Slides Test

CMAP Ecole Polytechnique

Samy Mekkaoui

In the morning

Contexte

- **Qui sommes-nous ?**

Contexte

- **Qui sommes-nous ?**
  - des **data scientists** de l'Insee

Contexte

- **Qui sommes-nous ?**
  - des **data scientists** de l'Insee
  - frustrés par l'**approche** souvent purement **technique** de la data science

Contexte

- **Qui sommes-nous ?**
  - des **data scientists** de l'Insee
  - frustrés par l'**approche** souvent purement **technique** de la data science
  - convaincus que les **bonnes pratiques de développement** valent à être enseignées

Qu'est ce qu'un data scientist ?

- Tendance à la **spécialisation** : *data analyst*, *data engineer*, *ML Engineer*...

Qu'est ce qu'un data scientist ?

- Tendance à la **spécialisation** : *data analyst*, *data engineer*, *ML Engineer*...
- Rôle d'**interface** entre métier et équipes techniques

Qu'est ce qu'un data scientist ?

- Tendance à la **spécialisation** : *data analyst*, *data engineer*, *ML Engineer*...
- Rôle d'**interface** entre métier et équipes techniques
  - **Compétences mixtes** : savoir métier, modélisation, IT

Getting up

- Let's put some maths here to check if it works :

$$f(x) = \int_0^T g(s, x) ds$$

La notion de mise en production

- **Mettre en production** : faire **vivre** une application dans l'espace de ses **utilisateurs**

La notion de mise en production

- **Mettre en production** : faire **vivre** une application dans l'espace de ses **utilisateurs**
  - Notion simple mais mise en oeuvre compliquée !

La notion de mise en production

- **Mettre en production** : faire **vivre** une application dans l'espace de ses **utilisateurs**
  - Notion simple mais mise en oeuvre compliquée !
- **Dépasser le stade de l'expérimentation**

La notion de mise en production

- **Mettre en production** : faire **vivre** une application dans l'espace de ses **utilisateurs**
  - Notion simple mais mise en oeuvre compliquée !
- **Dépasser le stade de l'expérimentation**
  - Comprendre les **besoins** des utilisateurs

La notion de mise en production

- **Mettre en production** : faire **vivre** une application dans l'espace de ses **utilisateurs**
  - Notion simple mais mise en oeuvre compliquée !
- **Dépasser le stade de l'expérimentation**
  - Comprendre les **besoins** des utilisateurs
  - **Bonnes pratiques** de développement

La notion de mise en production

- **Mettre en production** : faire **vivre** une application dans l'espace de ses **utilisateurs**
  - Notion simple mais mise en oeuvre compliquée !
- **Dépasser le stade de l'expérimentation**
  - Comprendre les **besoins** des utilisateurs
  - **Bonnes pratiques** de développement
  - Techniques informatiques d'**industrialisation**

Contenu du cours

- **Pré-requis**

Contenu du cours

- **Pré-requis**
  - Introduction au terminal `Linux`

Contenu du cours

- **Pré-requis**
  - Introduction au terminal `Linux`
  - **Contrôle de version** avec `Git`

Contenu du cours

- **Pré-requis**
  - Introduction au terminal `Linux`
  - **Contrôle de version** avec `Git`
- **Bonnes pratiques** de développement

Contenu du cours

- **Pré-requis**
  - Introduction au terminal `Linux`
  - **Contrôle de version** avec `Git`
- **Bonnes pratiques** de développement
  - **Travail collaboratif** avec `Git`

Contenu du cours

- **Pré-requis**
  - Introduction au terminal `Linux`
  - **Contrôle de version** avec `Git`
- **Bonnes pratiques** de développement
  - **Travail collaboratif** avec `Git`
  - **Qualité** du code

Contenu du cours

- **Pré-requis**
  - Introduction au terminal `Linux`
  - **Contrôle de version** avec `Git`
- **Bonnes pratiques** de développement
  - **Travail collaboratif** avec `Git`
  - **Qualité** du code
  - **Structure** des projets

