

NExT

Nova Experiência de Trabalho

Ellen Coelho
ecxc@cesar.org.br

Ronnie Santos
ress@cesar.org.br



Agenda

1ª Aula (07/11)

Introdução a
Gerência de
Configuração

Introdução ao
Git

Comandos Git

2ª Aula (08/11)

Explorando o
GitHub

Criando
Read.me

Criando
Repositório e
adicionando
projetos ao
Github

3ª Aula (09/11)

**Aplicação do
Fluxo de
Trabalho no
GitHub**

**Mesclando
Branches**

Review Process


Conflitos

4ª Aula (10/11)

Gerência de
configuração e
Qualidade de
software

Integração
contínua

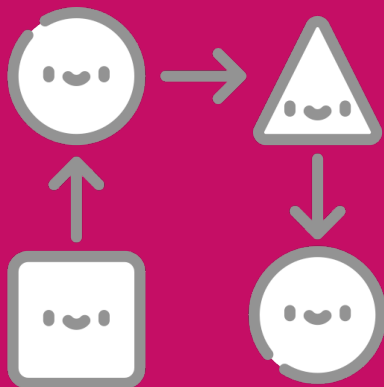
Testes
automatizados
e Github

Todo nosso material de aula, exercícios e chamada vão ser disponibilizados diariamente via Google Classroom 

Classroom

Código da sala:

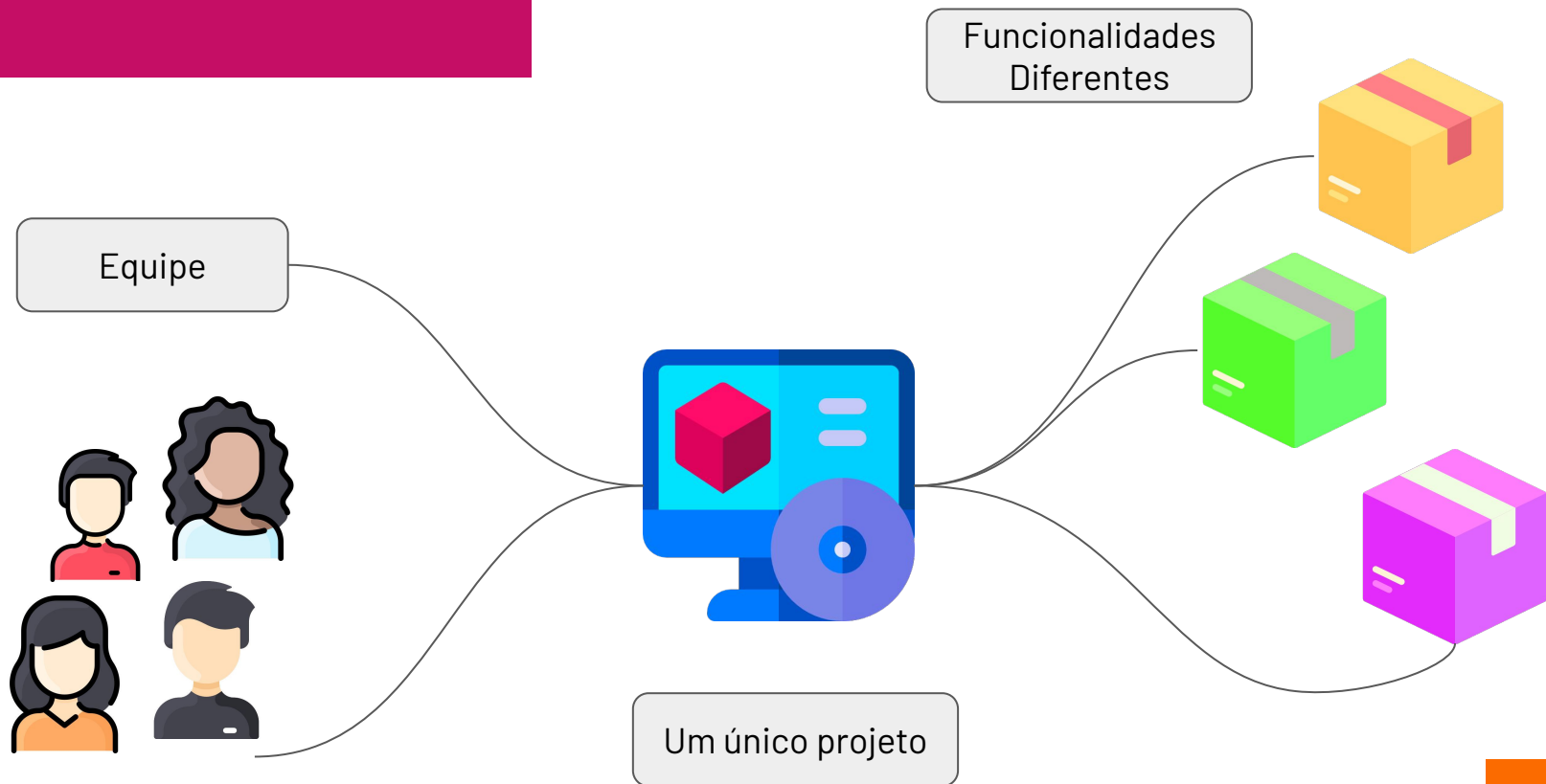
qjr4fg2



Workflow

Workflow

@CESAR 2022 | Todos os Direitos Reservados



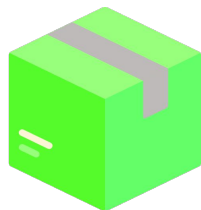
Branch

Branch, em tradução literal, significa “ramo”.

