PoC 2023-2024

Preuve de concept

Présentation du module

bruno.denis@ens-paris-saclay.fr
Bruno Denis

kevin.godineau@ens-paris-saclay.fr
Kevin Godineau

école normale supérieure paris—saclay

Intervenants

Bruno Denis

- Maitre de Conférences, ENS Paris-Saclay
- Thématique de recherche : Systèmes à évènements discrets et systèmes dynamiques hybrides

Kevin Godineau *(Responsable du module)*

- Maitre de Conférences, ENS Paris-Saclay
- Thématique de recherche : Pilotage des machines de fabrication additive LPBF

Sylvain Lavernhe

- Professeur des Universités, Université Paris-Saclay
- Thématique de recherche : Amélioration des procédés additifs et soustractifs

Objectifs

1. Améliorer vos compétences en programmation (Python et Matlab)
   - Reproduire des travaux décrits dans la littérature scientifique

2. Rendre compte de vos développements scientifiques
   - Analyser de manière pertinente les développements effectués :
     - Au travers de cas d'études judicieusement choisi
     - Au travers de critères (complexité algorithmique, complexité en temps …)
   - Fournir un travail reproductible

3. Maitriser des outils de transmission
   - Maitriser les outils adaptés (à la présentation, à la diffusion, à la reproductibilité) des développements informatiques effectués

Il ne **s'agit pas de Génie Logiciel** mais plutôt de **bonnes pratiques scientifiques** associées au développement logiciel

Déroulement du module (1/2)

Première partie – mise en place des outils, des méthodes

- Séance 1 : 2h - BD et KG
  - Présentation du module, des objectifs, des outils et des ressources
- Séance 2 : 2h - KG
  - Utilisation de Git, prise en main d'un outil de gestion de version collaboratif
- Séance 3 : 4h (2h + 2h) - BD et KG
  - Exercices d'applications (Python)
  - Exercices d'applications (Matlab)
- Séance 4 : 4h - BD
  - Programmation Orientée Objet, import, traitement et affichage (Python)
- Séance 5 : 4h - SL
  - Manipulation de tenseurs (matrice n-dimensions), calcul symbolique (Matlab)
- Séance 6 : 4h (2h + 2h) - BD et KG
  - Évaluation sur la maitrise des outils – CSC sur Python, IN2P sur Matlab
  - Présentation des sujets et précisions sur le travail attendu et les modalités de l'évaluation du projet

Déroulement du module (2/2)

Deuxième partie – réalisation d'une preuve de concept

- Séance 7 : Suivi de projet
  - Vous nous présentez votre travail de manière informelle, vos difficultés, votre avancement
  - Nous sommes disponibles pour répondre à vos questions
- Séance 8 : Suivi de projet
  - idem
- Séance 9 : Évaluation
  - Rendu sous GitHub de votre projet
  - Code commenter et fonctionnel
  - Rapport écrit en Markdown qui explicite vos travaux
  - Soutenance orale

L'évaluation

Maitrise d'outils --10 %

- Évaluation de la séance 6

Suivi de projet -- 40 %

- Avancement du projet entre fin octobre et fin décembre
- Développement cohérent et en parallèle du code, des commentaires, des tests et du rapport explicatif.

Rendu écrit – 30 %

- Clarté du code déposé
- Pertinence du rapport

Présentation orale – 20 %

- Soutenance
- Réponse aux questions

Plan

1. Introduction
2. Qu'est-ce que l'on entend par « preuve de concept »
3. Compétences attendues à l'issue du module
4. Présentation des outils utilisés
5. Présentation des ressources

Plan

1. Introduction
2. Qu'est-ce que l'on entend par « preuve de concept »
3. Compétences attendues à l'issue du module
4. Présentation des outils utilisés
5. Présentation des ressources

Description générale

« La preuve (de concept) semble être une preuve de possibilité dont la réalisation est démontrée dans une pratique expérimentale. Elle prouve que le lien de causalité hypothétique, la structure proposée, la fonction suggérée ou l'approche méthodologique adoptée dans la recherche est vérifiée dans au moins un cas réel - le cas test » [Kendig, 2016]

