🚀 Fiche d'Apprentissage Git & GitHub pour Débutants 🎯

🔷 1. Bases de Git : Les Commandes Essentielles

Commande	Explication
`git init`	Initialise un dépôt Git dans un dossier
`git status`	Vérifie l'état des fichiers suivis et non suivis
`git add .`	Ajoute tous les fichiers modifiés au suivi Git
`git commit -m "Message"`	Enregistre les modifications avec un message
`git log --oneline`	Affiche l'historique des commits de manière condensée
`git branch`	Liste toutes les branches du projet
`git checkout nom-branche`	Change de branche
`git push origin nom-branche`	Envoie une branche vers GitHub
`git pull origin nom-branche`	Récupère les dernières mises à jour de GitHub

🔷 2. Travailler avec des Branches

✅ Créer une nouvelle branche et y travailler

```
bash
CopierModifier
git checkout -b nom-branche
```

👉 Crée et bascule sur une nouvelle branche.

✅ Changer de branche

```
bash
CopierModifier
git checkout nom-branche
```

👉 Passe sur une autre branche.

✅ Fusionner une branche dans une autre

```
bash
CopierModifier
git checkout branche-cible  # Exemple : main ou dev
git merge nom-branche-a-fusionner
```

```
git push origin branche-cible
```

👉 Fusionne une branche dans une autre (ex: `feature-x` dans `dev`).

✅ Supprimer une branche une fois fusionnée

```
bash
CopierModifier
git branch -d nom-branche
git push origin --delete nom-branche
```

👉 Supprime une branche localement et sur GitHub.

🔷 3. Gestion des Modifications

Situation	Commande
Voir l'état des fichiers	`git status`
Ajouter un fichier spécifique	`git add fichier.swift`
Ajouter tous les fichiers	`git add .`
Annuler un `git add`	`git restore --staged fichier.swift`
Enregistrer les modifications	`git commit -m "Message"`
Annuler un commit avant `push`	`git reset --soft HEAD~1`

🔷 4. Envoyer et Récupérer du Code depuis GitHub

Action	Commande
Envoyer une branche sur GitHub	`git push origin nom-branche`
Récupérer les dernières mises à jour	`git pull origin nom-branche`
Cloner un dépôt GitHub	`git clone URL_du_depot`

🔷 5. Cas Pratiques (Exemples de Workflow)

💡 Créer une nouvelle branche pour une fonctionnalité et la fusionner

1️⃣ Créer une nouvelle branche pour une mise à jour UI :

```
bash
CopierModifier
```

```
git checkout -b feature-ui-update
```

2️⃣ Faire des modifications, puis les ajouter et commiter :

```
bash
CopierModifier
git add .
git commit -m "Mise à jour des couleurs et de la mise en page"
```

3️⃣ Envoyer cette branche sur GitHub :

```
bash
CopierModifier
git push origin feature-ui-update
```

4️⃣ Une fois validée, fusionner avec `dev` :

```
bash
CopierModifier
git checkout dev
git merge feature-ui-update
git push origin dev
```

5️⃣ Supprimer la branche après la fusion :

```
bash
CopierModifier
git branch -d feature-ui-update
git push origin --delete feature-ui-update
```

🎯📝 Points Clés à Retenir

✅ **Git permet de suivre l'évolution du code et de revenir en arrière si besoin.**

✅ **Les branches permettent de tester sans impacter le code principal.**

✅ **Toujours faire `git pull origin branche` avant de travailler pour éviter les conflits.**

✅ **Faire des commits clairs et bien nommés.**

✅ **Toujours fusionner dans `dev` avant `main` pour garder une version stable.**

🚀 **Tu es prêt pour ton évaluation ! Bonne chance !** 💪💡🔥