Changelogs e TownCrier

Live de Python # 278

Roteiro

1. Releases?

O ciclo sem fim...

2. ChangeLogs

Do que se alimentam

3. TownCrier

A nossa ferramenta!

4. Documentação

Juntado tudo

apoia.se/livedepython

pix.dunossauro@gmail.com

patreon.com/dunossauro

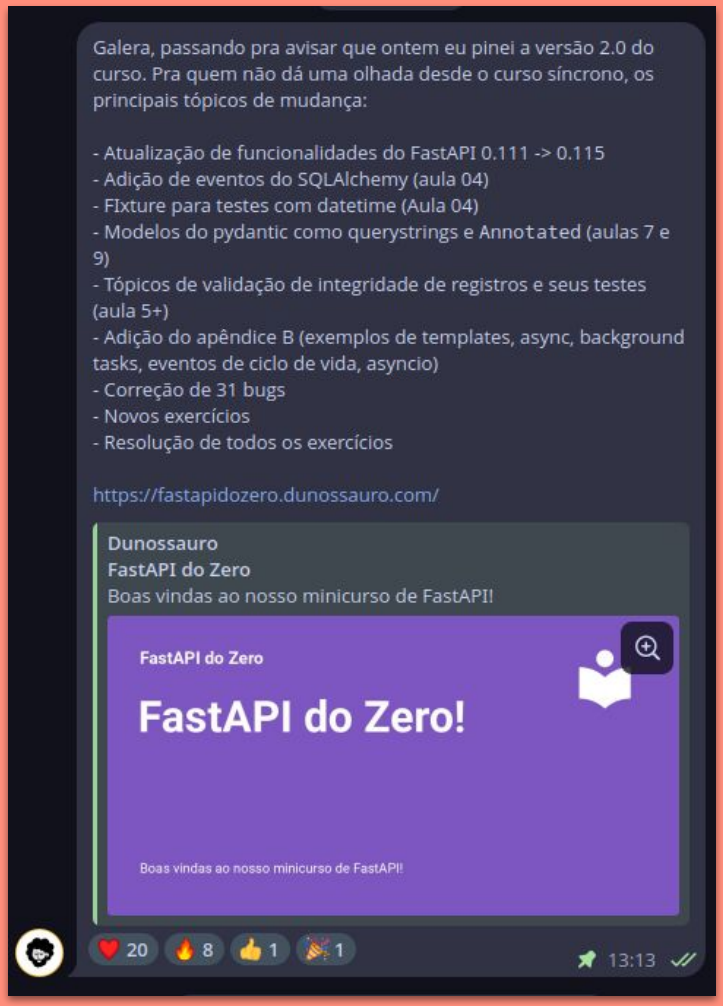
Ajude o projeto <3

Adriana Cavalcanti, Alexandre Girardello, Alexandre Harano, Alexandre Lima, Alexandre Takahashi, Alexandre Villares, Alfredo Braga, Allan Kleitson, Alysson Oliveira, Andre Azevedo, Andre Makoski, Andre Paula, Apc 16, Arthur Santiago, Aslay Clevisson, Aurelio Costa, Belisa Arnhold, Bernarducs, Biancarosa, Brisa Nascimento, Bruno, Bruno Barcellos, Bruno Batista, Bruno Freitas, Bruno Ramos, Bruno Russian, Caio Nascimento, Carlos Gonçalves, Carlos Ramos, Celio Araujo, Christian Fischer, Cleiton Fonseca, Controlado, Curtos Treino, Daniel Aguiar, Daniel Bianchi, Daniel Brito, Daniel Souza, Daniel Wojcickoski, Danilo Boas, Danilo Silva, David Couto, David Kwast, Denis Bernardo, Dgeison, Diego Guimarães, Dino, Edgar, Eduardo Pizorno, Emerson Rafael, Érico Andrei, Érico Cavalcanti, Everton Silva, Fabio Barros, Fabio Faria, Fabiokleis, Felipe Adeildo, Felipe Augusto, Felipe Corrêa, Fernanda Prado, Fernandocelmer, Fichele Marias, Francisco Aclima, Francisco Faria, Frederico Damian, Fulvio Murenu, Gabriel Lira, Gabriel Mizuno, Gabriel Paiva, Gabriel Ramos, Gabriel Simonetto, Gabriel Sobrenome, Geilton Cruz, Geisler Dias, Giovanna Teodoro, Giuliano Silva, Guibeira, Guilherme Felitti, Guilherme Ostrock, Guilherme Piccioni, Gustavo Suto, Haelmo Almeida, Harold Gautschi, Heitor Fernandes, Hellyson Ferreira, Helton, Helvio Rezende, Henri Alves, Henrique Andrade, Henrique Machado, Hiago Couto, Igor Taconi, Ivan Santiago, Janael Pinheiro, Jean Melo, Jean Victor, Jeferson Vitalino, Jefferson Antunes, Jefferson Silva, Jerry Ubiratan, Jlx, Joao Rocha, John Peace, Jonas Araujo, Jonatas Leon, Jônatas Silva, Jorge Silva, Jose Barroso, Joseíto Júnior, Jose Mazolini, José Predo, Josir Gomes, Jrborba, Juan Felipe, Juliana Machado, Julio Batista-silva, Julio Franco, Kaio Peixoto, Leandro O., Leandro Pina, Leandro Vieira, Leonan Ferreira, Leonardo Adelmo, Leonardo Mello, Leonardo Nazareth, Lisandro Pires, Lucas Carderelli, Lucas Castro, Lucas Mello, Lucas Mendes, Lucas Nascimento, Lucas Schneider, Lucas Simon, Luciano Ratamero, Luciano Teixeira, Luiz Carlos, Luiz Duarte, Luiz Martins, Luiz Paula, Luiz Perciliano, Mackilem Laan, Marcelo Araujo, Marcelo Fonseca, Marcelo Grimberg, Marcio Freitas, Marcio Junior, Marcos Almeida, Marcos Oliveira, Marina Passos, Marlon Rocha, Mateusamorim96, Mateus Lisboa, Matheus Vian, Mírian Batista, Mlevi Lsantos, Murilo Carvalho, Ocimar Zolin, Otávio Carneiro, Patrick Felipe, Pedro Henrique, Peterson Santos, Phmmdev, Prof Santana, Pytonyc, Rafael Faccio, Rafael Romão, Raimundo Ramos, Ramayana Menezes, Renan, Renan Sebastião, Rene Pessoto, Renne Rocha, Ricardo Silva, Ricardo Viana, Rinaldo Magalhaes, Rodrigo Barretos, Rodrigo Oliveira, Rogério Nogueira, Rui Jr, Samanta Cicilia, Santhiago Cristiano, Sergio Nascimento, Sherlock Holmes, Shirakawa, Tenorio, Téo Calvo, Tharles Andrade, Thiago Araujo, Thiago Lucca, Thiago Paiva, Tiago, Tiago Emanuel, Tiago Henrique, Tomás Tamantini, Valdir, Varlei Menconi, Vinicius Silva, Vinicius Souza, Vinicius Stein, Vladimir Lemos, Williamslews, Willian Lopes, Zeca Figueiredo, Zero! Studio