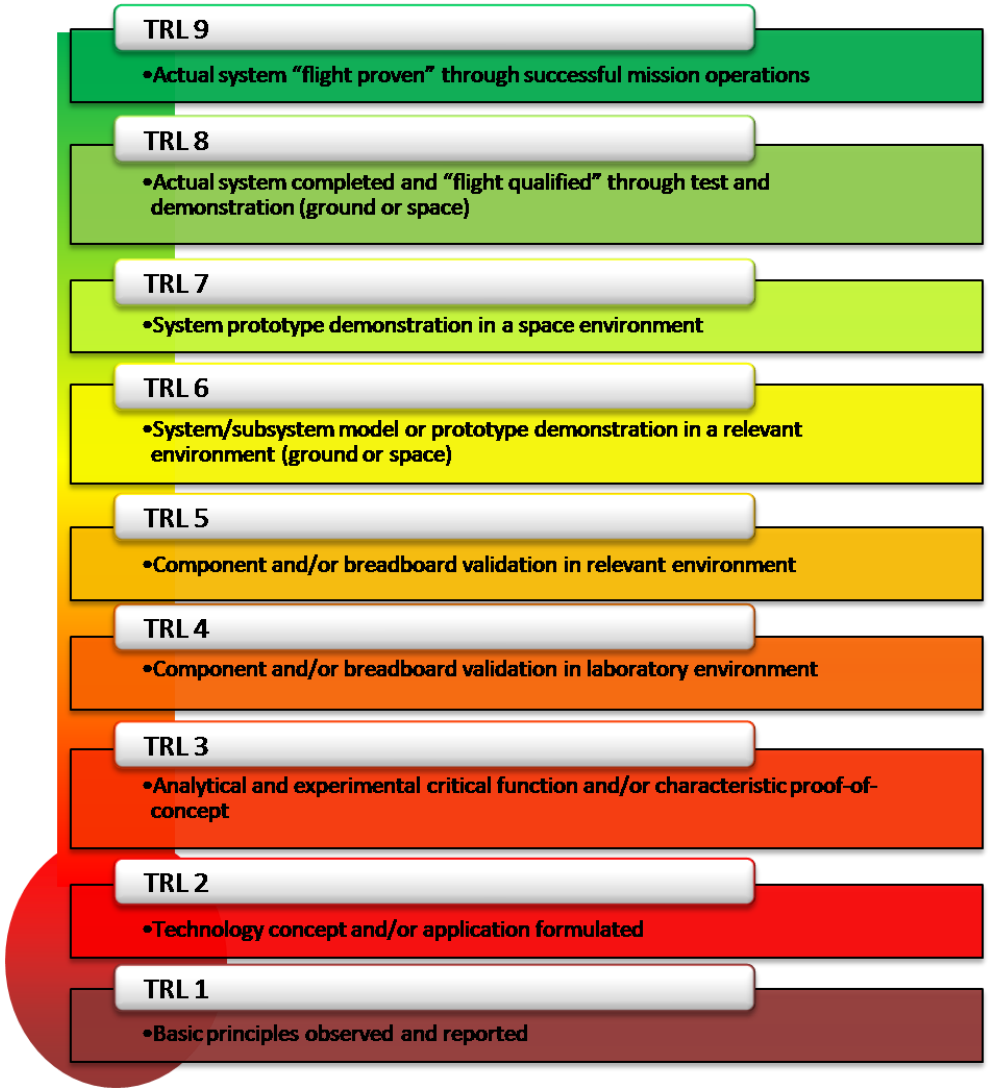
[Kendig, 2016] Kendig, Catherine Elizabeth, 2016. What is Proof of Concept Research and how does it Generate Epistemic and Ethical Categories for Future Scientific Practice? *Science and Engineering Ethics*. June 2016. Vol. 22, no. 3, p. 735–753. doi : 10.1007/s11948-015-9654-0

Appliquée a nos domaines

En science de l'ingénieur, avec un regard industriel on parle de : notion de TRL (Technology Readiness Level) ou Niveau de maturité technologique

- La preuve de concept correspond ici au TRL 3
- Notion centrée sur le développement d'un « produit, système, procédé »

À l'échelle d'un laboratoire de recherche on ne dépasse que rarement le TRL 4



Pourquoi enseigner cela

Aujourd'hui, tous les scientifiques doivent développer des modèles logiciels pour valider les résultats de leurs recherches (Proof of Concept). Il s'agit de :

- Modéliser et coder un algorithme issu de vos travaux de recherche et déterminer sa classe de complexité temporelle et spatiale.
- Collecter, générer, stocker et formater des ensembles de données à traiter par l'algorithme
- Faire traiter les ensembles de données par votre algorithme
- Évaluer la qualité des résultats produits par l'algorithme à l'aide d'une évaluation clairement définie.

