

# Release it!

Versionamento e geração de releases automáticos

# Introdução

Antes de falarmos sobre a ferramenta *Release It!*, precisamos entender algumas coisas: O que é **versionamento** e o que são **releases**.

# Versionamento

Versionamento é uma metodologia de classificação adotada por programadores com o intuito de controlar e acompanhar o histórico de alterações em um software. Essa metodologia permite distinguir as mudanças realizadas, facilitando a identificação de cada uma delas. Atualmente uma metodologia muito utilizada é o **Versionamento Semântico**.

# Versionamento Semântico

O Versionamento Semântico é um conjunto simples de regras que ditam como os números de versão são atribuídos e incrementados.

# Regras do Versionamento Semântico

Diante do número de versão (ex: 1.0.0) incremente-o utilizando as regras: (1)**MAJOR**.(0)**MINOR**.(0)**PATCH**:

1. **MAJOR**: Quando há funcionalidades incompatíveis com as versões anteriores do sistema;
2. **MINOR**: Quando há funcionalidades adicionadas ao sistema porém com compatibilidade com as versões anteriores;
3. **PATCH**: Quando há uma ou mais correções em relação as funcionalidades existentes.

Para mais informações consulte a documentação oficial: <https://semver.org/>

# Releases

Releases são versões de um software que podem ser disponibilizados para um público em geral. Geralmente é disponibilizado por plataformas de hospedagem de código, como: *GitHub*, *GitLab* ou *Bitbucket*. Aqui nessa apresentação vamos abordar o **GitHub Releases**.

## GitHub Releases


A página de releases nos projetos hospedados no GitHub é um local que contém todo o histórico do projeto, com todas as suas versões e notas de lançamento.

# GitHub Releases

ReleasesTags

Find a release

6 days ago

 webpro

15.4.2


1b7f287

Compare


Release 15.4.2 Latest

- Update dependencies ( [97095d5](#) )
- Defer dry run bail out in asset globbing (to include the warning in dry runs) ( [bf6ccc8](#) )
- Housekeeping for Actions ( [feff2eb](#) )


Assets 2

 Source code (zip)


6 days ago

 Source code (tar.gz)

6 days ago



21 days ago

 webpro

15.4.1


2f565d0

Compare

Release 15.4.1

- Handle file paths and dots in git urls ( [055a4ff](#) )
- Update dependencies (including git-url-parse) ( [1851650](#) )