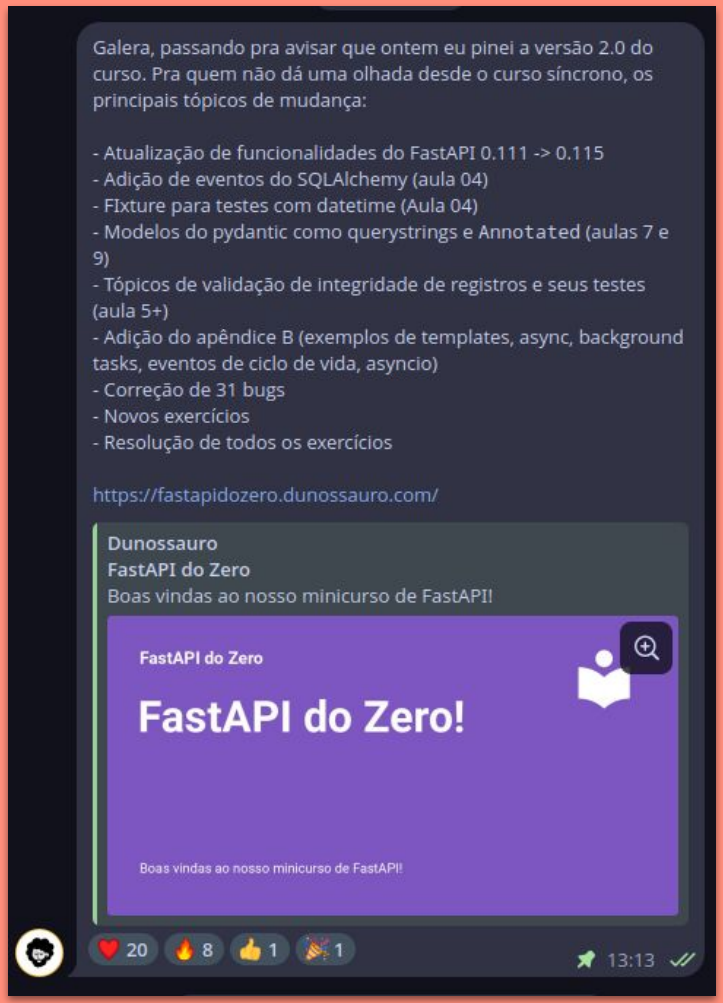
Obrigado você

Uma pequena história / A razão dessa live

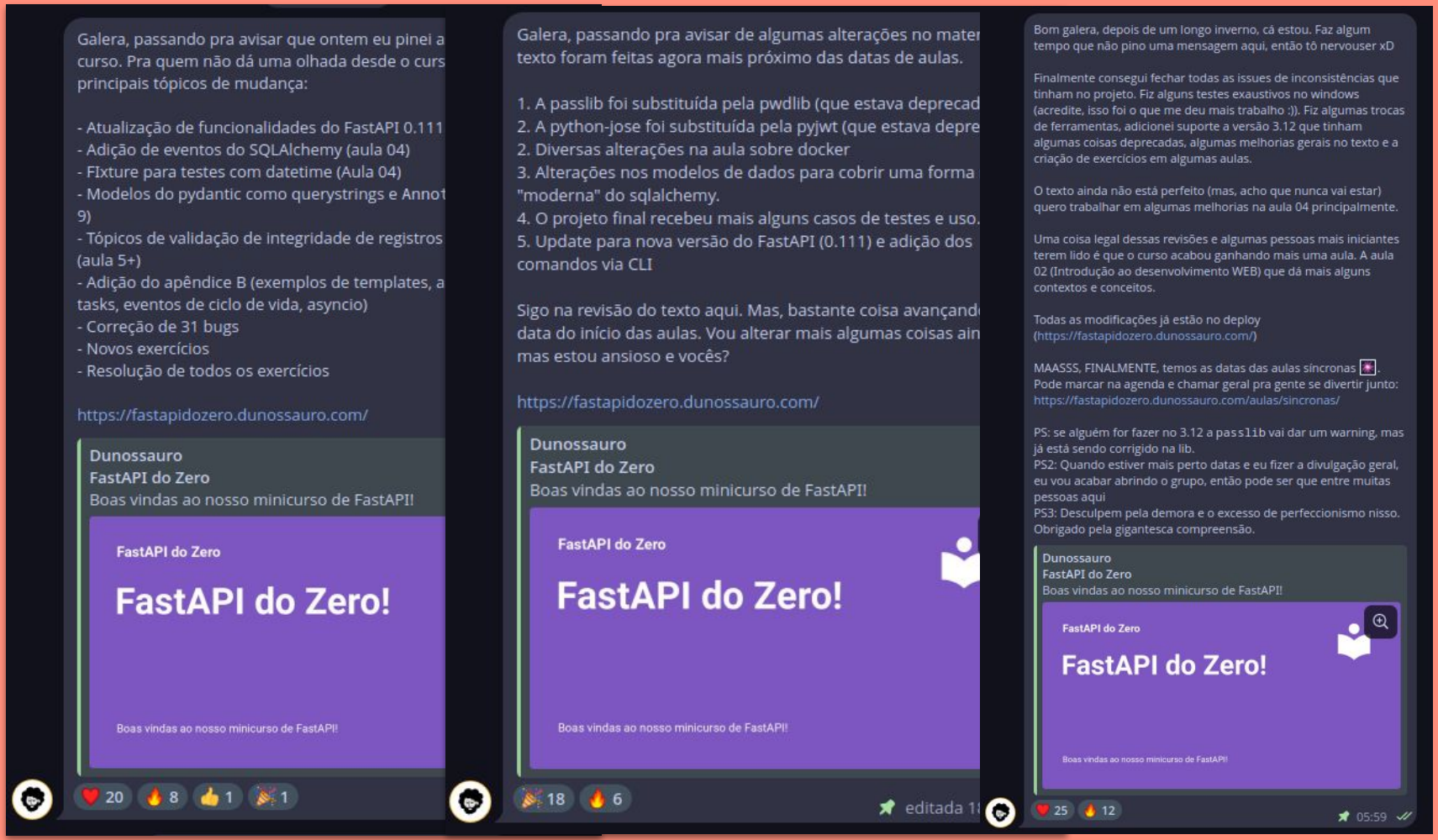
O estado das coisas



O estado das coisas



O estado das coisas



"As pessoas confiaram no meu trabalho, menos sem ter a menor ideia do que eu estava fazendo"

Empatia

"O doido sumiu, cadê o material?
Não fala mais nada..."

Ou a falta dela

O ciclo sem fim...

Releases?

Releases?

Softwares são vivos, eles estão em constante modificação. Em uma janela maior ou menor de tempos eles:

- Adicionam ou removem **código**
- Adicionam ou removem **documentação**
- Adicionam ou removem **testes**
- Alteram a **infraestrutura**
- Alteram o **ambiente**
- ...

Releases?