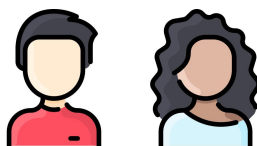
No mundo da programação, ela tem o mesmo significado: uma **branch** é uma ramificação do seu projeto.



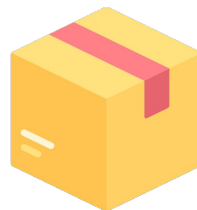
Atribuições



Feature 1
Funcionalidade de
Cadastrar Usuário



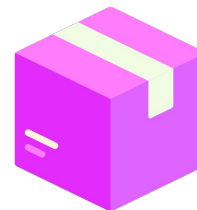
2 pessoas
responsáveis



Feature 2
Funcionalidade de
Listar Usuários



1 pessoa
responsável



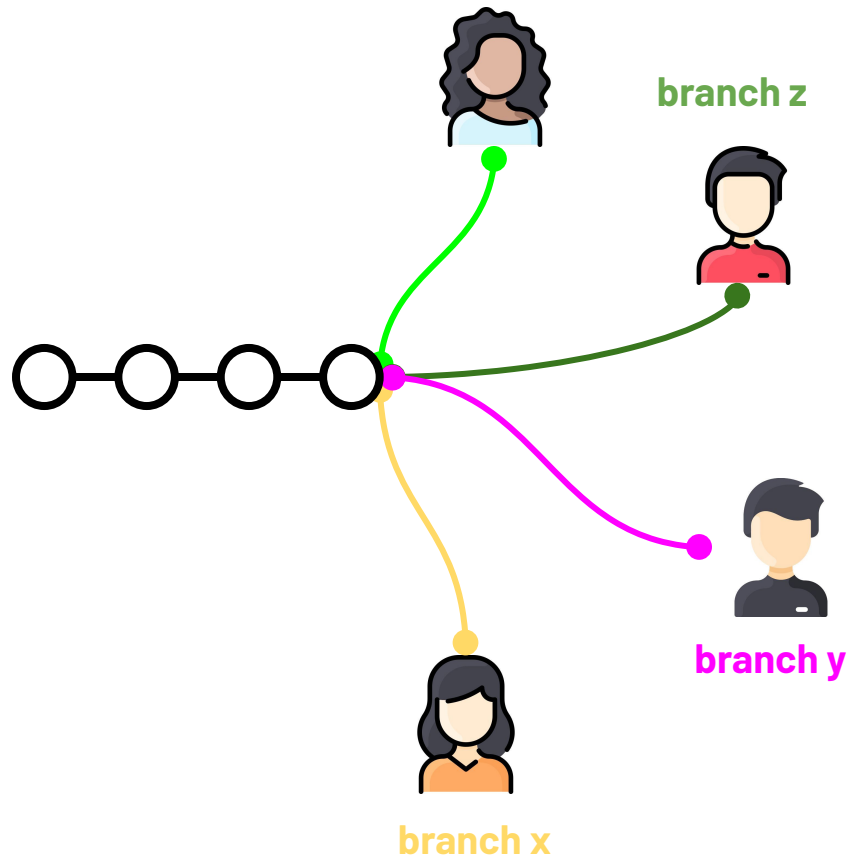
Feature 3
Funcionalidade de
Deletar Usuário



