Tutorial de Fluxo de Trabalho com Git – IonicNews

Este tutorial descreve os principais comandos Git utilizados no fluxo de trabalho do projeto **IonicNews**.

Nosso fluxo é baseado na utilização de branches organizadas: `main`, `dev` e `feature`.

1. Clonar o repositório

```
git clone https://github.com/David-Evangelist/IonicNews.git
```

2. Configurar o usuário (caso ainda não tenha feito)

```
git config --global user.name "Seu Nome"
git config --global user.email "seu@email.com"
```

3. Criar uma branch para uma nova funcionalidade ou correção

```
# Trocar "minha-feature" pelo nome da sua funcionalidade
git checkout -b feature/minha-feature
```

4. Adicionar alterações para commit

```
git add .
```

Ou arquivos específicos:

```
git add src/app/novocomponente.ts
```

5. Realizar o commit seguindo o padrão

Padrão de commit:

`<tipo>: <descrição curta>`

Exemplo:

“feat: adiciona funcionalidade de favoritos”

Para realizar o commit:

```
git commit -m "feat: adiciona funcionalidade de favoritos"
```

6. Enviar a branch para o repositório remoto

```
git push origin feature/minha-feature
```

7. Atualizar a branch de desenvolvimento (`dev`)

Antes de começar uma nova feature, sempre atualize sua branch local:

```
git checkout dev
git pull origin dev
```

8. Fazer merge da feature para `dev`

Após o desenvolvimento e testes:

```
git checkout dev
git merge feature/minha-feature
```

Depois envie as alterações:

```
git push origin dev
```

9. Atualizar `main` após testes completos

Após validação e aprovação, atualize a `main` com as alterações estáveis:

```
git checkout main
git merge dev
git push origin main
```

10. Deletar branch de feature (opcional)

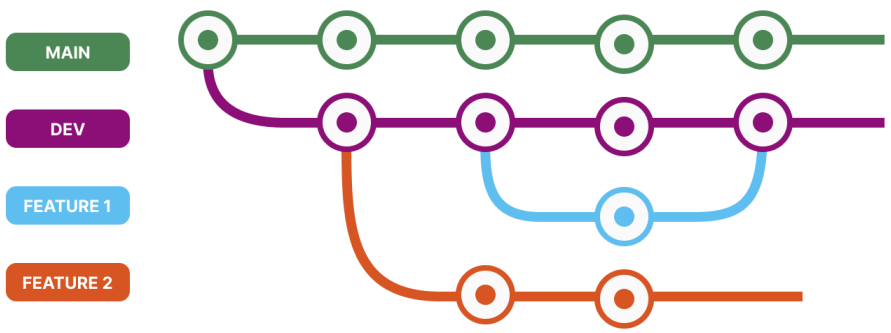
Depois que a branch de feature foi mergeada:

```
git branch -d feature/minha-feature
```

Resumo do fluxo:

1. Crie uma branch `feature/xyz`.
2. Desenvolva e commite com padrão.
3. Faça `push` para o remoto.
4. Abra um **Pull Request** para `dev`.
5. Após revisão, `dev` é mergeado na `main`.

Esquema visual do funcionamento das branches:



Padrão de branches:

Branch	Função
`main`	Produção, versão estável.
`dev`	Desenvolvimento e integração.
`feature/xyz`	Funcionalidades ou correções específicas.

Padrão de commits:

Tipo	Quando usar
`feat`	Nova funcionalidade
`fix`	Correção de bug
`docs`	Documentação
`style`	Ajustes de formatação
`refactor`	Refatoração
`test`	Testes
`chore`	Manutenção

Ferramentas utilizadas:

- **Git**: controle de versão.
- **GitHub**: repositório remoto.

Recomendações:

- Mantenha o repositório sempre atualizado (`git pull`).
- Faça commits frequentes e significativos.
- Escreva mensagens claras e objetivas.