

GitHub Workflow

Comment va-t-on s'organiser grâce à GitHub ?

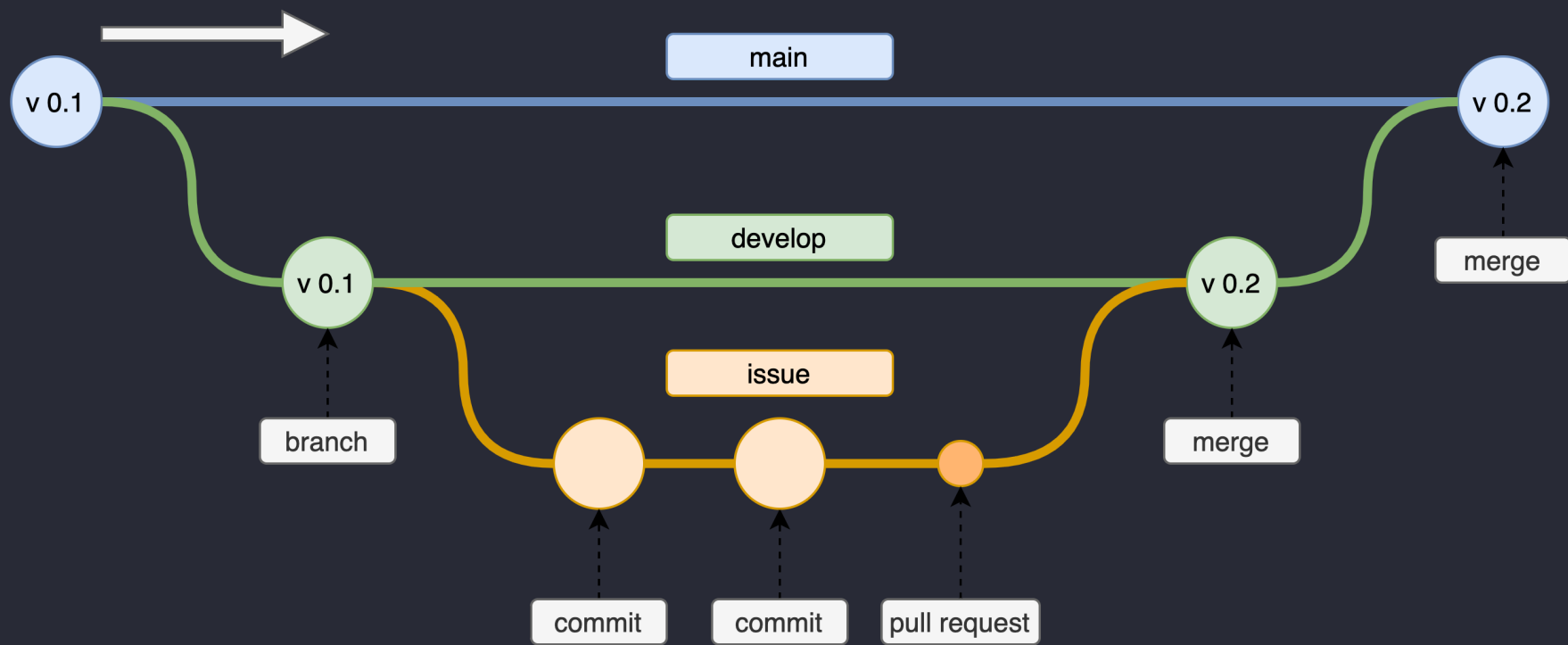
0 + Offrant

- 2 projets
 - Back-end
 - Front-end
- Même logique dans GitHub pour la soumission des modifications apporter dans le code.

GitHub workflow

Dans l'ordre des actions *recommandées*

1. Créer une **branche**
2. Faire une modification dans le code
3. Créer une **pull request**
4. La review du code
5. Merge votre **pull request**
6. Supprimer la branche



1. Créer une branche

- Bonne pratique = ne pas travailler sur la branche **master**
- Se positionner sur la **branche develop** (development)
- A partir de la branche develop, créer une branche relative à l'issue. (Exemple ajouter une navbar)

```
git checkout develop  
git checkout -b navbar
```

2. Modifier le code

Dans la branche navbar

```
git status
git add A
git commit -m "Add search bar to navbar"
...
git add -A
git commit -m "Add logo to navbar"
```