1 pessoa
responsável

Branches

main

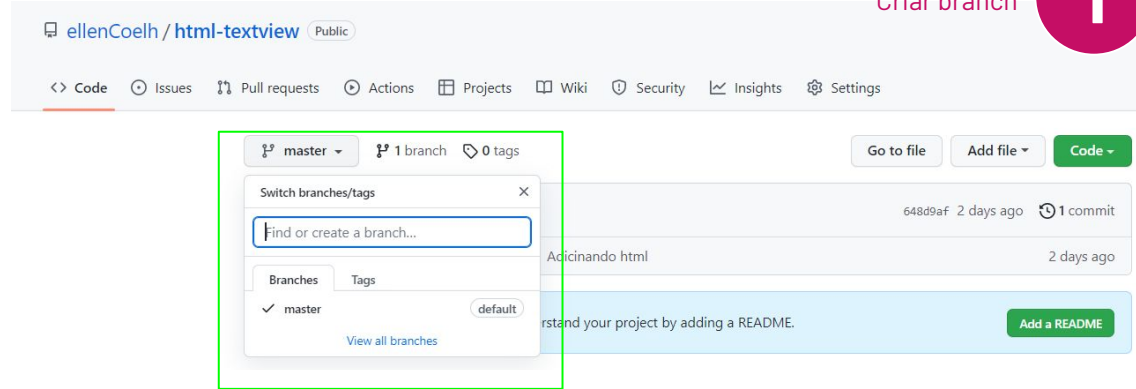


Criando Branches

Pelo GitHub

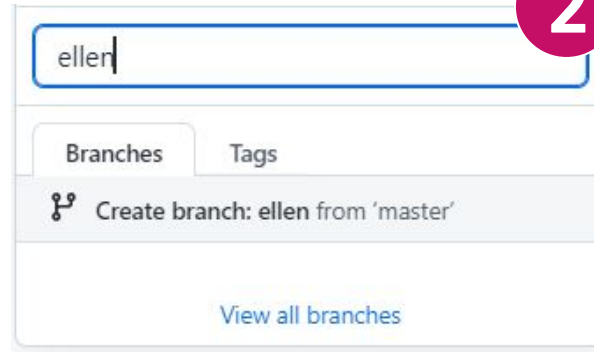
Criar branch

1



Inserir nome

2



Atualizar e Validar criação (localmente)

3

```
git fetch
git branch -- all
```



Por linha de código



1. **Mude para branch que deseja utilizar como base, será a partir dela que será criado uma nova branch**

```
git checkout "Nome da branch"
```

2. **Crie a nova branch**

```
git branch -M "Nome da nova branch"
```

3. **Valide a criação**

```
git branch -- all
```

4. **Enviar para o GitHub**

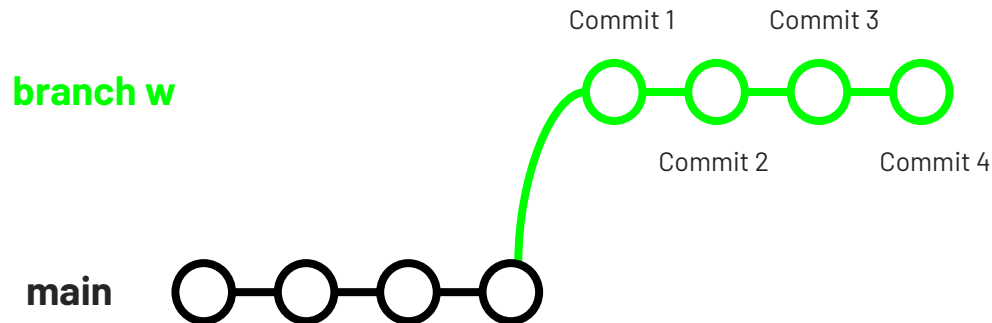
```
git push -u origin "Nome da nova branch"
```

5. **Valide a criação no GitHub - Atualização da Página**

Criando Branches

To practice 

Commits



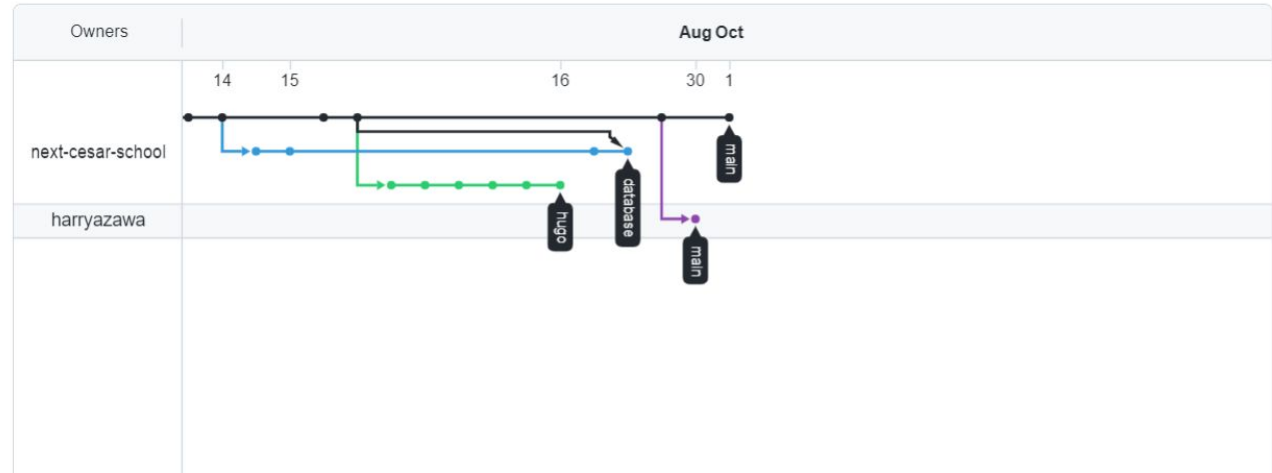
Cada comentário, commit, adicionado gera um ponto histórico

Graph

[Code](#)[Issues](#)[Pull requests](#) 1[Actions](#)[Projects](#)[Security](#)[Insights](#)[Pulse](#)[Contributors](#)[Community Standards](#)[Commits](#)[Code frequency](#)[Dependency graph](#)[Network](#)[Forks](#)

