TP pour manipuler git, github et git flow

1 Création de l’arborescence

Créer une première page html pour reproduire la page du cv suivant :
https://www.ifaw.org/fr/hommes/employes/jason-bell

Pour cela, créez un un répertoire cv_jason_bell dans lequel vous écrivez le fichier html suivant :

```
<!DOCTYPE html>
<html lang="fr">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>CV</title>
</head>

<body>
  <header>

    <img src="./images/JASON-BELL.jpg" alt="">
    <h1>Jason Bell <br>Vice-président exécutif, Stratégie, programmes et opérations sur le
terrain</h1>
  </header>
  <main>
    <section>
      <p>Jason Bell a intégré IFAW en 1997 en tant que directeur du programme Éléphants. Quelques
semaines après son
        entrée en fonction, il a assisté à une réunion au Zimbabwe qui a tout changé — pour lui et pour
le monde entier.
      </p>

      <p>Au cours des 10 années précédentes, les braconniers avaient fait chuter les populations
d’éléphants de moitié.
        Dans un ultime effort pour sauver l’espèce, plusieurs nations à travers le monde ont décidé
d’interdire le
        marché de l’ivoire. En 1997, les populations de pachydermes commençaient tout doucement à
se rétablir, et les
        gouvernements, qui avaient confisqué des tonnes d’ivoire, n’ont pu résister à l’envie d’en tirer
des bénéfices
        juteux. Au terme de cette fameuse réunion, l’interdiction du commerce de l’ivoire a été
levée.</p>
```

Zoologiste et écologiste qualifié, Jason a tout de suite compris que les braconniers sauteraient sur l’occasion
pour vendre leurs marchandises dès la première journée de réouverture des marchés. Pour lui, il ne s’agissait
pas que d’une mauvaise décision politique. Il en a fait une affaire personnelle. Jason a grandi en Afrique du
Sud, et, avec sa famille, ils avaient l’habitude de se rendre aux points d’eau pour observer les troupeaux
d’éléphants venus s’y abreuver.

Il ne pouvait s’atteler seul au démantèlement du commerce de l’ivoire. Jason s’est donc mis en quête de
partenaires. Sous sa houlette, IFAW a collaboré avec plusieurs organismes afin de sécuriser les habitats
essentiels à la survie des éléphants dans le sud et l’est de l’Afrique, ainsi qu’en Inde et en Chine. Jason a
reçu l’appui et l’assentiment d’innombrables communautés locales pour son projet de relocalisation d’éléphants :
son équipe a notamment déplacé un troupeau de 83 pachydermes pour leur permettre d’échapper aux conflits
violents et incessants qui les opposaient aux villageois dans la région de Phirilongwe, au Malawi. De concert
avec des scientifiques à la renommée mondiale, Jason a convaincu le gouvernement d’Afrique du Sud de réviser son
approche de gestion des populations d’éléphants, et d’aligner les lois nationales sur les avancées
scientifiques.

Aujourd’hui, Jason est le vice-président exécutif, de la stratégie, des programmes et des opérations sur le
terrain. Fort de plus de 20 ans d’expérience en tant que directeur régional pour l’Afrique australe, puis comme
vice-président des opérations internationales et puis vice-président, conservation & sauvetage animalier, Jason
dirige les stratégies mondiales ainsi que les opérations sur le terrain dans plus de 40 pays.

Jason détient plusieurs diplômes de l’université de Pretoria.
</section>
<aside>
<h2>Contact presse :</h2>
Pour planifier une entrevue, contactez :
press@ifaw.org
</aside>
</main>
<footer>
<h2>ifaw</h2>
</footer>
</body>

</html>

1.1 Ajout d’une image

Ajoutez l’image de Jason dans le sous répertoire cv_jason_bell/images
cf https://d1jyxxz9imt9yb.cloudfront.net/person/587/detail_image/webp_regular/JASON-BELL_017.webp

1.2 Premières actions git

Initialisez votre dépôt git local (cf https://coopernet.fr/formation/git)

Ajoutez l’ensemble de l’arborescence dans la staging area puis dans le repository

1.3 Github

Créez votre repository github au nom de jason par exemple (cf https://coopernet.fr/formation/git)
« Poussez » votre code local dans votre nouveau repository sur github.