Com as alterações vem:

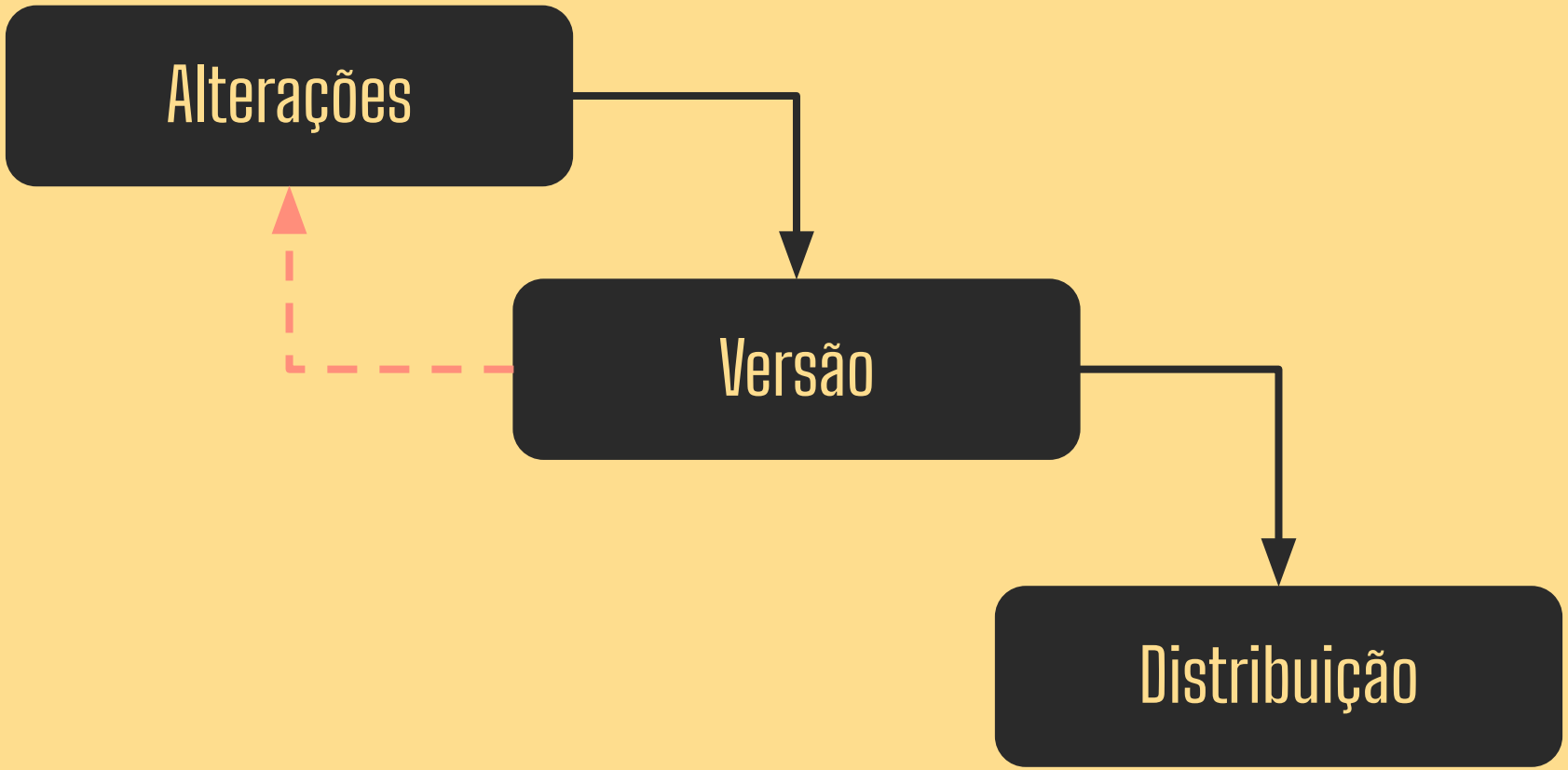
- Novas **versões**
  - 1.0.0 -> 1.0.1 (SemVer)
  - 24.12.3 -> 25.01(CalVer)
- Novos **pacotes**
  - wheels
  - executáveis
    - exe, appimage, dmg, flatpak, ...
- Novos **artefatos**
  - Arquivos estáticos
  - Documentação
  - ...

Releases?

Como consequência, uma nova **distribuição** precisa ser feita:

- Uploads para plataformas
- Disponibilizar para download
- Times internos
  - Marketing
  - Vendas
  - ...

Releases



O lado interno do fluxo

E os clientes com isso?

De fora do projeto, vivemos um **ciclo de updates**.

O vivido pelo cliente vai variar de acordo com o tipo de distribuição:

- Biblioteca
- Software / Programa

E os clientes com isso?

De fora do projeto, vivemos um **ciclo de updates**.

O vivido pelo cliente vai variar de acordo com o tipo de distribuição:

- **Biblioteca**
  - Algo precisa ser alterado?
    - Algo que usamos **foi** removido?
    - Algo que usamos **será** removido?
    - A API de código mudou?
  - Quais as novidades?
    - Funcionalidades novas?
  - Bugs foram corrigidos?
- Software / Programa

E os clientes com isso?

De fora do projeto, vivemos um **ciclo de updates**.

O vivido pelo cliente vai variar de acordo com o tipo de distribuição:

- Biblioteca
- **Software / Programa**
  - O que existe de novo?
  - É compatível com as versões passadas?
  - Algo foi removido?
  - Bugs foram corrigidos?

Release vs Cliente

Deve existir algo que notifique os clientes sobre as alterações. As informações devem responder às perguntas sobre alterações, bugs, novidades e etc.

Para isso, **em teoria**, existem três tipos possíveis de documentos:

- Lançamento (Lauch)
- Notas da versão (Release Notes)
- Logs de alterações (changelogs)

Cada um desses documentos tem objetivos únicos e atende diferentes tipos de cliente.

Exemplos dizem mais

Antes de explicar o que cada documento se destina a ser, vamos ver com exemplos reais.

Os documentos do Python:

- Lançamento
- Notas da release
- Logs de alteração

Exemplo de um software comercial OnlyOffice:

- Lançamento
- Release notes
- Changelogs

Lançamento

Um documento mais voltado para marketing, vendas, público geral.

- **Baixa granularidade**
- Novidades
  - O que o software tem de funcionalidades?
  - O que entrou?
- Benefícios
  - O que faz valer o update?
- Coisas críticas.
  - Erros / Bugs / Segurança
- Forma bastante **resumida**
- **Linguagem amigável**