Network graph

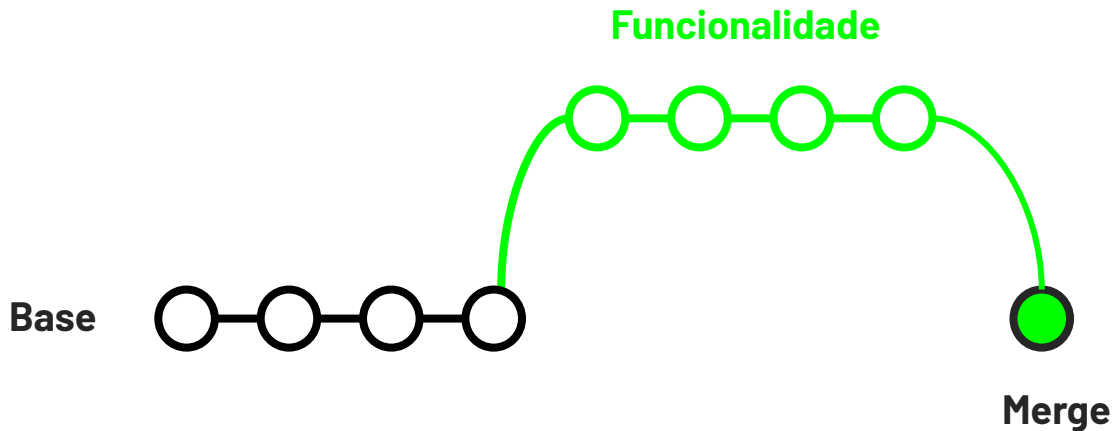
Timeline of the most recent commits to this repository and its network ordered by most recently pushed to.