C'est dans ce contexte informatique que le module d'enseignement est proposé.

- Pour mettre en pratique ces techniques et outils, chaque étudiant travaillera sur une étude de cas. L'étude de cas sera un projet de recherche déjà publié, dont les résultats devront être reproduits, documentés et éventuellement étendus.

- Reproduire un travail existant, c'est à la fois le valider et en comprendre le contenu.

Plan

1. Introduction
2. Qu'est-ce que l'on entend par « preuve de concept »
3. Compétences attendues à l'issue du module
4. Présentation des outils utilisés
5. Présentation des ressources

Les compétences attendues

À l'issue de la séance 6 (fin octobre)

- Maitrise d'outil et d'un langage de programmation
- On évaluera spécifiquement :
  - Import et export de données sous format .csv
  - Traitement de données (tri, calcul matriciel …)
  - Représentation des résultats 2d et 3d
  - Rédaction d'un rapport explicatif du développement effectué

À l'issue du module (mi-décembre)

- Implémenter, valider et tester un algorithme issu d'un article scientifique (base)
- Mettre en place des jeux de données pertinents
- Valider tester l'algorithme sur ces jeux de données
- Evaluer les performances au travers de critères
- Représenter les résultats
- Analyser, expliciter et retranscrire cela dans un rapport explicitant le code, les développements

Exemple simple de ce que l'on attend

Présentation de deux exemples disponible sur GitHub

(lien 1)

- Exemple sur un cas d'application simple scientifiquement.
- Travail fait par Bruno Denis à titre d'exemple
- Sujet : intersection entre un plan et un maillage au format .stl

(lien 2)

- Exemple d'un étudiant d'il y a 2 ans
- Sujet : Mise en place d'un filtrage morphologique pour une analyse de rugosité de surface, comparaison avec les traitements effectués sur le logiciel propriétaire et la norme.

Deux langages de programmation pourquoi !

Le master 2 AMSS possède 2 parcours IN2P et CSC. Dans chacun des domaines scientifiques, historiquement des langages différents ont été utilisés :

- Matlab (IN2P)
  - Manipulation de matrice
  - Représentation et affichage
  - Intégration avec Simulink (commande des systèmes continue)
- Python (CSC)
  - Programmation orientée objet
  - Un langage de programmation open-source
  - Une communauté de développeur conséquente

L'évaluation de la maitrise d'outil se fera donc sur Matlab pour les IN2P et Python pour les CSC avec des compétences techniques similaires.

Plan

1. Introduction
2. Qu'est-ce que l'on entend par « preuve de concept »
3. Compétences attendues à l'issue du module
4. Présentation des outils utilisés
5. Présentation des ressources

Présentation générale

Langages informatiques

- Programmation
  - Matlab
  - Python

- Documentation
  - Markdown
  - Latex math

Système de gestion de version

- Git

Présentation générale

Langages informatiques

- Programmation
  - Matlab
  - Python

- Documentation
  - Markdown
  - Latex math

Logiciel informatiques

- Environnement de développement
  - Matlab
  - Anaconda

Système de gestion de version

- Git

Présentation générale

Langages informatiques

- Programmation
  - Matlab
  - Python

- Documentation
  - Markdown
  - Latex math

Logiciel informatiques

- Environnement de développement
  - Matlab
  - Anaconda

- Editeurs
  - Marktext
  - Or online editor

Système de gestion de version

- Git

Présentation générale

Langages informatiques

- Programmation
  - Matlab
  - Python

- Documentation
  - Markdown
  - Latex math

Logiciel informatiques

- Environnement de développement
  - Matlab
  - Anaconda

- Editeurs
  - Marktext
  - Or online editor

Système de gestion de version

- Git

Outils de gestion de versions

- Git (client mode texte)
- Fork (client graphique)
- Github.com (hebergement)

Git (1/2)

Description

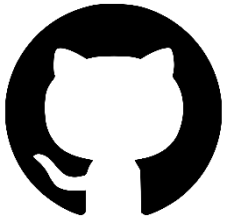
- Logiciel de gestion de versions collaboratif.