Release notes

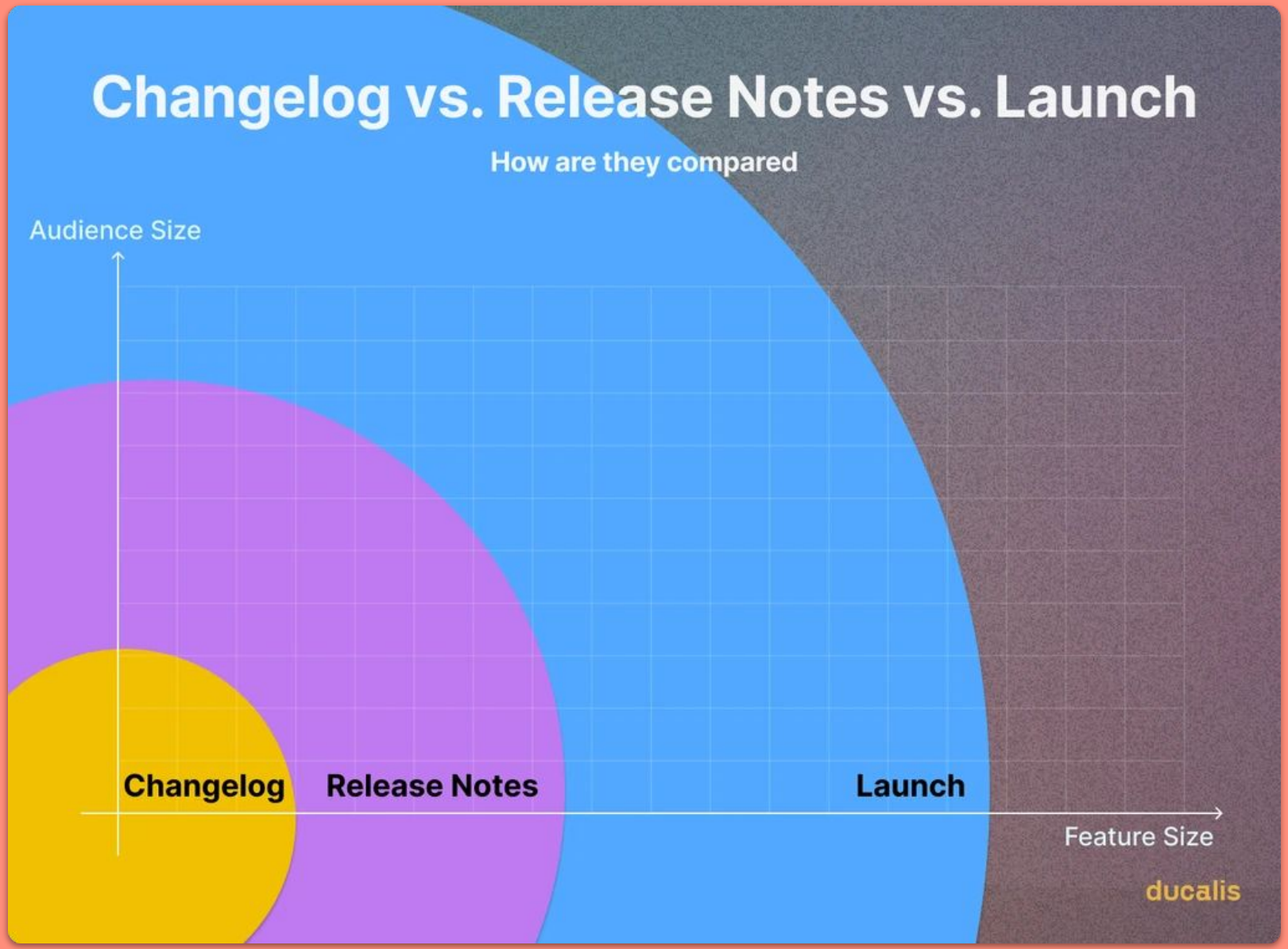
Usuários ativos do sistema, pessoas que precisam entender a nova versão.

- **Média granularidade**
- Mudanças
  - Como mudou?
  - Por que mudou?
  - Como usar as coisas novas?
    - Exemplos de uso.
- Algo foi removido?
  - O que foi removido?
  - Tem substituto?
- Linguagem **razoável.**

Changelogs

Desenvolvedores, usuários avançados, curiosos...

- **Alta granularidade**
- O que foi feito?
  - Quando foi feito?
  - Quem fez?
  - Tem issue?
- Alterou onde?
- Que tipo de alteração?
- Linguagem **técnica**



https://hi.ducalis.io/changelog/release-notes-best-practices

O motivo de estarmos aqui!

Change logs

Uma visão menos superficial

Changelogs são documentos que expõem **alterações visíveis aos usuários** e que valem a pena serem mencionadas a quem acompanha o software.

São escritos em ordem cronológica reversa (mais recente → menos recentes)

Existem alguns padrões para changeslogs (diferentes e às vezes incompatíveis)

- GNU
- keepachangelog
- common changelog

GNU [~ decada de 1990]

O guia de boas práticas do GNU (6.8) define que devemos ter um arquivo chamado **ChangeLog** e seu formato deve ter um formato bastante rígido para cada entrada:

```
2025-01-27  João Silva  <joao.silva@email.com>

    Corrige erro de cálculo em função de soma

    * src/calculadora.c (soma): Corrige erro que causava o cálculo incorreto
      quando os valores passavam de 1000.
    * test/calculadora_testes.c (testa_soma): Atualiza teste para cobrir
      cenário com valores altos.
```

Exemplo: GNU emacs

Discutindo a entrada

```
2025-01-27  João Silva  <joao.silva@email.com>

    Corrige erro de cálculo em função de soma

    * src/calculadora.c (soma): Corrige erro que causava o cálculo incorreto
      quando os valores passavam de 1000.
    * test/calculadora_testes.c (testa_soma): Atualiza teste para cobrir
      cenário com valores altos.

2025-01-20  Maria Oliveira  <maria.oliveira@email.com>

    Adiciona verificação de divisão por zero

    * src/calculadora.c (divisao) [#ifdef VERIFICA_DIV_ZERO]: Adiciona verificação
      para evitar divisão por zero, aplicável apenas quando a macro
      VERIFICA_DIV_ZERO está definida.
```

Keep a changelog

Um padrão mais "moderno" é o Keep A Changelog, iniciado por Olivier Lacan em 2014.

Algumas premissas são bastante interessantes:

- Changelogs **são para humanos,** não máquinas
- Versões e seções devem ser vinculáveis (links)
- A data de lançamento de versões deve ser exibida
- Introduz o conceito de **tipos** de alteração
- Introduz uma seção **"Não publicado"**
  - Alterações feitas, mas não lançadas
- Convenciona o arquivo "CHANGELOG.md"
- Coloca VCS como uma prática ruim
- *Existem críticas se esse formato é changelogs ou release notes!*

Tipos de mudança

Uma das características principais nas entradas, são os **tipos de mudança**.

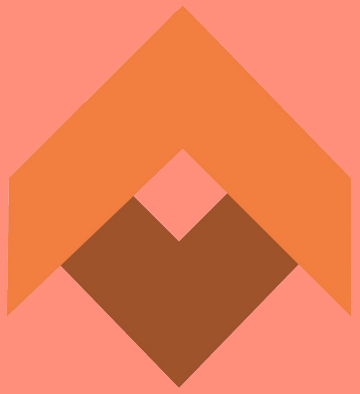
Que tendem a **categorizar entradas** e as agrupar no changelog:

- **Adicionado**: Novas funcionalidades
- **Modificado**: Alterações em funções existentes
- **Obsoleto**: Funcionalidades que serão removidas
- **Removido**: Funcionalidades removidas nesta versão
- **Corrigido**: Qualquer correção de bug
- **Segurança**: Vulnerabilidades de segurança

Um visão do **CHANGELOG.md**

```
# Changelog

## [Unreleased]
### Adicionado
- Implementação de uma função para processamento de dados em formato JSON. [#45](link)
- Suporte a variáveis de ambiente usando `python-dotenv`. [#42](linkissue)

### Modificado
- Refatoração do módulo `auth.py` para utilizar o pacote `cryptography` em vez de `pycryptodome`. [#38](link)

## [1.0.0] - 2025-01-20
### Adicionado
- Primeira versão do projeto com funcionalidades básicas de autenticação via OAuth [#30](link)
- Endpoint `/login` para autenticação de usuários via POST. [#32](linkissue)

### Modificado
- Ajustes no tratamento de erros para arquivos CSV corrompidos. [#28](linkissue)

### Removido
- Remoção do suporte à versão do Python 2.7. [#25](linkissue)
```

Common Changelog



Criado em 2021 por Vincent Weevers como um conjunto de "regras" mais restritas que o KAC, por exemplo:

- Adiciona um cabeçalho de avisos
- Entradas sempre no imperativo
- Adiciona autoria a alteração
- Links para quando não existe issue
- Prefixos para alterações significativas

Exemplo em ação

```
# Changelog
## 2.0.0 - 2025-01-27
_Se você está atualizando: veja o arquivo [`UPGRADING.md`](UPGRADING.md)._

### Alterado
- **Breaking:** Refatoração do fluxo de autenticação para usar OAuth2. ([`abc1234`](link)) (Fuasto)
- Refatoração da API interna para melhorar a busca com grandes volumes de dados. ([`d3f4b32`](link)) (F.)

### Adicionado
- Adicionada opção de configuração para controlar a verbosidade dos logs. ([`ef56789`](link) (Tião do gas)

### Removido
- **Breaking:** Removido suporte para métodos de autenticação tokens customizados. ([`56d1a32`](link)) (Ju)

### Corrigido
- Corrigido erro em que tokens de sessão não eram invalidados após o logout. ([#45](link)) (taconi)

## 1.1.0 - 2025-01-15
_Primeira versão com suporte ao OAuth2._

### Adicionado
- Adicionada autenticação via OAuth2. ([`a1b2c3d`](link)) (Sergião berranteiro)
```

[polêmica] Git serve como changelog?

Hacker News

new | past | comments | ask | show | jobs | submit

▲ Git log is not a changelog (agateau.com)

124 points by agateau on July 16, 2022 | hide | past | favorite | 88 comments

https://news.ycombinator.com/item?id=32122028

Git serve como changelog?

Diffs de registros de commits

Utilizar diffs de registros de commits como changelogs é uma má ideia: eles estão cheios de bagunça. Coisas como commits de mesclagem, commits com títulos estranhos, alterações de documentação etc.

O propósito de um commit é documentar a etapa na evolução do código fonte. Alguns projetos limpam os commits, outros não.

O propósito de uma entrada de changelog é documentar as diferenças notáveis, muitas vezes de múltiplos commits, para comunicá-las de forma clara aos usuários.

https://keepachangelog.com/pt-BR/1.1.0/#bad-practices

Git serve como changelog?

Hacker News
new | past | comments | ask | show | jobs | submit

▲ Git log is not a changelog (agateau.com)

Diffs de registros de commits

Utilizar diffs de registros de commits como changelogs é uma má ideia: eles estão cheios de bagunça. Coisas como

4.2. Commits Convencionais

Os Commits Convencionais adicionam sobrecarga cognitiva ao se injetarem no meio de um fluxo de trabalho. As alterações não começam nem terminam com commits.

Para gerar changelogs, os Commits Convencionais ajudam a categorizar inicialmente as alterações, mas o uso dos Commits Convencionais também significa que as mensagens de commit devem ser convertidas para um formato mais legível. Usar esse formulário em primeiro lugar alinhará o conteúdo de maneira geral. Começando com histórias e épicos, depois tickets, depois commits, depois pull requests e depois changelogs. Em última análise, isto significa menos sobrecarga cognitiva, abrindo espaço para que todos compreendam e reconheçam cada mudança apresentada em múltiplos contextos. Esta é a abordagem “ponta a ponta” do Common Changelog.

https://common-changelog.org/#4-antipatterns

Ferramentas

Existem diversas ferramentas para auxiliar na criação de changelogs, algumas baseadas em git:

- Git-cliff
- Commitzen

E ferramentas com interações humanas:

- Openstack Reno
- **Towncrier**

Um breve passeio pelo nosso ecossistema

- FastAPI (misto: Notes/Logs):
  - https://fastapi.tiangolo.com/release-notes/
- Mypy (misto: Notes/Logs):
  - https://mypy-lang.blogspot.com/
- Material Mkdocs (changelogs bruto):
  - https://squidfunk.github.io/mkdocs-material/insiders/changelog
- Pytest (changelogs com um Q de notes):
  - https://docs.pytest.org/en/stable/changelog.html

Town crier

A ferramenta

Towncrier

Projeto de CLI para gerenciar changelogs de projetos python. Criado pela Amber Brown. Mantenedora do Twisted e também do Incremental.

- Licença: **MIT**
- Primeira release: **15.0.0** (CalVer)
- Release atual: **24.8.0**

```
pip install towncrier
```

Termo

town crier

noun [C]

UK /ˌtaʊn ˈkraɪ.əʳ/ US /ˌtaʊn ˈkraɪ.ɚ/

Add to word list

in the past, a person whose job was to make official announcements in a town or village by calling them out in public

https://dictionary.cambridge.org/dictionary/english/town-crier

https://en.wikipedia.org/wiki/Peter_Moore_%28town_crier%29

Termo

Arauto

noun [C]

UK /ˌtaʊn ˈkraɪ.əʳ/ US /ˌtaʊn ˈkraɪ.ɚ/

Add to word list

No passado, trabalho de ler os comunicados oficiais [do rei] as cidades e vilarejos de forma verbal.

https://dictionary.cambridge.org/dictionary/english/town-crier

a.org/wiki/Peter_Moore_%28town_crier%29

Hear ye, hear ye, says the

`towncrier`

Docs Read The Docs | license MIT | pypi v24.8.0

Arauto dos dias atuais...

https://youtu.be/eQ8AdwWwLIs

Funcionamento

O towncrier funciona por meio de **fragmentos**. Pequenos arquivos de texto que são criados pelo CLI em um diretório escolhido por nós.

```
# pyproject.toml
[tool.towncrier]
directory = "changes"
filename = "NEWS.rst" # "CHANGELOG.md"
```

Uso

Com a configuração mínima feita, podemos criar os fragmentos usando a CLI:

```
towncrier create
    --content 'Mensagem de log'
    +.misc.md
```

Uso

content é a mensagem que será gravada no fragmento

```
towncrier create
    --content 'Mensagem de log'
    +.misc.md
```

Uso

- **+**: Significa que não está relacionada a nenhuma issue
- **misc**: É o **tipo** do fragmento
- .md: o fragmento vai ser do tipo markdown (pode ser rst também)

```
towncrier create
    --content 'Mensagem de log'
    +.misc.md
```

Modo interativo

Vamos ver o que rola!

```
towncrier create
```

Tipos

Por padrão, towncrier cria os tipos:

- **feature**: Uma nova funcionalidade
- **bugfix**: Uma correção de bug
- **doc**: Melhorias na documentação
- **removal**: Remoção ou "deprecated" de alguma API
- **misc**: Uma ação feita que não diz respeito a algo de código

Tipos são customizáveis

```toml
[[tool.towncrier.type]]
directory = "security"
name = "Security"
showcontent = true

[[tool.towncrier.type]]
directory = "removed"
name = "Removed"
showcontent = true

[[tool.towncrier.type]]
directory = "deprecated"
name = "Deprecated"
showcontent = true
```

Você pode adicionar tipos customizados ao towncrier via pyproject.

O único ponto é que quando um único tipo é definido, ele para de apresentar os defaults.