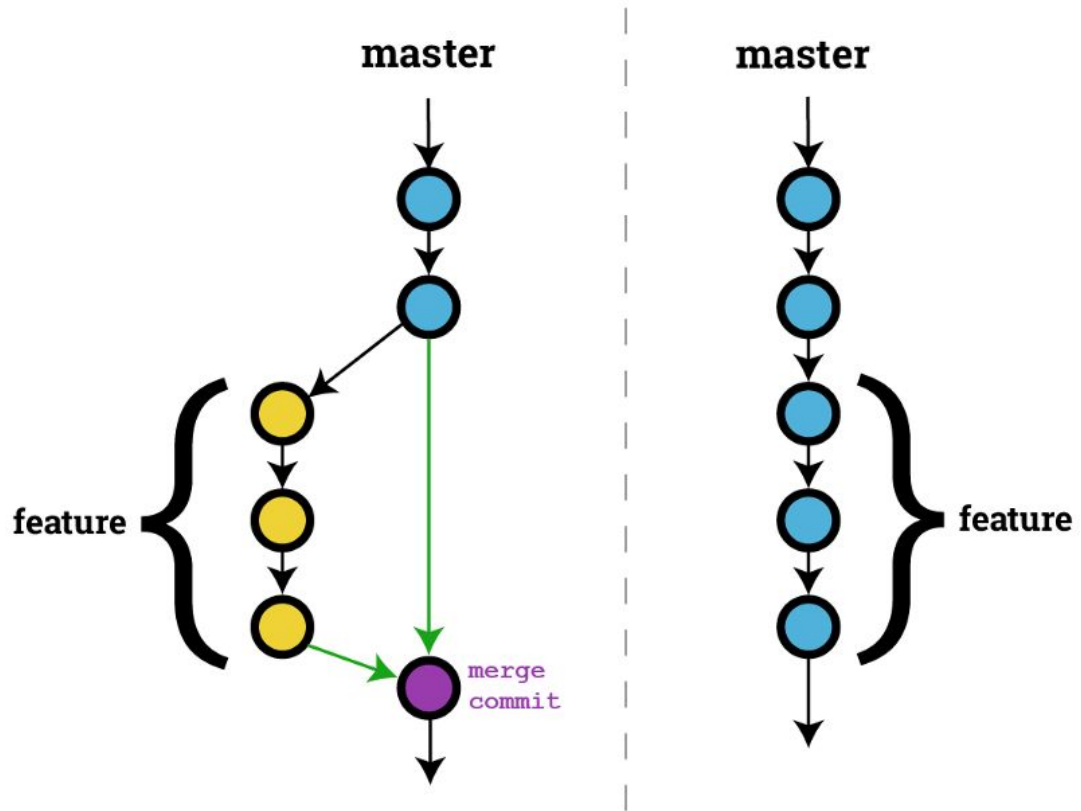
Merge

Merge

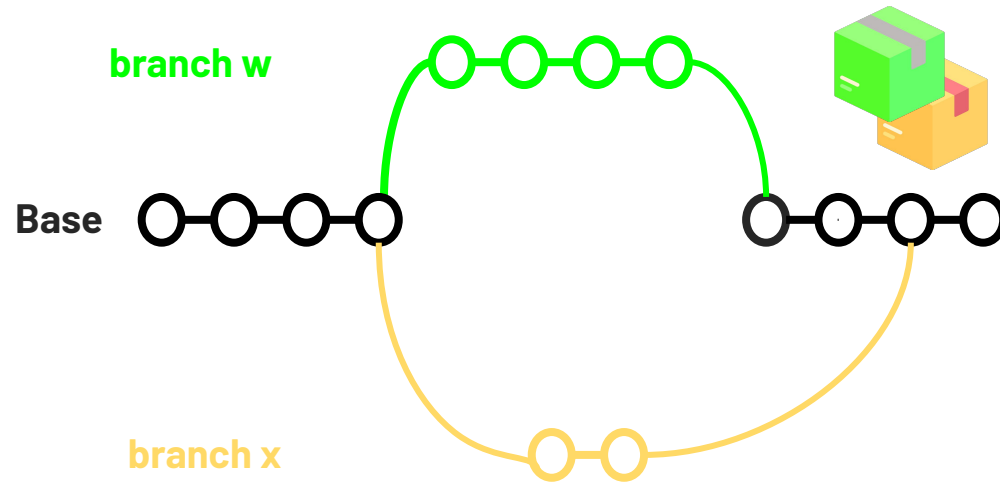
Mesclagem é o jeito do Git de unificar um histórico bifurcado. O processo de mesclagem é utilizado para combinar dois branches.



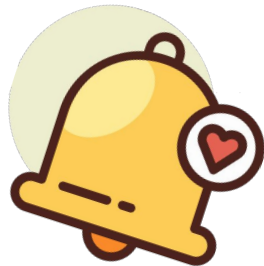
Merge



Merge



Lembrete



Dê um clone no repositório

`https://github.com/ellenCoelh/html-textview`

Merge

1. **Crie uma nova branch a partir da main;**
2. **Faça uma edição na branch;**
Edite um arquivo `aluno.txt`, com seu nome;
3. **Mescle essa nova branch com a branch *"pratica_merge"*.**
Verifica-se os arquivos da outra branch foram mesclados.

Merge

Mescle sua branch com a branch de um colega!

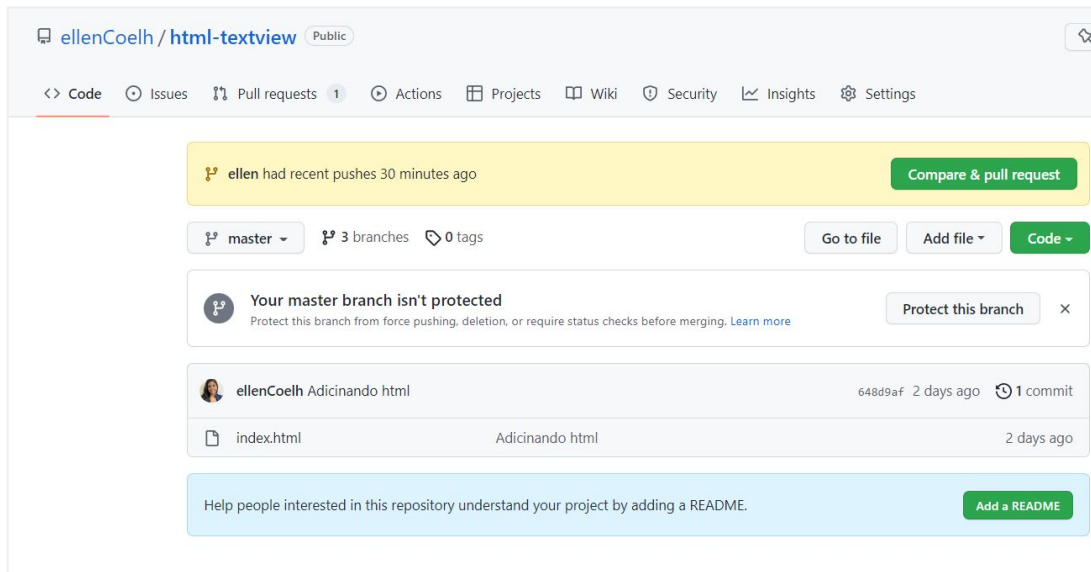
1. `git branch -- all`
2. `git checkout "Nome Da Branch do Colega"`
3. `git pull`
4. `git checkout "Nome Da Sua Branch"`
5. `git merge origin/"Nome Da Branch do Colega"`

Observação: Lembre-se de inserir a mesma referência da branch remota, normalmente vem com o **origin/**.

Pull Request

Pull Request quer dizer solicitação de puxa.

E isso está diretamente ligado ao fato de que, ao enviar a notificação, às demais pessoas desenvolvedoras saberão que precisam fazer o **merge** do código em uma branch.



Selecionando as branches



Destino: Para onde desejo adicionar minhas alterações.



Origem (Branch de trabalho - Local): Onde realizei minhas alterações (Branch que eu estava trabalhando)

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

base: master ← compare: ellen ✓ Able to merge. These branches can be automatically merged.