Contenu du cours

- **Pré-requis**
  - Introduction au terminal `Linux`
  - **Contrôle de version** avec `Git`
- **Bonnes pratiques** de développement
  - **Travail collaboratif** avec `Git`
  - **Qualité** du code
  - **Structure** des projets
  - Traitement des **données volumineuses**

Contenu du cours

- **Pré-requis**
  - Introduction au terminal `Linux`
  - **Contrôle de version** avec `Git`
- **Bonnes pratiques** de développement
  - **Travail collaboratif** avec `Git`
  - **Qualité** du code
  - **Structure** des projets
  - Traitement des **données volumineuses**
  - Favoriser la **portabilité** d'une application

Contenu du cours

- **Pré-requis**
  - Introduction au terminal `Linux`
  - **Contrôle de version** avec `Git`
- **Bonnes pratiques** de développement
  - **Travail collaboratif** avec `Git`
  - **Qualité** du code
  - **Structure** des projets
  - Traitement des **données volumineuses**
  - Favoriser la **portabilité** d'une application
- **Mise en production**

Contenu du cours

- **Pré-requis**
  - Introduction au terminal `Linux`
  - **Contrôle de version** avec `Git`
- **Bonnes pratiques** de développement
  - **Travail collaboratif** avec `Git`
  - **Qualité** du code
  - **Structure** des projets
  - Traitement des **données volumineuses**
  - Favoriser la **portabilité** d'une application
- **Mise en production**
  - **Déploiement**

Contenu du cours

- **Pré-requis**
  - Introduction au terminal `Linux`
  - **Contrôle de version** avec `Git`
- **Bonnes pratiques** de développement
  - **Travail collaboratif** avec `Git`
  - **Qualité** du code
  - **Structure** des projets
  - Traitement des **données volumineuses**
  - Favoriser la **portabilité** d'une application
- **Mise en production**
  - **Déploiement**
  - **MLOps**

Partie 1 : bonnes pratiques de développement

Plan de la partie

1. **Travail collaboratif** avec `Git`
2. **Qualité** du code
3. **Structure** des projets
4. Traitement des **données volumineuses**
5. Favoriser la **portabilité** d'une application

1 Le travail collaboratif avec `Git`

Pourquoi utiliser Git ?

Concepts essentiels

Bonnes pratiques

Que versionne-t-on ?

- Essentiellement du **code source**

Bonnes pratiques

Que versionne-t-on ?

- Essentiellement du **code source**
- **Pas d'outputs** (fichiers `.html`, `.pdf`, modèles...)

Bonnes pratiques

Que versionne-t-on ?

- Essentiellement du **code source**
- **Pas d'outputs** (fichiers `.html`, `.pdf`, modèles...)
- **Pas de données**, d'informations locales ou sensibles

Bonnes pratiques

Que versionne-t-on ?

- Essentiellement du **code source**
- **Pas d'outputs** (fichiers `.html`, `.pdf`, modèles...)
- **Pas de données**, d'informations locales ou sensibles

Bonnes pratiques

Que versionne-t-on ?

- Essentiellement du **code source**
- **Pas d'outputs** (fichiers `.html`, `.pdf`, modèles...)
- **Pas de données**, d'informations locales ou sensibles

> **Note**
>
> Pour définir des règles qui évitent de committer tel ou tel fichier, on utilise un fichier nommé **`.gitignore`**.
> Si on mélange du code et des éléments annexes (*output*, données...) dans un même dossier, il **faut consacrer du temps à ce fichier**.
> Le site `gitignore.io` peut vous fournir des modèles.
> N'hésitez pas à y ajouter des règles conservatrices (par exemple `*.csv`), comme cela est expliqué dans la documentation `utilitR`.

Bonnes pratiques

Breakfast

- Eat eggs

Breakfast

- Eat eggs
- Drink coffee

In the evening

Dinner

- Eat spaghetti

Dinner

- Eat spaghetti
- Drink wine

Going to sleep

- Get in bed

Going to sleep

- Get in bed
- Count sheep