Build

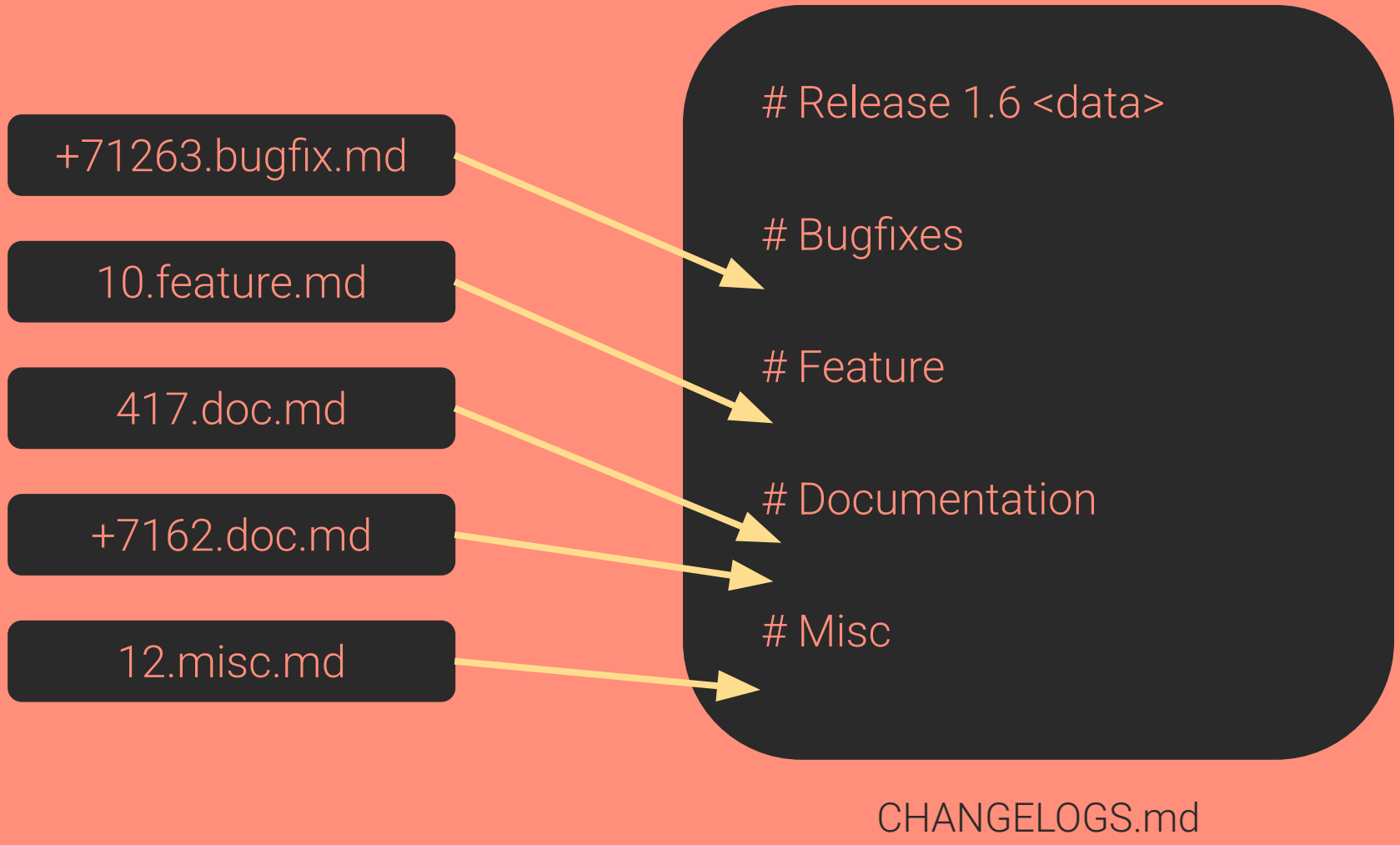
Pra criar o arquivo final de release com todos os fragmentos, usamos:

```
towncrier build --version 1.0.1
```

Caso o pacote tenha um __version__, e a opção de pacote seja configurada no pyproject a versão será detectada automaticamente

PS: ele não pega a chave project.version do pyproject

Fragmentos


+71263.bugfix.md
10.feature.md
417.doc.md
+7162.doc.md
12.misc.md
Release 1.6 <data>
Bugfixes
Feature
Documentation
Misc
CHANGELOGS.md

Arquivo final

Algumas coisas ainda precisam ser ajustadas, afinal... Nosso projeto não é o twisted. Mas os fragmentos foram adicionados!

```
## [1.0](https://github.com/twisted/my-project/tree/1.0) - 2025-01-25

### Added

- Evento de teste

### Fixed

- Um fix qualquer
```

Estilo

Existem alguns campos que podem nos ajudar a pré-formatar o documento:

- **issue_format**: Formatação para links de issues
  - quando diferente de +
- **title_format**: Formatação para o título da release
- **template**: Caminho do arquivo jinja com o template da release
- Mais opções...

```toml
[tool.towncrier]
package = "live_278"
directory = "changelog.d"
filename = "CHANGELOG.md"
title_format = "## [{version}](https://github.com/dunossauro/projeto/tree/{version}) - {project_date}"
issue_format = "[#{issue}](https://github.com/dunossauro/projeto/issues/{issue})"
template = "changelog.d/_template.md"
```

https://towncrier.readthedocs.io/en/stable/configuration.html#top-level-keys

Exemplo com template Jinja

A issue: https://github.com/twisted/towncrier/issues/454

O exemplo: https://github.com/twisted/towncrier/blob/trunk/src/towncrier/templates/default.md

Documentação

Juntando tudo

Changelog na documentação

Para exibir o changelog na documentação, isso vai depender de qual biblioteca de documentação estamos usando. Farei um exemplo com **mkdocs**, mas é possível fazer com sphinx também:

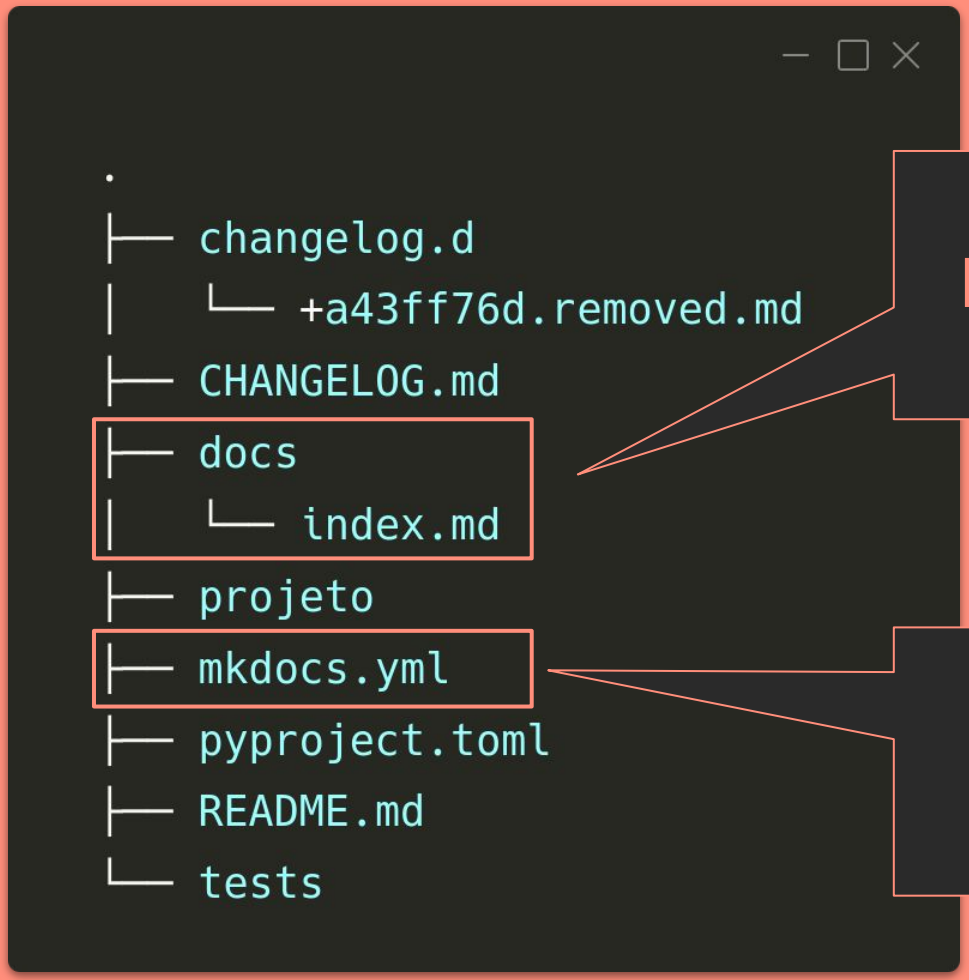
```
pip install
    mkdocs  # Lib
    pymdown-extensions  # Extensão de formatação
    mkdocs-towncier  # Extensão para o towncrier
```

