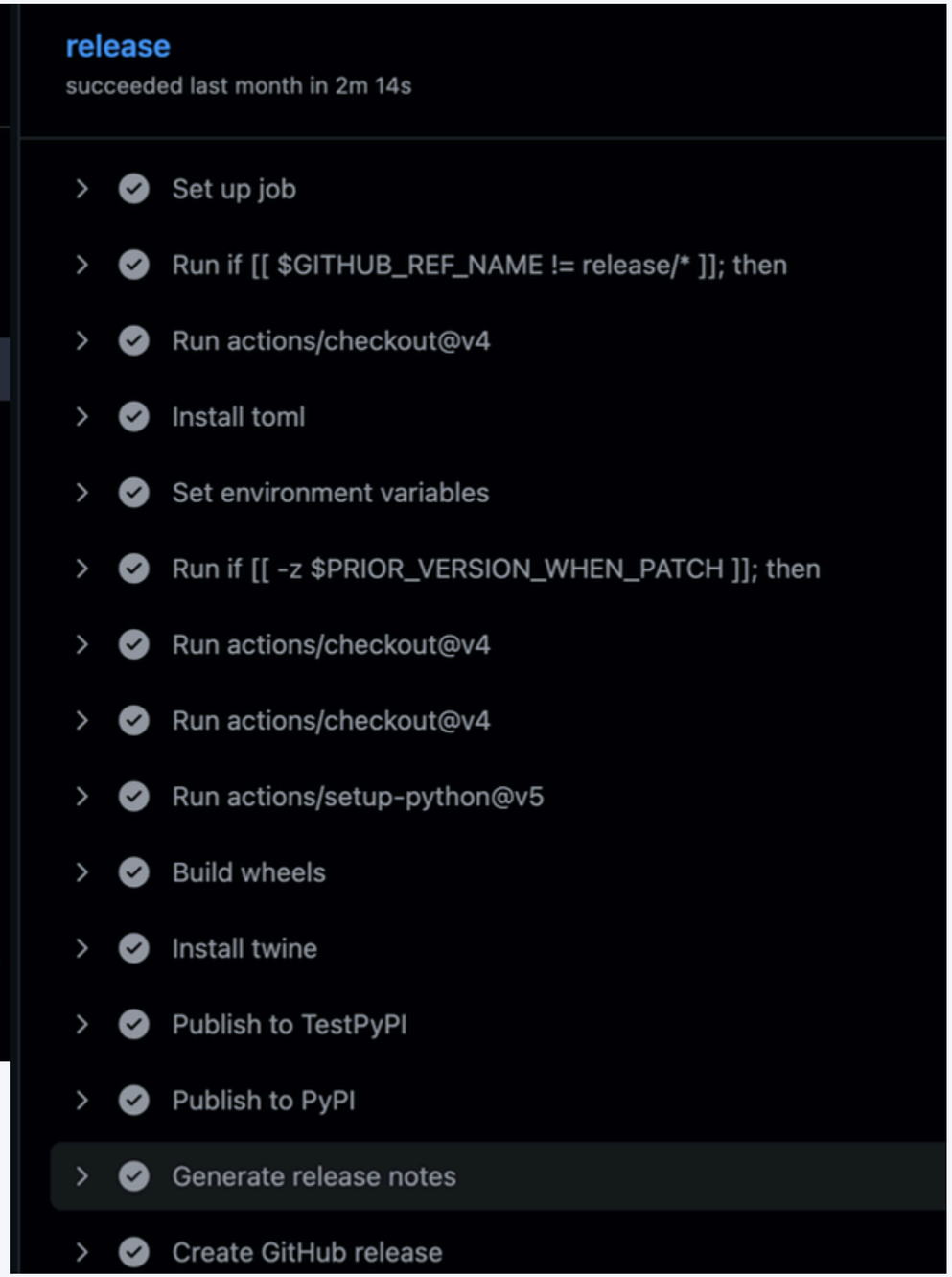
- 1 para fazer bump da versão nos arquivos removendo o sufixo .dev para ser mergeado na branch de release (o que vai ser publicado no PyPi)
- 1 para fazer bump da versão mantendo o sufixo .dev para a branch main (manter o ciclo de desenvolvimento ativo)



Release principal: Automação via GHA para realizar o build dos pacotes python e publicar no pypi a partir do que tiver na branch de release

PR para atualizar a main com a versão de dev (próxima versão)