Interface graphique Fork :

- Une interface graphique pour utiliser Git à la souris. Une parmi beaucoup d'autre
- Avantage : compatibilité Mac et Windows + accès gratuit aux répertoires privés sur GitHub

Partage et documentation des données via GitHub

- Plateforme web d'hébergement et de gestion de développement de logiciels, utilisant le logiciel de gestion de versions Git.
- Plus importante plateforme hébergeur de code informatique
- On aurait souhaité utiliser le Gitlab Université Paris-Saclay mais vous n'y avez pas encore tous accès via votre adresse mail ENS.



Git (2/2)

Pourquoi utiliser Git

- Pour éviter les V1, V2, Vfinale, Vfinalefinale
- Car vous sauvegarder une arborescence et pas uniquement la trace à l'instant t
- Outil utilisé internationalement dans le développement (logiciel, jeux vidéo …)

Présentation d'un exemple

Pour aller plus loin : GitFlow (lien)

- Méthodologie permettant de structurer des projets proprement

Eco-system Python (1/4)

Programming language

Python is a high-level, general-purpose programming language (v3.x)

Language implementation

CPython the reference implementation written in C and Python

IPython, which stands for Interactive Python (notebook interface, interactive data…)

Libraries

Standard modules such as math, random, statistics, sys, …

Third-party modules such as NumPy, mathplotlib, SciPy, Graphviz, TensoFlow, …

Package manager Pip, Conda

Environment management system venv, Conda

Notebook Jupyter

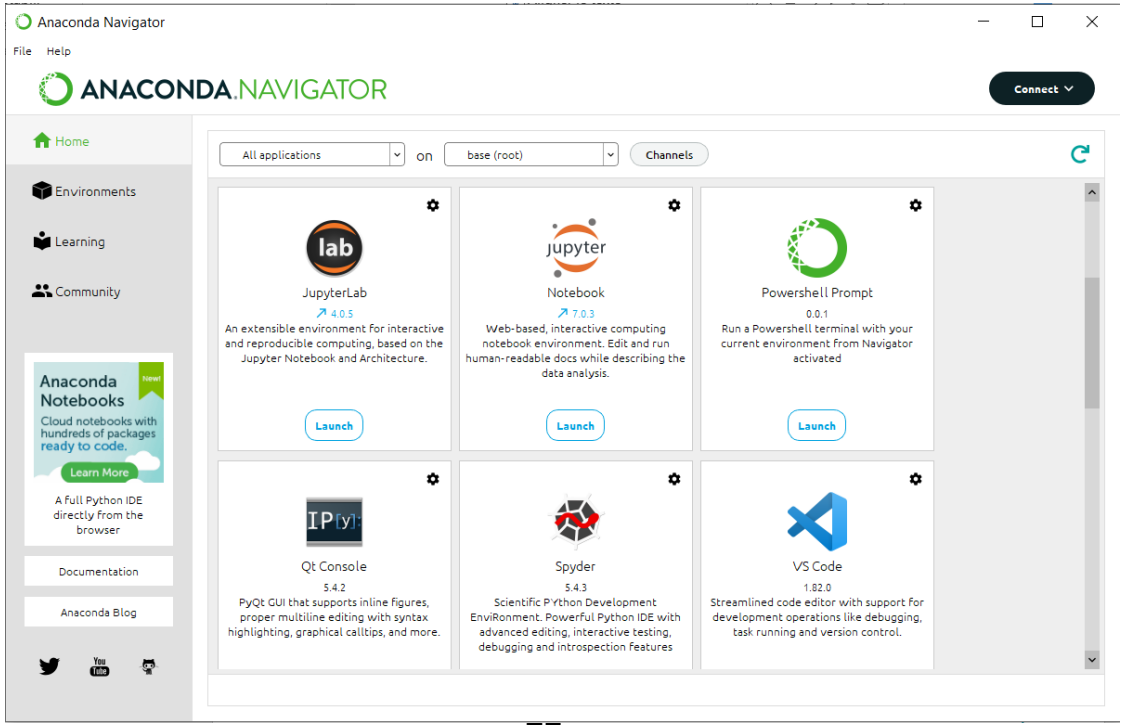
Integrated Development Environment (IDE) Sypder, PyCharm, VsCode, …

Eco-system Python (2/4)