Criando um projeto mkdocs

Entrando no diretório do projeto podemos criar um projeto com o comando:

```
mkdocs new .
```

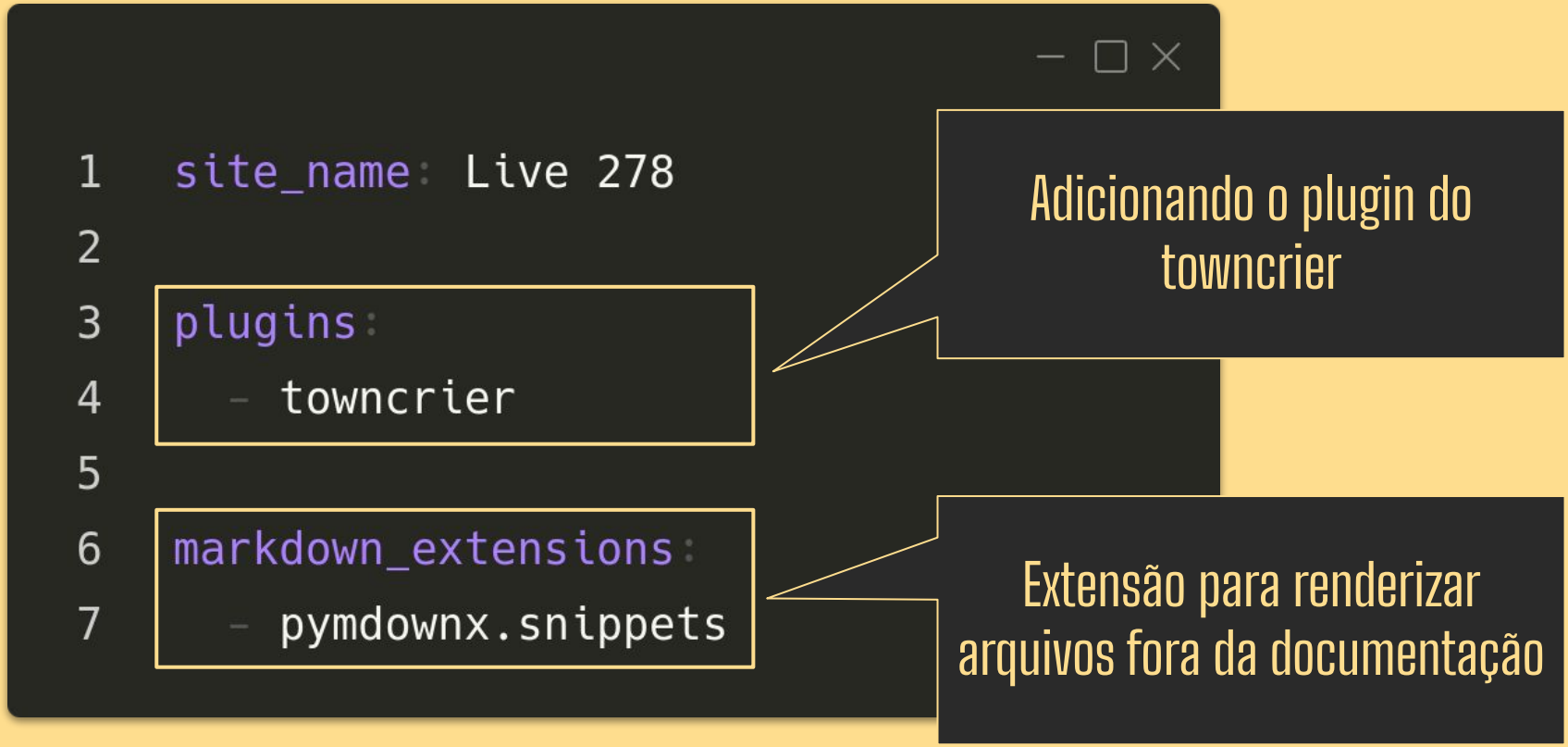
A nossa estrutura ficará assim



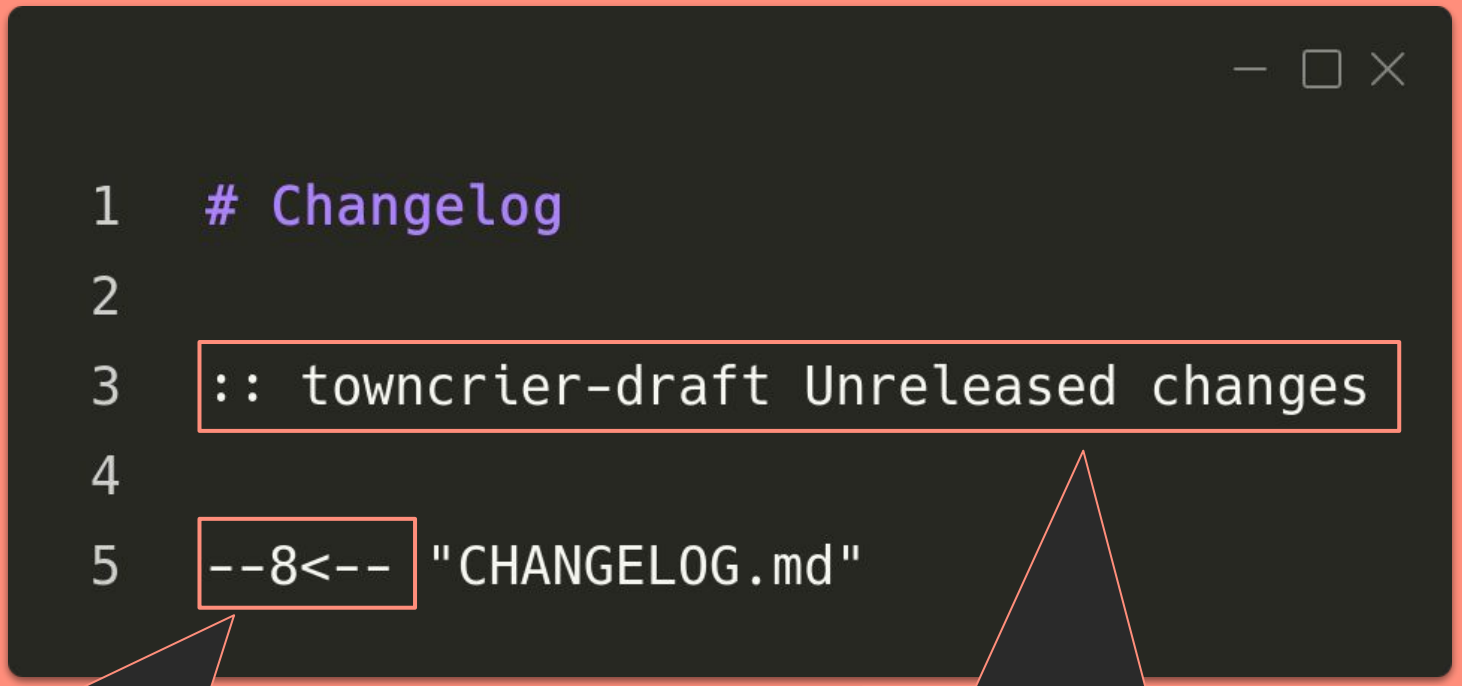
Diretório da documentação

Configuração do mkdocs

mkdocs.yaml


site_name: Live 278
plugins:
- towncrier
markdown_extensions:
- pymdownx.snippets
Adicionando o plugin do towncrier
Extensão para renderizar arquivos fora da documentação

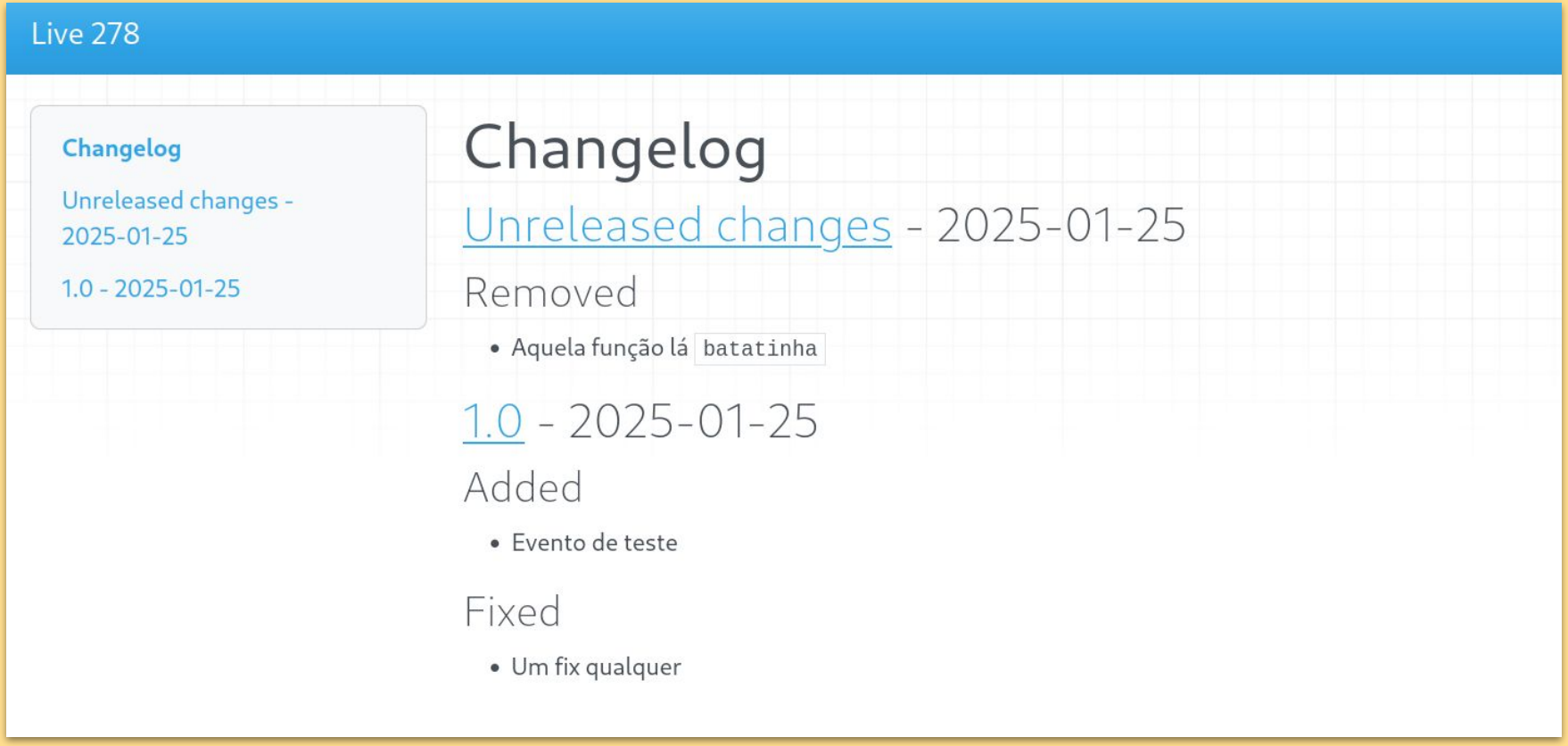
Arquivo markdown da documentação



Carrega um arquivo externo

Carrega os fragmentos

Resultado



http://127.0.0.1:8000/

Referências e links

- Python cases
  - Lançamento: https://pythoninsider.blogspot.com/2025/01/python-3140-alpha-4-is-out.html
  - Notas da release: https://docs.python.org/3.14/whatsnew/3.14.html
  - Logs de alteração: https://docs.python.org/3.14/whatsnew/changelog.html#python-3-14-0-alpha-4
- Libs do ecossistema
  - FastAPI (misto: Notes/Logs): https://fastapi.tiangolo.com/release-notes/
  - Mypy (misto: Notes/Logs): https://mypy-lang.blogspot.com/