Assets 2



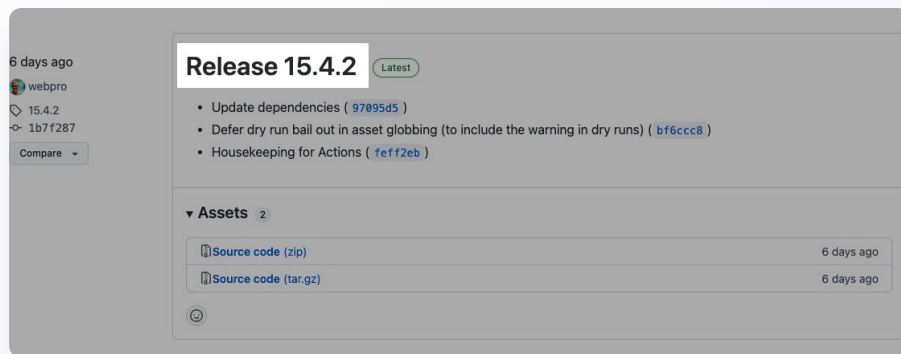


## Quais são as propriedades de um release?

- Título
- Notas de lançamento
- Assets (arquivos binários)
- Git Tag
- Autor

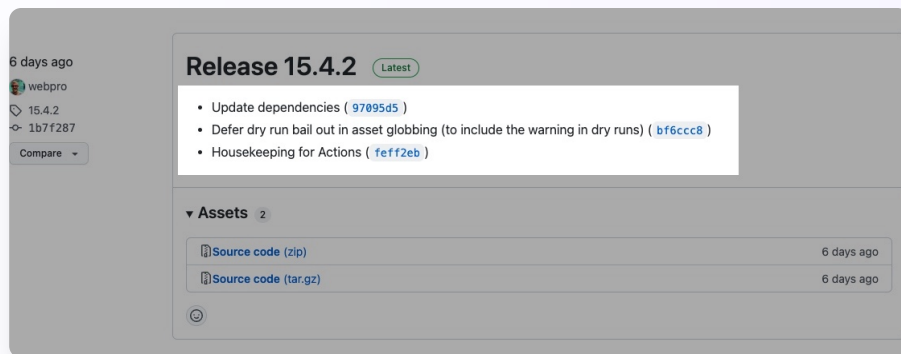
# Título

Utilizado como um identificador do release



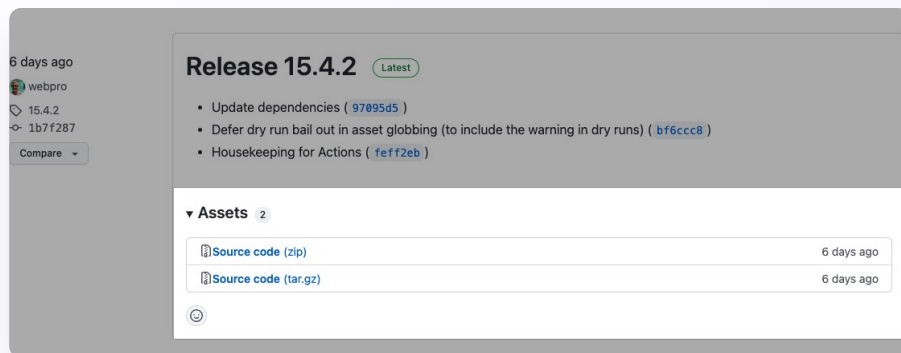
# Notas de lançamento

Abrange todas as alterações realizadas naquele release



# Assets


Qualquer arquivo que pode ser disponibilizado para download





# Autor

Usuário que criou o release

6 days ago

 webpro

 15.4.2


 1b7f287


Compare ▾


## Release 15.4.2 Latest

- Update dependencies ( [97095d5](#) )
- Defer dry run bail out in asset globbing (to include the warning in dry runs) ( [bf6ccc8](#) )
- Housekeeping for Actions ( [feff2eb](#) )

▼ Assets 2

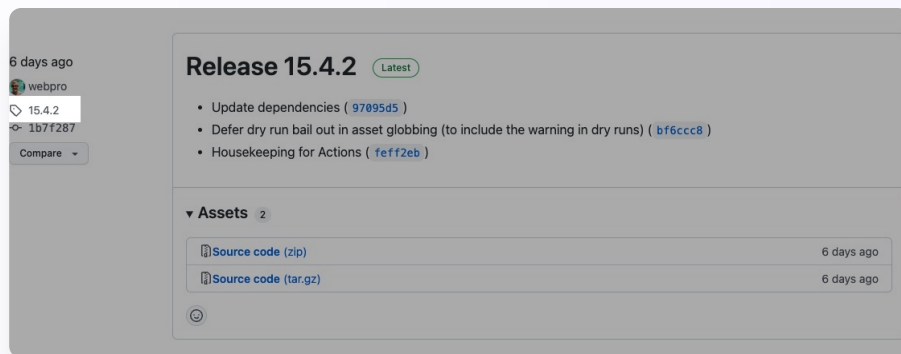
 Source code (zip) 6 days ago

 Source code (tar.gz) 6 days ago



# Tag do Git

Uma tag do Git é vinculada ao release



## Ok... mas por quê criar um release?

Um release informa por meio das notas de lançamento mudanças significativas realizadas em cada versão de um projeto, ou seja, facilitando que usuários e contribuidores vejam precisamente quais mudanças foram realizadas entre cada versão publicada.

## e mais...

Para os contribuidores do projeto, o vínculo entre o release e a tag do Git pode auxiliar em muitos casos. A tag indica um determinado estado do código e esse estado sempre pode ser recuperado, não importa quantos commits e branches seguem, esse estado é definitivo.



## Uma tag do Git pode ser utilizada para:

- Obter uma versão específica do projeto;
- Distribuir essa versão como um pacote;
- Realizar deploy dessa versão em um servidor;
- Reverter uma versão com falha.

**Sabendo de tudo isso, já podemos criar nosso release...**

## Criando um novo release

Criar um novo release manualmente pode ser um pouco tedioso pois envolve um número considerável de etapas:

- Realizar a atualização da versão em alguns arquivos (ex: *package.json*, *composer.json*, entre outros...);
- Realizar tarefas pré-release, como o lint do projeto e/ou testes automatizados antes da publicação;
- Realizar a compilação/build do código;
- Criar uma tag no Git;
- Dar commit e realizar o push de todas essas modificações para finalmente gerar o release...

## Aí que entra o Release It!

Essa ferramenta nos auxilia a automatizar as etapas chatas para criar um release. Ele executa as etapas ditas anteriormente por meio de uma CLI com a sua intervenção em cada etapa ou de forma completamente automatizada.

## Alguns recursos do Release It! são:

- Atualização de versão em arquivos específicos (ex: *package.json* ou em qualquer arquivo utilizando o plugin `@release-it/bumper`);
- Criação de tags no Git utilizando o Versionamento Semântico;
- Criação de releases automáticos no GitHub ou GitLab;
- Criação das notas de lançamento automaticamente com base nos commits ou seguindo o padrão *Keep A Changelog*;
- Publicação automática de pacotes no *NPM*;
- Possibilidade de criar *hooks* para automação de tarefas pré-release ou pós-release;
- e muito mais...