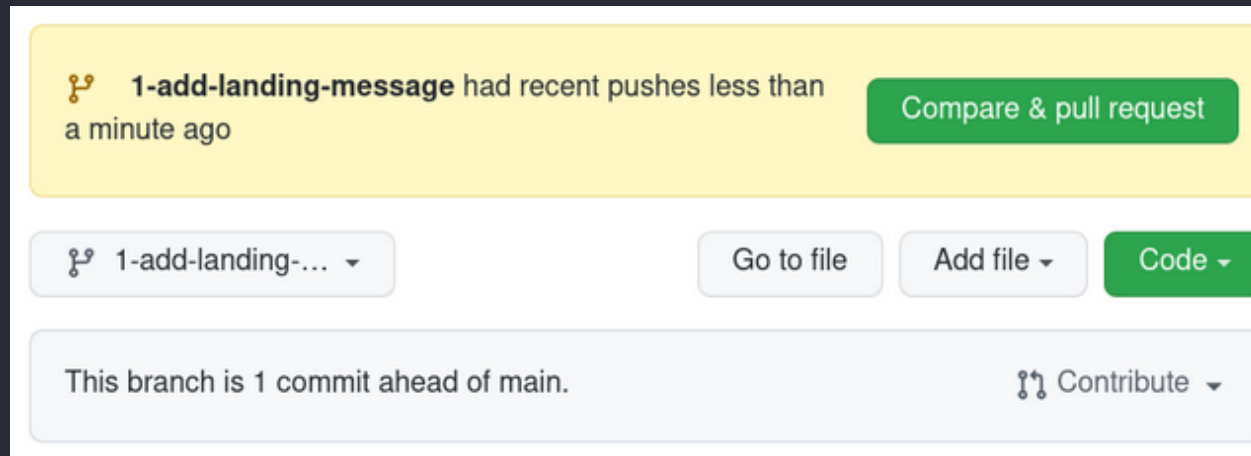
- Les commentaires des commits doivent être en anglais.

Lorsque que l'on est satisfait des modifications
apportées

```
git push
```

3. Créer une pull request

- Pour créer une pull request on se retrouve sur GitHub



- Clique sur Compare & pull request

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: develop ▼



compare: 3-form-placeholder ▼

✓ **Able to merge.** These branches can be automatically merged.

Et on compare par rapport à la **branche develop**

4. Review du code

Ensuite il y a une review

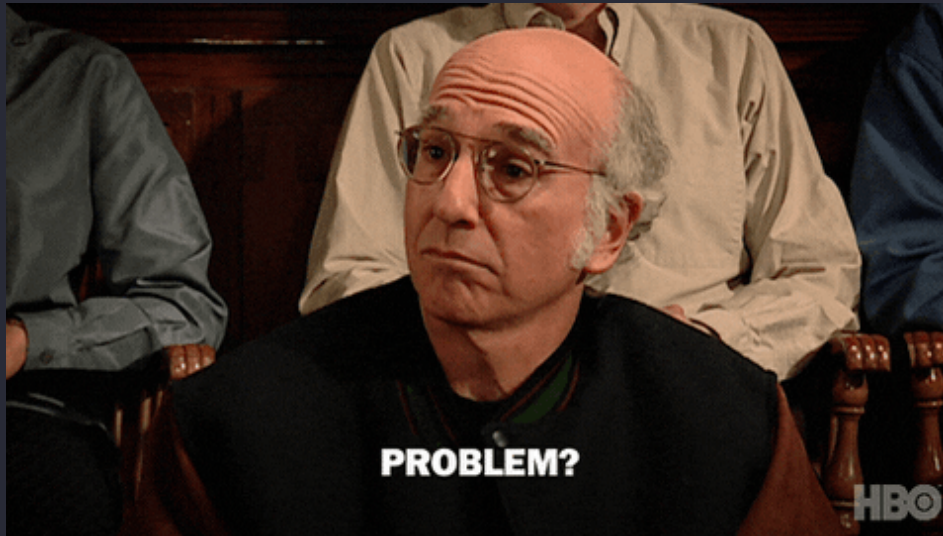
- 2 situations possibles



- Soit le code est validé = pull request mergée sur la branche develop et l'issue fermée.



- Soit il y a discussion, modification puis validation.




- Il existe un 3ème cas: **les conflits**



Des conflits surviennent souvent lorsque des coéquipiers travaillent sur différentes branches qui affectent un fichier commun.