  - Material Mkdocs (changelogs bruto): https://squidfunk.github.io/mkdocs-material/insiders/changelog
  - Pytest (changelogs com um Q de notes): https://docs.pytest.org/en/stable/changelog.html
- Changelogs
  - https://en.wikipedia.org/wiki/Changelog
  - https://keepachangelog.com/en/1.1.0/
  - https://common-changelog.org/
  - GNU (6.8): https://www.gnu.org/prep/standards/html_node/index.html#SEC_Contents
  - https://hi.ducalis.io/changelog/release-notes-best-practices
  - https://news.ycombinator.com/item?id=32122028
- Towncrier termo
  - https://dictionary.cambridge.org/dictionary/english/town-crier
  - https://pt.wikipedia.org/wiki/Arauto
- Towncrier ferramenta
  - https://towncrier.readthedocs.io/en/stable/index.html
  - https://github.com/davfsa/mkdocs-towncrier
  - https://github.com/twisted/towncrier/issues/454
  - https://github.com/twisted/towncrier/blob/trunk/src/towncrier/templates/default.md
- Meu fluxo de ideias: https://excalidraw.com/#json=aQQTPahttlUvnnZFWiFe2,ofsCaovoR6Afnxqx8Ab1mA

apoia.se/livedepython

pix.dunossauro@gmail.com

patreon.com/dunossauro

Ajude o projeto <3