Adicionando novo html

Write Preview

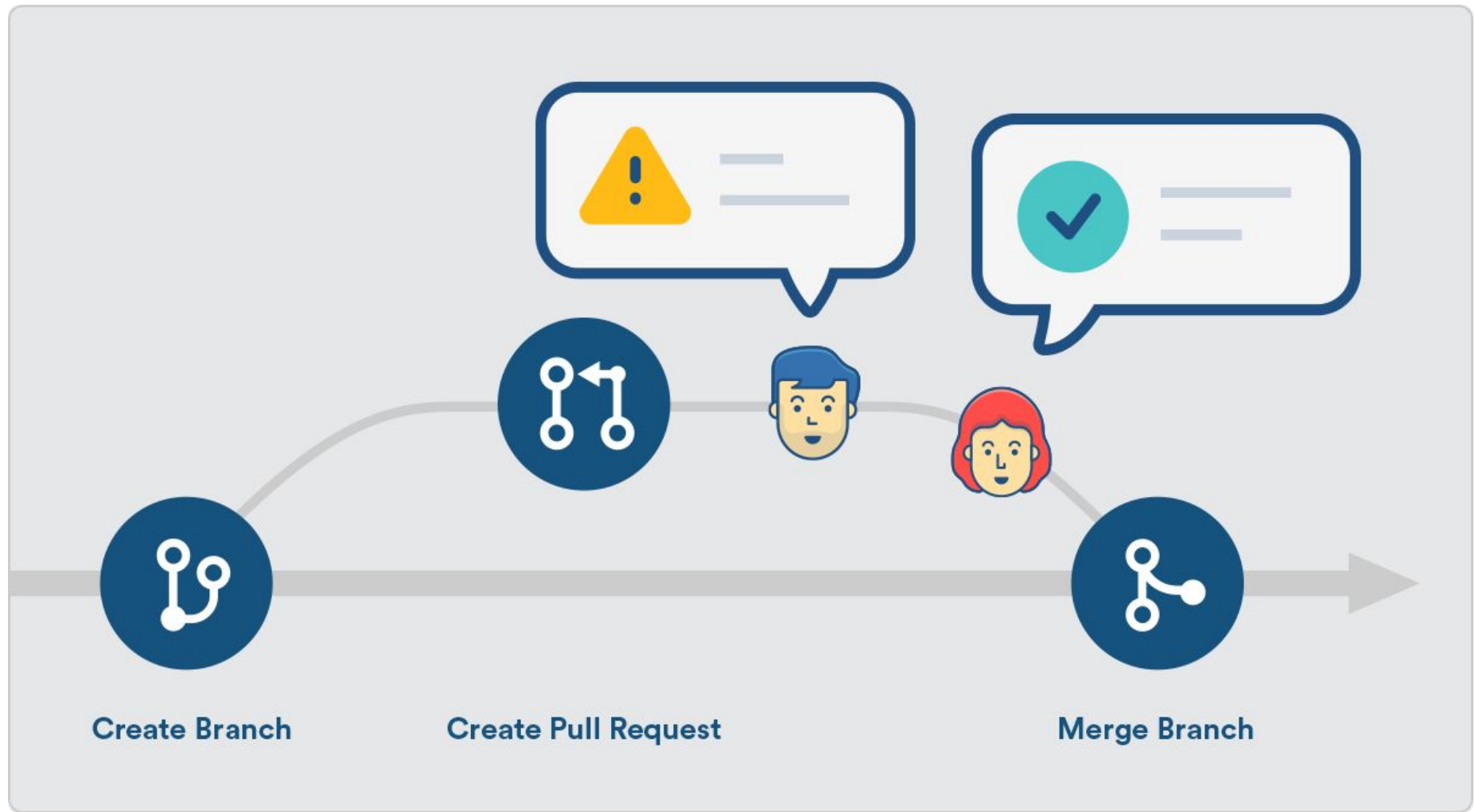
Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Create pull request