1.4 Premier push

Poussez votre repository sur github. Suivez l’aide de github :
…or push an existing repository from the command line

git remote add origin git@github.com:yvandouenel/jason.git
git branch -M main
git push -u origin main

2 Gestion des branches sans gitflow

2.1 Ajouter la branche develop

Créez la branche develop (cf https://coopernet.fr/formation/git)

2.2 Gestion de la première modification

2.2.1 Premières modifications

Ajoutez une ligne dans le head de votre fichier html pour appeler les css de boostrap 5 :
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous">

Modifier votre code de façon utiliser les classes de bootstrap :

- container (balise body)
- row (balise header)
- mb-5 (balise header)
- img-fluid (balise img)

- col-md-4 (balise img)
- col-md-8 (balise h1)

Vérifiez que l'image de Jason est bien à gauche de son nom et sa fonction

2.2.2 Commit local puis envoi sur github

Ajoutez les modifications à la staging area puis ajoutez au repository. « Poussez » sur github. Le cas échéant, utilisez git push --set-upstream origin develop pour que la branche local develop suive la branche develop sur le repository « origin » de github.

2.2.3 Merge de develop sur main

Faites un « merge » de la branche develop sur la branche main (cf https://coopernet.fr/formation/git)

2.3 Gestion d'une « Feature » sans git flow

Admettons que vous ayez envie d'essayer une fonctionnalité js qui permettra de montrer un texte qui donne l'impression que Jason écrit à la main.

Comme vous n'êtes pas sûr de vouloir garder cette fonctionnalité, vous créez une nouvelle branche :

feature_text_js

Attention à bien créer cette nouvelle branche depuis la branche develop

Vous ajoutez pour cela dans le head du html :

```
<link rel="stylesheet" href="./css/style.css">
<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapis.com/css2?family=Caveat&display=swap" rel="stylesheet">
<script src="./js/draw_text.js"></script>
```

Puis ajouter le code ci-dessous dans main > section :

```
<canvas id="canvas"></canvas>
```

Enfin vous créer le fichier /js/draw_text.js :

```
window.onload = function () {
  function draw() {
    let txt = "Je me bats pour la biodiversité.";
    var ctx = document.getElementById('canvas').getContext('2d');
```

```
    ctx.font = "1.8rem Caveat";
    let current_text = "";
    let timeout = 1;
    for (let letter of txt) {
      setTimeout(function () {
        current_text += letter;
        ctx.fillText(current_text, 0, 50);
      }, 100 * timeout);
      timeout++;
    }
  }
  draw();
}
```

Puis vous ajoutez à la « staging area » le fichier html et le répertoire js.

Vous « committez ».

Finalement, vous décidez de garder cette fonctionnalité et vous décidez donc de faire un « merge » de la branche feature_text_js vers la branche develop :

Enfin vous « poussez » le fruit de votre travail sur la branche develop distante de github.

Vous supprimez la branche feature_text_js

3 Gestion des branches avec git flow

Initialisez git flow avec la commande appropriée (cf https://coopernet.fr/formation/git)

3.1 Gestion de la première modification

3.1.1 Premières modifications

Ajoutez une ligne dans le head de votre fichier html pour appeler les css de boostrap 5 :
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous">

Modifier votre code de façon utiliser les classes de bootstrap :

- img-fluid
- container
- row
- col-md-4
- col-md-8

3.1.2 Commit local puis envoi sur github

Ajoutez les modifications à la staging area puis ajoutez au repository. « Poussez » sur github. Le cas échéant, utilisez git push --set-upstream origin develop pour que la branche local develop suive la branche develop sur le repository « origin » de github.