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: develop ▼ ← compare: 3-form ▼

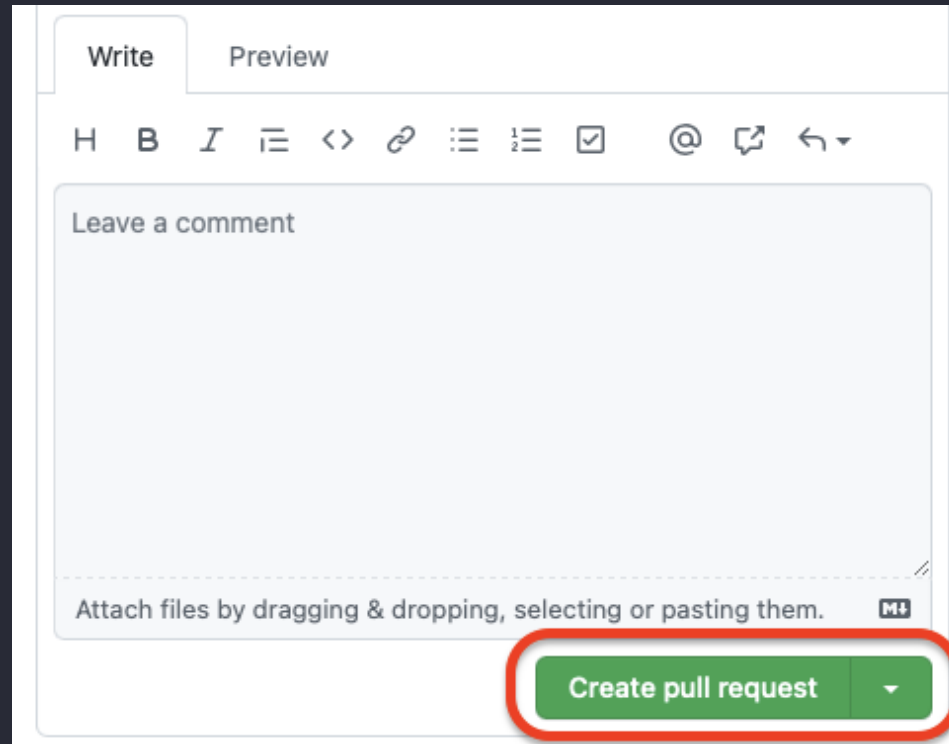
✗ **Can't automatically merge.** Don't worry, you can still create the pull request.

Sa branche ne reflète plus l'état de la branche **develop** et que la branche develop **a changé parce que quelqu'un d'autre a fusionné** (avant lui) **les changements** affectant son ou ses fichiers.

Ne pas paniquer !!

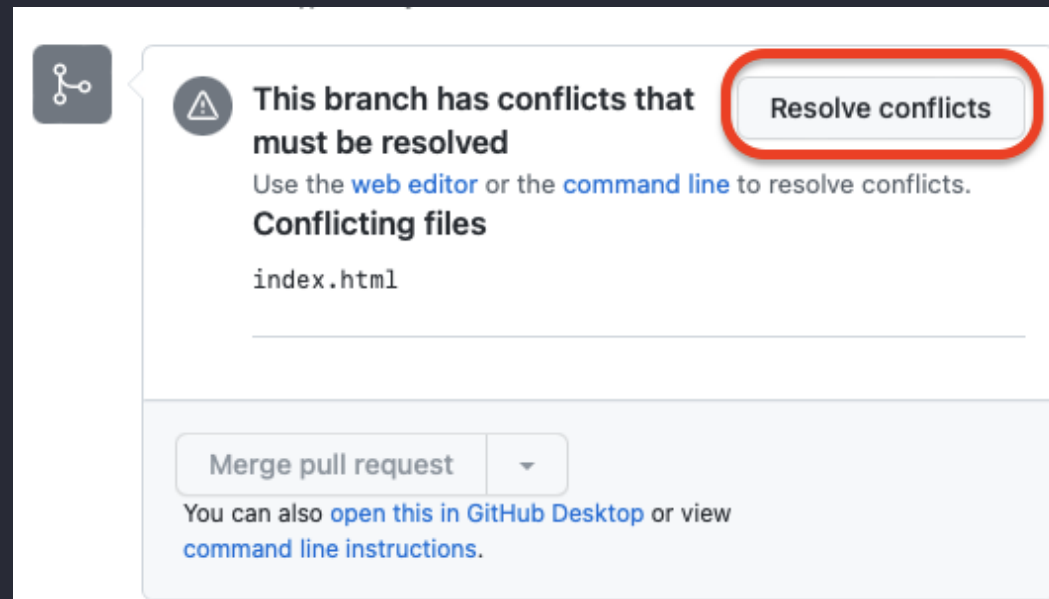


Créer quand même une **pull request**



The image shows a screenshot of a GitHub pull request creation interface. At the top, there are two tabs: 'Write' and 'Preview'. Below the tabs is a rich text editor toolbar with icons for heading (H), bold (B), italic (I), list (ul), code (<>), link, table, checklist, mention (@), share, and undo. Below the toolbar is a large text area with the placeholder text 'Leave a comment'. At the bottom of the text area, there is a dashed line and the text 'Attach files by dragging & dropping, selecting or pasting them.' with a small 'M4' icon. At the bottom right of the interface, there is a green button labeled 'Create pull request' with a dropdown arrow, which is highlighted by a red circle.