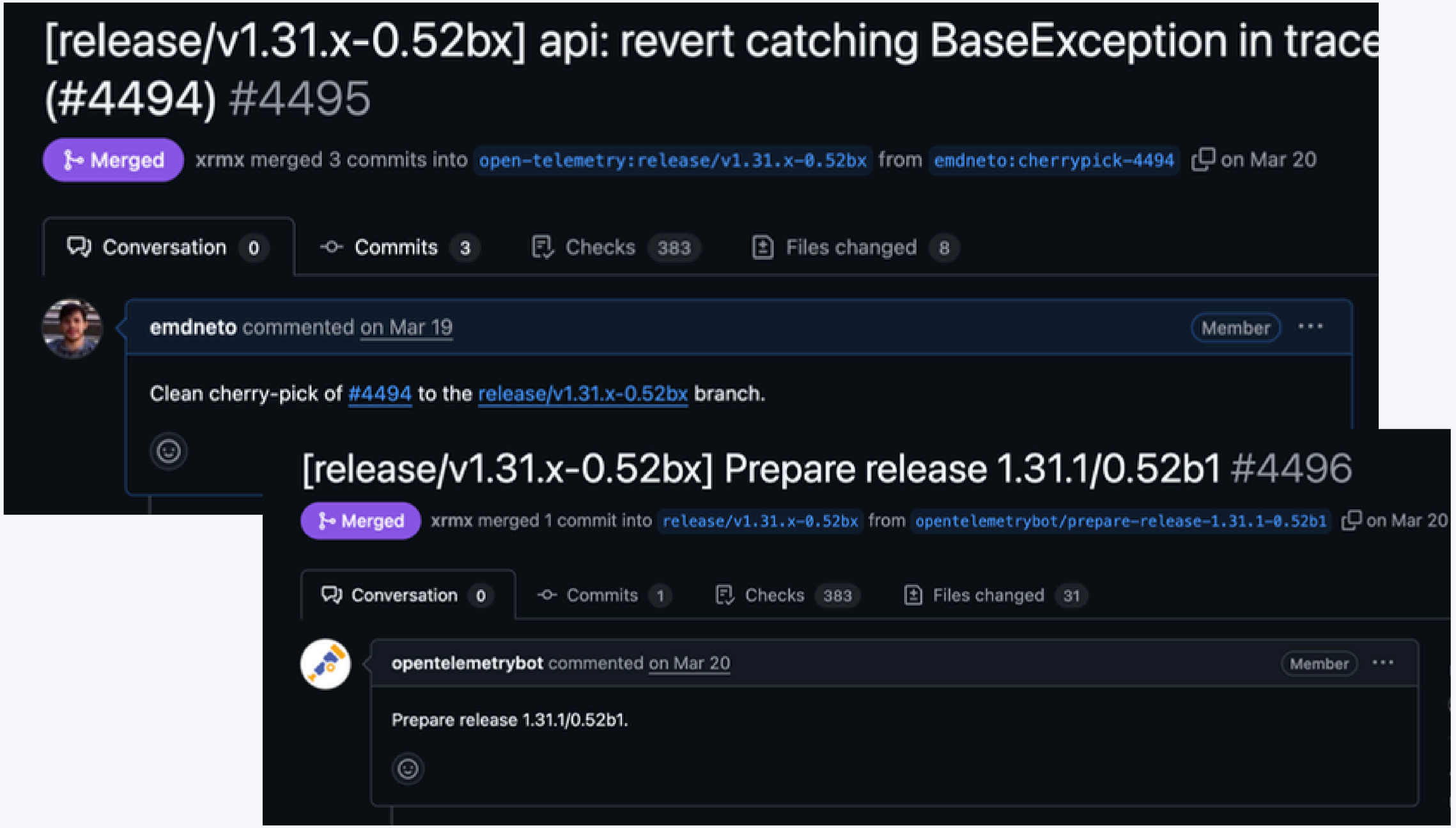


Release principal

Fluxo de aplicar patches:

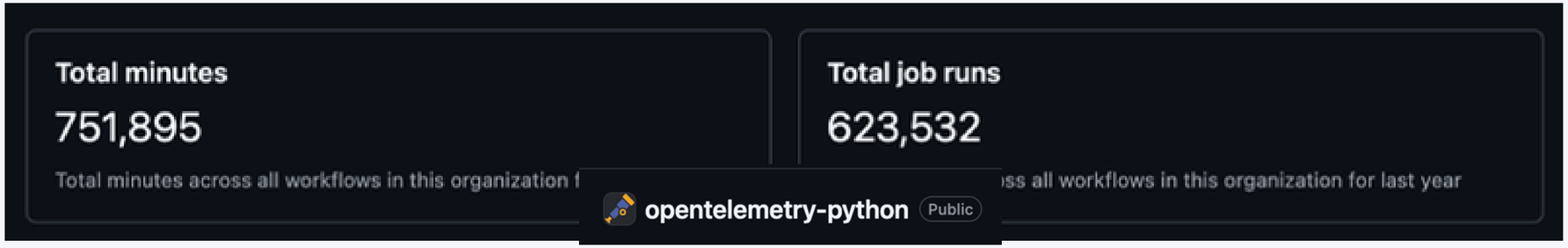
1. Abrir PR com o fix apontando pra main
2. Approve + Merge do PR com o fix, adicionar changelog etc
3. Fazer git cherry-pick do commit na branch de release e abrir um novo PR para branch de release (backport)
4. Approve + Merge do PR na branch de release