3.2 Gestion d'une « Feature » avec git flow

Admettons que vous ayez envie d'essayer une fonctionnalité js qui permettra de montrer un texte qui donne l'impression que Jason écrit à la main.

Vous ajoutez pour cela dans le head du html :

```
<link rel="stylesheet" href="./css/style.css">
<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapis.com/css2?family=Caveat&display=swap" rel="stylesheet">
<script src="./js/draw_text.js"></script>
```

Puis ajouter le code ci-dessous dans main > section :

```
<canvas id="canvas"></canvas>
```

Enfin vous créer le fichier /js/draw_text.js :

```
window.onload = function () {
  function draw() {
    let txt = "Je me bats pour la biodiversité.";
    var ctx = document.getElementById('canvas').getContext('2d');
    ctx.font = "1.8rem Caveat";
    let current_text = "";
    let timeout = 1;
    for (let letter of txt) {
```

```
    setTimeout(function () {
      current_text += letter;
      ctx.fillText(current_text, 0, 50);
    }, 100 * timeout);
    timeout++;
  }
 }
 draw();
}
```

Comme vous n’êtes pas sûr de vouloir garder cette fonctionnalité, vous créez une nouvelle branche :

git flow feature start text_js

Puis vous ajoutez à la « staging area » le fichier html et le répertoire js.

Vous « committez ».

Finalement, vous décidez de garder cette fonctionnalité et vous décidez donc de faire un « merge » de la branche feature/text_js vers la branche develop :

git flow feature finish text_js

Vous devriez obtenir une réponse de ce type :

```
Branches 'develop' and 'origin/develop' have diverged.
And local branch 'develop' is ahead of 'origin/develop'.
Basculement sur la branche 'develop'
Votre branche est en avance sur 'origin/develop' de 1 commit.
  (utilisez "git push" pour publier vos commits locaux)
Mise à jour c2c7cb8..ac76336
Fast-forward
 index.html      |  8 ++++++++
 js/draw_text.js | 19 ++++++++++++++++++
 2 files changed, 27 insertions(+)
 create mode 100644 js/draw_text.js
```

To github.com:yvandouenel/tpgit01.git

- [deleted] feature/text_js

Branche feature/text_js supprimée (précédemment ac76336).

Summary of actions:

- The feature branch 'feature/text_js' was merged into 'develop'

- Feature branch 'feature/text_js' has been locally deleted; it has been remotely deleted from 'origin'

- You are now on branch 'develop'

Enfin vous « poussez » le fruit de votre travail sur la branche develop distante de github.

3.3 Release avec git flow

Une fois que vous avez fusionné la branche de feature avec la branche develop, vous décidez qu'il est temps de créer une première release qui aboutira à une fusion entre la branche develop et la branche main.

A l'aide du support de cours (https://coopernet.fr/formation/git#release), réalisez cette première release.

3.4 Hotfix

Vous vous rendez compte que vous n'avez pas renseigné correctement la balise title du fichier html.

Créez un « hotfix » en suivant les instructions du support de cours :https://coopernet.fr/formation/git#hotfix

4 Utilisation de github.com

Maintenant que vous avez une arborescence avec plusieurs branches sur vos « repositories » local et distant, on peut regarder quelques fonctionnalités offertes par le site github.com

1. se rendre sur votre repository (du type https://github.com/yvandouenel/tpgit01) et voir comment on navigue à entre les branches et l'arborescence
2. se rendre sur la visualisation d'un fichier et repérer le lien « History » et voir comment on peut accéder au fichier en question lors des commits successifs.
3. Cliquer sur le bouton <> « Browse the repository at this point in the history » pour vérifier que l'on peut retourner à l'état du code correspondant au premier commit.

4. Utiliser le lien
5. utiliser la fonction de comparaison en ajoutant dans l’url /compare. Ex : https://github.com/yvandouenel/tpgit01/compare

   Regarder comment comparer le code entre la branche main et develop par exemple