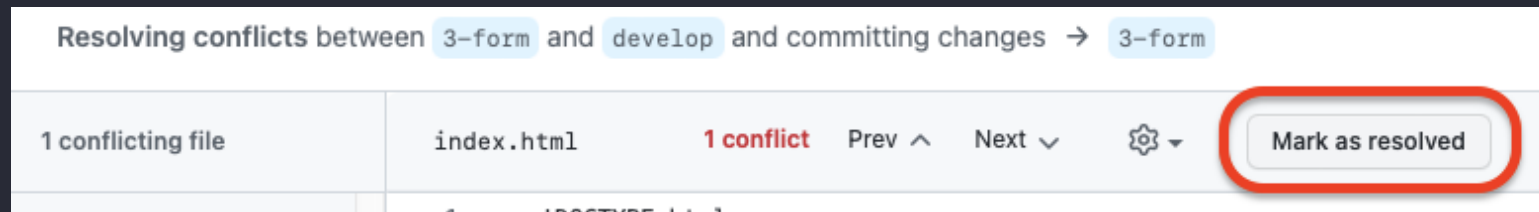
Essayer de **résoudre** le conflit



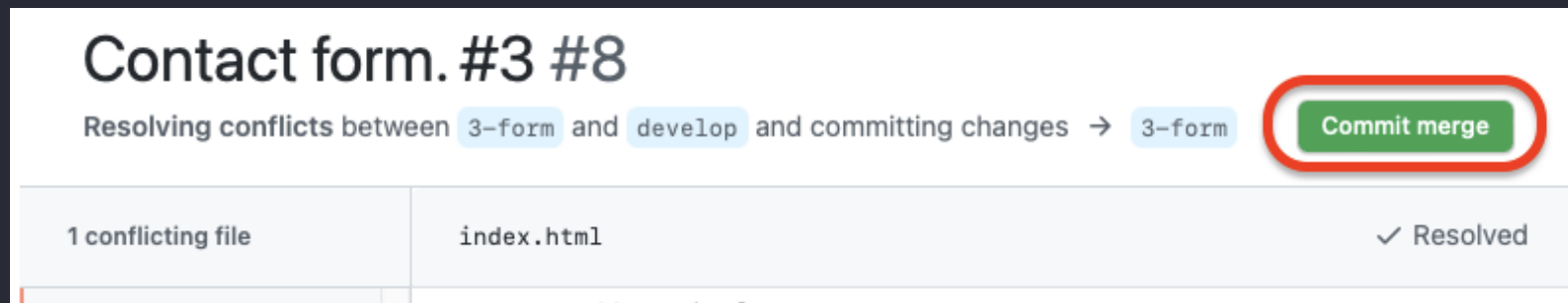
Peut être directement résolu depuis l'éditeur de
GitHub

Si résolu dans GitHub

- Appuyer sur **Mark as resolved**



- Puis cliquer sur le bouton **Commit merge**



Ensuite, si tout se passe bien la branche est libérée du conflit et la pull request peut être mergée sur la branche develop.



Une bonne pratique pour éviter les **merge conflicts**:

1. C'est revenir sur la **branche develop** et de **pull** (pour récupérer les modifications).

```
git branch  
develop  
navbar
```

```
git checkout develop  
git pull
```

- Revenir sur la branche sur laquelle vous travaillez et merger cette branche sur la branche develop (gérer les conflits s'il y en a).

```
git checkout navbar  
  
git merge develop  
# You make some changes on the files of the navbar branch
```

- Une fois les modifications faites, faire un push suivi d'une pull request sur GitHub.

```
git add-A  
git commit -m "Change conflit on navbar"  
git push
```

Conclusion :

- Créer une *nouvelle branche à partir de la branche develop* (pas master).
- *Soumettre les pull request* sur GitHub par *rapport à la branche develop* (pas par rapport à master).
- Avant de soumettre une pull request sur GitHub *une bonne pratique* et de *récupérer les modifications de la branche develop* et de régler les conflits de sa branche en interne.

Références :

- How to Use Git and GitHub in a Team like a Pro
- reveal.js

```
git branch  
navbar
```

```
git checkout develop
```

```
git pull
```

```
git checkout navbar
```

```
git merge develop
```

```
# You make some changes on the files of the navbar branch
```

```
git add -A
```