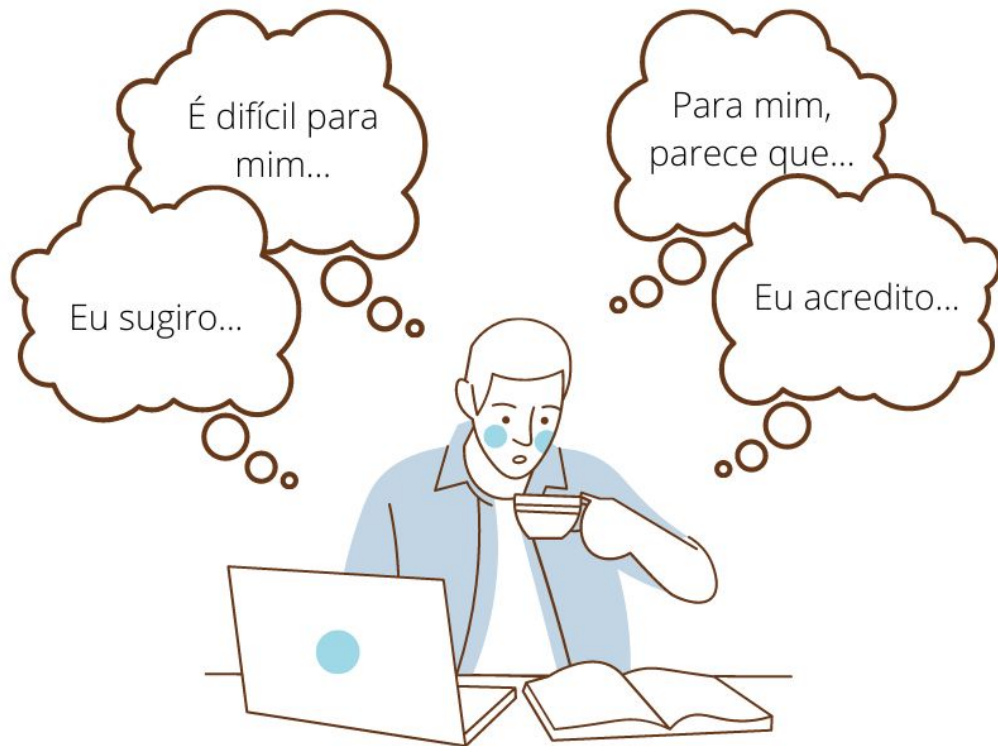
Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Code Review



Feedback da revisão de código

- Formule o feedback do seu ponto de vista, expressando seus pensamentos, sentimentos e impressões pessoais.
- Deve ser construtivo e os comentários devem ser sobre o código, não pessoais sobre o autor.
- O revisor deve adotar um tom sugestivo e positivo, não deve gerar conflitos, forneça links ou exemplos para explicações detalhadas de um problema.



Pull Request & Code Review

To practice 

Pull Request

To practice 

1. **Salve as alterações e suba-as**
2. **Abra um *Pull Request* para a branch de origem (main).**