Para patch release: Triggar workflow de prepare Patch release via Github Actions + Mergear o PR gerado + Triggar workflow de release (mesmo usado para release principal)


[release/v1.31.x-0.52bx] api: revert catching BaseException in trace (#4494) #4495
Merged
xrmx merged 3 commits into open-telemetry:release/v1.31.x-0.52bx from emdneto:cherrypick-4494 on Mar 20
Conversation 0
Commits 3
Checks 383
Files changed 8
emdneto commented on Mar 19
Member
Clean cherry-pick of #4494 to the release/v1.31.x-0.52bx branch.
[release/v1.31.x-0.52bx] Prepare release 1.31.1/0.52b1 #4496
Merged
xrmx merged 1 commit into release/v1.31.x-0.52bx from opentelemetrybot/prepare-release-1.31.1-0.52b1 on Mar 20
Conversation 0
Commits 1
Checks 383
Files changed 31
opentelemetrybot commented on Mar 20
Member
Prepare release 1.31.1/0.52b1.

```yaml
      - name: Update version
        run: .github/scripts/update-version-patch.sh $STABLE_VERSION $UNSTABLE_VERSION $STABLE_VERSION_PREV $UNSTABLE_VERSION_PREV

      - name: Set up Python 3.9
        uses: actions/setup-python@v5
        with:
          python-version: 3.9
      - name: Install tox
        run: pip install tox
      - name: run tox
        run: tox -e generate

      - name: Update the change log with the approximate release date
        run: |
          date=$(date "+%Y-%m-%d")
          sed -Ei "s/^## Unreleased$/## Version ${STABLE_VERSION}\/${UNSTABLE_VERSION} ($date)/" CHANGELOG.md

      - name: Use CLA approved github bot
        run: .github/scripts/use-cla-approved-github-bot.sh

      - uses: actions/create-github-app-token@df432ceedc7162793a195dd1713ff69aefc7379e # v2.0.6
        id: otelbot-token
        with:
          app-id: ${{ vars.OTELBOT_APP_ID }}
          private-key: ${{ secrets.OTELBOT_PRIVATE_KEY }}

      - name: Create pull request
        id: create_pr
        env:
          # not using secrets.GITHUB_TOKEN since pull requests from that token do not run workflows
          GITHUB_TOKEN: ${{ steps.otelbot-token.outputs.token }}
        run: |
          message="Prepare release ${STABLE_VERSION}/${UNSTABLE_VERSION}"
          branch="otelbot/prepare-release-${STABLE_VERSION}-${UNSTABLE_VERSION}"

          git commit -a -m "$message"
          git push origin HEAD:$branch
          pr_url=$(gh pr create --title "[$GITHUB_REF_NAME] $message" \
                      --body "$message." \
                      --head $branch \
                      --base $GITHUB_REF_NAME)
```

Desafios e Lições Aprendidas


Total minutes
751,895
Total minutes across all workflows in this organization f
Total job runs
623,532
ss all workflows in this organization for last year
opentelemetry-python
Public


Total minutes
2,402,779
Total minutes across all workflows in this organization for last year
Total job runs
2,325,445
Total job runs across all workflows in this organization for last year
opentelemetry-python-contrib
Public

52.577h


Avg job run time
30s
Average run time of jobs in this organization for last month
Avg job queue time
2s
Average queue time of jobs in this organization for last month
Job failure rate
3%
Failure rate across jobs in this organization for last month
Failed job usage
1,541
Total minutes used across failed jobs in this organization for last month
opentelemetry-python
Public


Avg job run time
21s
Average run time of jobs in this organization for last month
Avg job queue time
5s
Average queue time of jobs in this organization for last month
Job failure rate
1%
Failure rate across jobs in this organization for last month
Failed job usage
3,815
Total minutes used across failed jobs in this organization for last month
opentelemetry-python-contrib
Public

1. Arquivos tox.ini imensos -- Chegando a ~1040 linhas
2. Difícil manter sincronizado o que está no tox.ini VS o que roda na CI
3. Instalação de pacotes lenta em alguns testes
4. CI de Lint demorava séculos para rodar
5. Testar um mesmo pacote em múltiplas versões do Python só que também para múltiplas versões de uma dependência específica

1. Matriz no Github Action tem limite de 250 Jobs -- sem chances

2. Testar o “básico” do repo -contrib em todo PR no repositório core

3. Testes no -contrib apontam para branch main no -core

```
[testenv]
test_deps =
  opentelemetry-api@{env:CORE_REPO}\#egg=opentelemetry-api&subdirectory=opentelemetry-api
  opentelemetry-semantic-conventions@{env:CORE_REPO}\#egg=opentelemetry-semantic-convention
  opentelemetry-sdk@{env:CORE_REPO}\#egg=opentelemetry-sdk&subdirectory=opentelemetry-sdk
  opentelemetry-test-utils@{env:CORE_REPO}\#egg=opentelemetry-test-utils&subdirectory=tests
```

Obrigado!

@emdneto