

# Hello World!

Quentin Fortier

February 20, 2022

**Git** est un logiciel qui permet de versionner du code informatique, c'est à dire de conserver un historique de toutes les modifications.  
Je l'utilise mais vous n'allez pas avoir besoin de l'utiliser.

**Git** est un logiciel qui permet de versionner du code informatique, c'est à dire de conserver un historique de toutes les modifications.  
Je l'utilise mais vous n'allez pas avoir besoin de l'utiliser.

**GitHub** est un serveur permettant d'héberger des dépôts (projets versionnés par Git). Je vais mettre tout mon cours sur GitHub à l'adresse <https://github.com/mp2i-fsm/mp2i-2021>

Nous allons utiliser 3 langages de programmation :

- **OCaml** : langage fonctionnel développé par l'INRIA à partir de 1996.  
Utilisé pour concevoir des programmes sûrs, par exemple par Facebook (ReasonML) et Microsoft (F#).

Nous allons utiliser 3 langages de programmation :

- **OCaml** : langage fonctionnel développé par l'INRIA à partir de 1996.  
Utilisé pour concevoir des programmes sûrs, par exemple par Facebook (ReasonML) et Microsoft (F#).
- **C** : langage bas niveau (proche du langage machine) impératif développé à partir de 1972.  
Utilisé surtout pour la programmation système (Linux...) et parallèle (CUDA...).

Nous allons utiliser 3 langages de programmation :

- **OCaml** : langage fonctionnel développé par l'INRIA à partir de 1996.  
Utilisé pour concevoir des programmes sûrs, par exemple par Facebook (ReasonML) et Microsoft (F#).
- **C** : langage bas niveau (proche du langage machine) impératif développé à partir de 1972.  
Utilisé surtout pour la programmation système (Linux...) et parallèle (CUDA...).
- **SQL** : langage de requêtes pour les bases de données.  
Utilisé par la plupart des entreprises pour stocker des données (utilisateurs d'un site web par exemple).

Voici le même algorithme (ajout dans un arbre de recherche) en **Python**...

---

```
def add(self, val):  
    if val < self.val:  
        if not self.left:  
            self.left = BST(val)  
        else:  
            self.left.add(val)  
    elif not self.right:  
        self.right = BST(val)  
    else:  
        self.right.add(val)
```

---

... et en **OCaml** :

---

```
let rec add abr e = match abr with
  | V -> N(e, V, V)
  | N(r, g, d) -> if e < r then N(r, add g e, d)
                  else N(r, g, add d e);;
```

---



Tous les cours et TDs seront sous forme de **notebooks Jupyter**, qui permettent de mélanger du code et du texte/image...

Vous pouvez suivre le cours de façon interactive en ammenant votre PC chargé.

# Jupyter avec Binder

**Binder** est un serveur Jupyter sur lequel j'ai mis OCaml, C, Python.  
Vous n'aurez donc rien à installer !

# Jupyter avec Binder

**Binder** est un serveur Jupyter sur lequel j'ai mis OCaml, C, Python.  
Vous n'aurez donc rien à installer !

Cliquer  sur

<https://github.com/mp2i-fsm/mp2i-2021>

# Jupyter avec Binder

**Binder** est un serveur Jupyter sur lequel j'ai mis OCaml, C, Python.  
Vous n'aurez donc rien à installer !

Cliquer  sur

<https://github.com/mp2i-fsm/mp2i-2021>

Prise en main de Jupyter

- Prenez de l'**intérêt** dans ce que vous faites pour que travailler devienne un plaisir et pas une contrainte.

# Conseils en informatique

- Prenez de l'**intérêt** dans ce que vous faites pour que travailler devienne un plaisir et pas une contrainte.
- Réfléchissez avant d'écrire du code compliqué. Très souvent le code qu'on vous demande est **simple** (< 10 lignes) : ne pas foncer tête baissée et perdre trop de temps sur une question simple.

- Prenez de l'**intérêt** dans ce que vous faites pour que travailler devienne un plaisir et pas une contrainte.
- Réfléchissez avant d'écrire du code compliqué. Très souvent le code qu'on vous demande est **simple** (< 10 lignes) : ne pas foncer tête baissée et perdre trop de temps sur une question simple.
- Connaissez les **méthodes de base** et sachez les réutiliser sur un exercice différent.

- Prenez de l'**intérêt** dans ce que vous faites pour que travailler devienne un plaisir et pas une contrainte.
- Réfléchissez avant d'écrire du code compliqué. Très souvent le code qu'on vous demande est **simple** (< 10 lignes) : ne pas foncer tête baissée et perdre trop de temps sur une question simple.
- Connaissez les **méthodes de base** et sachez les réutiliser sur un exercice différent.
- Pensez à **réutiliser** les questions/fonctions précédentes dans une question de DS.



- Prenez de l'**intérêt** dans ce que vous faites pour que travailler devienne un plaisir et pas une contrainte.
- Réfléchissez avant d'écrire du code compliqué. Très souvent le code qu'on vous demande est **simple** ( $< 10$  lignes) : ne pas foncer tête baissée et perdre trop de temps sur une question simple.
- Connaissez les **méthodes de base** et sachez les réutiliser sur un exercice différent.
- Pensez à **réutiliser** les questions/fonctions précédentes dans une question de DS.
- Essayez d'écrire du code **concis, propre et clair**. Ajouter un petit commentaire quand vous définissez une fonction/variable auxiliaire.

## Exemple de méthode à connaître

### Question

Soit  $\mathcal{P}$  une propriété et  $L$  une liste.

Comment savoir si  $\mathcal{P}$  est vrai pour tout élément de  $L$  ? ( $\mathcal{P}(e), \forall e \in L$  ?)

# Exemple de méthode à connaître

## Question

Soit  $\mathcal{P}$  une propriété et  $L$  une liste.

Comment savoir si  $\mathcal{P}$  est vrai pour tout élément de  $L$  ? ( $\mathcal{P}(e), \forall e \in L$  ?)

On regarde s'il existe un élément vérifiant la négation de  $\mathcal{P}$ .

---

```
def f(L):  
    for e in L:  
        if not P(e):  
            return False  
    return True
```

---

Exemples :

- Est-ce que tous les éléments de  $L$  sont positifs ?
- Est-ce que un élément  $e$  appartient à  $L$  ?

Si vous avez une erreur dans votre code :

- 1 Commencez par essayer de comprendre vous-même avec l'origine de l'erreur.

Si vous avez une erreur dans votre code :

- ❶ Commencez par essayer de comprendre vous-même avec l'origine de l'erreur.
- ❷ Chercher sur Google votre erreur.

Si vous avez une erreur dans votre code :

- ❶ Commencez par essayer de comprendre vous-même avec l'origine de l'erreur.
- ❷ Chercher sur Google votre erreur.
- ❸ Seulement si vous ne comprenez toujours pas, appelez-moi.