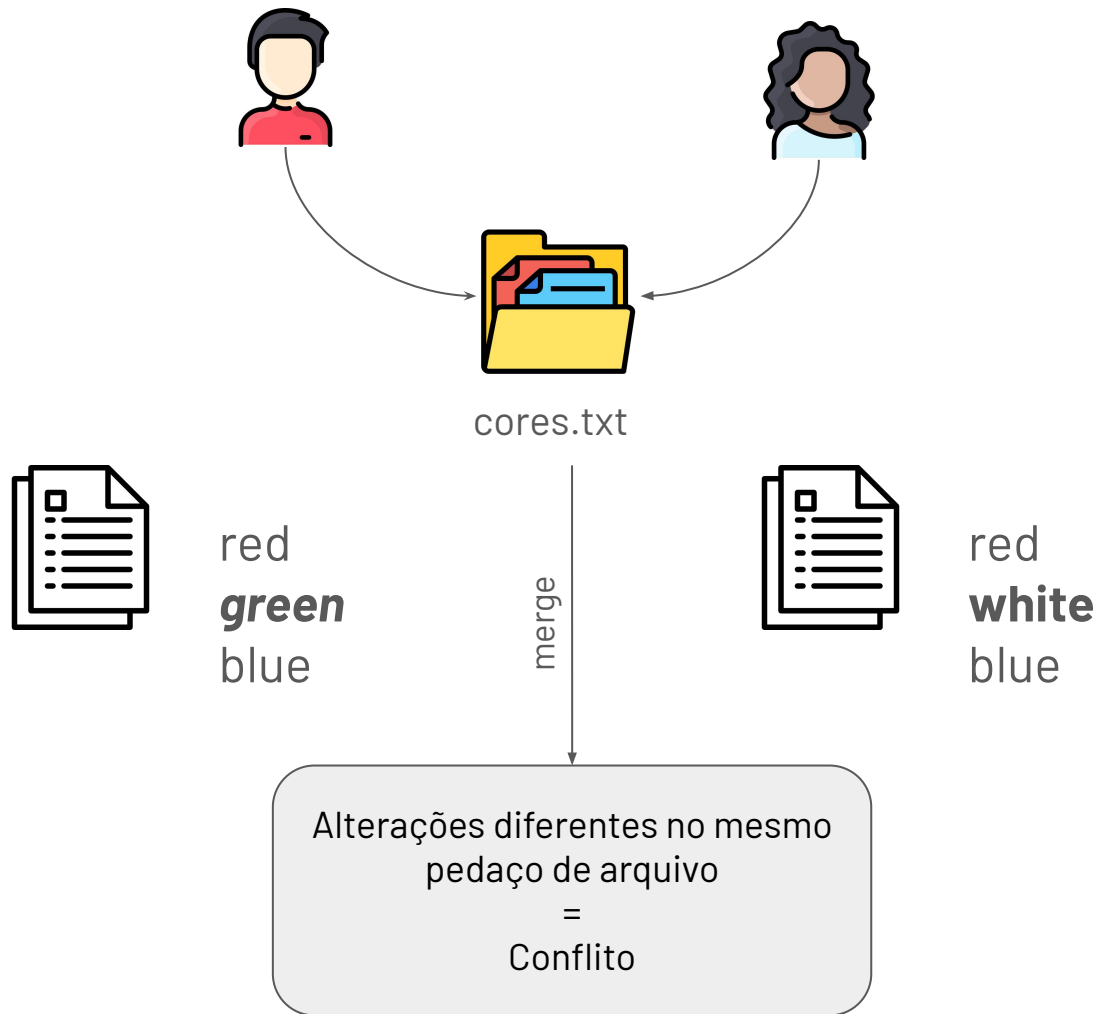
Pull Request

To practice 

**Repita o mesmo processo anterior
para uma branch de algum colega.**

Conflitos

Conflito



colors.txt ×

src > colors.txt

1 red

Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes

2 <<<<<< HEAD (Current Change)

3 green

4 =====

5 white

6 >>>>>> his-branch (Incoming Change)

7 blue

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

react-app-demo my-branch git merge his-branch

Auto-merging src/colors.txt

CONFLICT (content): Merge conflict in src/colors.txt

Automatic merge failed; fix conflicts and then commit the result.

react-app-demo my-branch >M<

1

Accept Current Change

=

Aceitar a mudança atual

2

Accept Incoming Change

=

Aceitar alteração recebida

3

Accept Both Changes

=

Aceitar ambas as
alterações

4

Compared Changes

=

Comparar Alterações

Conflito

É comum que surjam conflitos quando duas pessoas alteram as mesmas linhas no mesmo arquivo.

Ou então quando algum desenvolvedor exclui arquivos enquanto outra pessoa faz alterações.

Nesses casos, o Git não pode determinar qual está correto

O Git marca os arquivos em conflito e interrompe o processo de merge. A partir daí, é responsabilidade dos desenvolvedores resolver o conflito.

Resolvendo Código

To practice 

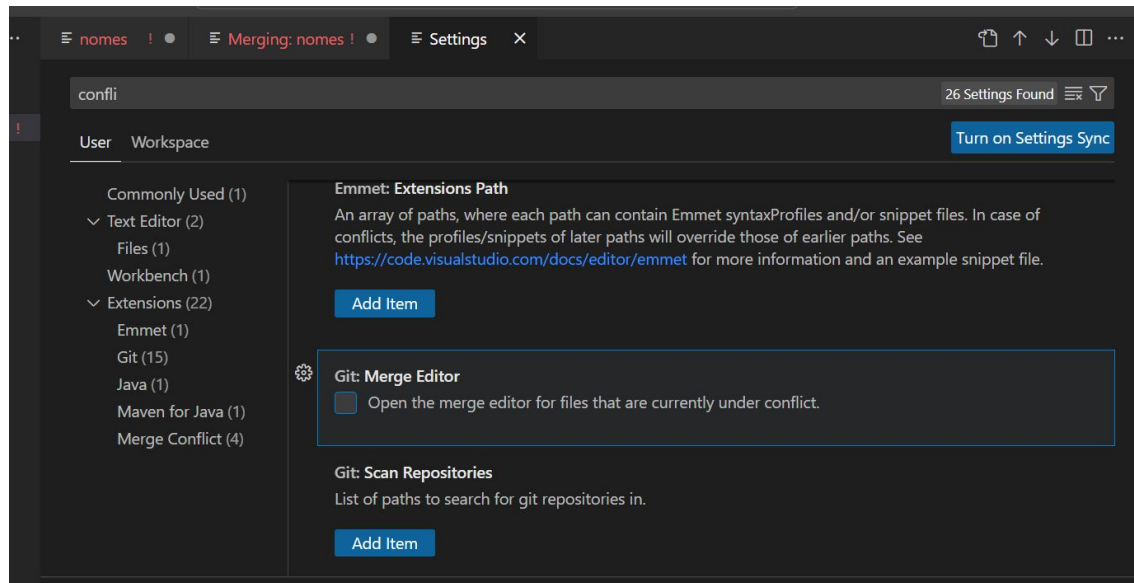
Resolvendo Código

To practice 

1. **Crie uma nova branch a partir da main;**
`git branch conflito_SEU_NOME`
2. **Crie um arquivo chamado *nomes* e insira seu nome;**
3. **Mude para branch *nomes* e mescle com a branch que sofreu alteração;**
4. **Resolva os conflitos.**

```
names
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
1 <<<<<< HEAD (Current Change)
2  Insira seu nome
3  =====
4  Ronnie Santos
5  >>>>>> origin/nomes2 (Incoming Change)
6
```

Resolvendo Código | Configuração



Monitoria

Aula 3

1. Criação de branch (pelo github e por linha de código);
 - a. `git branch -all`
 - b. `git branch "NOVA_BRANCH"`
2. Alteração/Mudança entre branches;
 - a. `git checkout`
3. Mesclagem de branches;
 - a. `git merge`
4. Abrir pull request no GitHub;
5. Realizar Code Review;
6. Resolver Conflitos.



C . E . S . A . R

Pessoas impulsionando inovação.
Inovação impulsionando negócios.