# Instalação

1. Certifique-se de ter instalado o Node  $\geq 14$  em sua máquina:

```
$ node -v  
> v14.20.0
```

2. Adicione as dependências ao projeto:

```
$ npm install release-it @release-it/keep-a-changelog -D
```

3. Pronto! Prossiga agora para a etapa de **Configuração** →

# Configuração

1. Dentro do arquivo *package.json* procure pela propriedade *scripts* e adicione o script:

```
"scripts": {  
  "release": "release-it"  
}
```

2. Após, vamos criar um arquivo *.release-it.json* na raiz do projeto e preencher o conteúdo do arquivo com as configurações da ferramenta.
3. Para demonstração vamos utilizar algumas configurações básicas. Veja mais informações sobre essas configurações →

# Configuração para o Git

```
"git": {  
  "requireBranch": "release",  
  "requireCleanWorkingDir": true,  
  "commitMessage": "Release v${version}",  
  "commitArgs": [],  
  "tag": true,  
  "tagName": "${version}",  
  "tagAnnotation": "Release v${version}"  
}
```



## Configuração para o GitHub

```
"github": {  
  "release": true,  
  "releaseName": "v${version}"  
}
```

# Configuração dos hooks

```
"hooks": {  
  "before:init": ["npm run lint"],  
  "after:release": [  
    "echo Updating and pushing changes to main...",  
    "git checkout main",  
    "git fetch origin main",  
    "git merge --no-ff release",  
    "git push origin main",  
  
    "echo Updating and pushing changes to develop...",  
    "git checkout develop",  
    "git fetch origin develop",  
    "git merge --no-ff release",  
    "git push origin develop",  
  
    "echo Successfully released ${name} v${version} to ${repo.repository}!"  
  ]  
}
```

# Configuração das notas de lançamento

```
"plugins": {  
  "@release-it/keep-a-changelog": {  
    "filename": "CHANGELOG.md"  
  }  
}
```

**Atenção:** Na raiz do projeto crie um arquivo *CHANGELOG.md* e preencha o conteúdo deste arquivo seguindo o padrão do Keep A Changelog. Para criar um novo release é **obrigatório** ter uma seção *[Unreleased]* com as notas de lançamento da nova versão.

Veja um exemplo →

# CHANGELOG.md

## # Changelog

Todas as mudanças notáveis neste projeto serão documentadas neste arquivo.

O formato é baseado em [\[Keep a Changelog\]\(https://keepachangelog.com/pt-BR/1.0.0/\)](https://keepachangelog.com/pt-BR/1.0.0/), e este projeto adere ao [\[Semantic Versioning\]\(https://semver.org/spec/v2.0.0.html\)](https://semver.org/spec/v2.0.0.html).

## ## [Unreleased]

### ### Adicionado

- Adicionada biblioteca `release-it` e plugin `@release-it/keep-a-changelog`.

## ## [1.0.0] - 2022-09-20

### ### Adicionado

- Primeira versão do projeto.

# Pronto! Agora já podemos gerar o release.

Execute o comando abaixo para executar a CLI com intervenção em cada etapa:

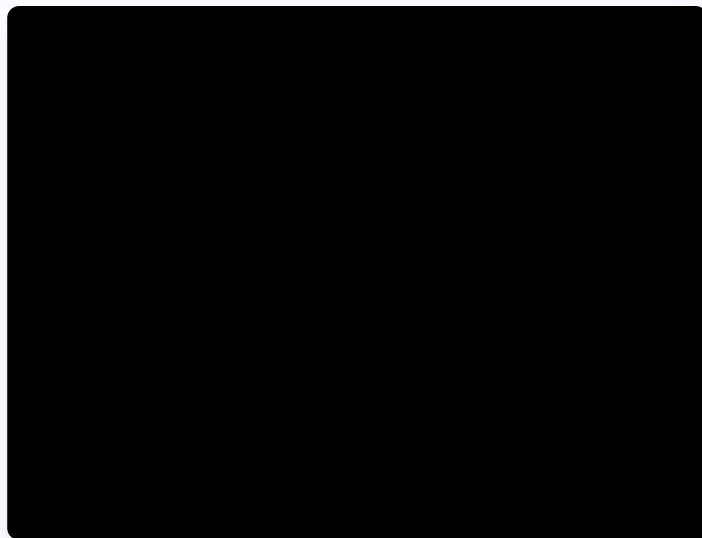
```
$ npm run release
```

Ou execute o comando abaixo para executar de forma completamente automatizada:

```
$ npm run release -- minor --ci
```

# Se tudo der certo...

Você irá ver algo parecido com:



**Agora seu projeto está pronto para gerar releases automáticos!**

# Obrigado!

Documentação 