Anaconda

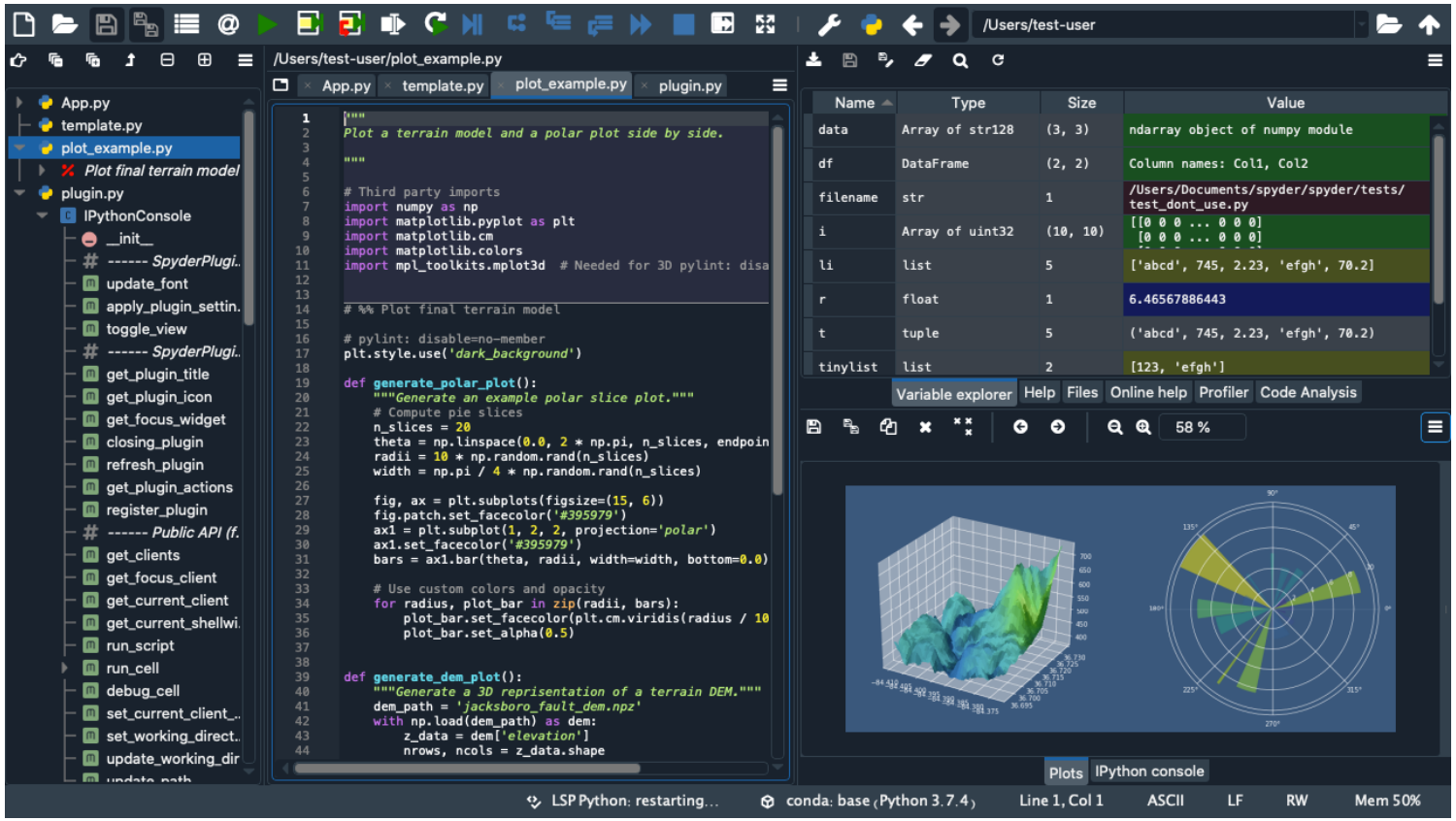
Anaconda is a distribution of the Python including : CPython, Ipython, Standard modules, many modules for scientific computing,IDE (Sypder, Vscode, Pycharm), Conda, Jupyter Notebook

- Download from depuis https://www.anaconda.com/products/distribution
- Choose Python 3.X version



Eco-system Python (3/4)

IDE example: Spyder



Eco-system Python (4/4)

Installation guide

- Download Anaconda distribution from https://www.anaconda.com/products/distribution

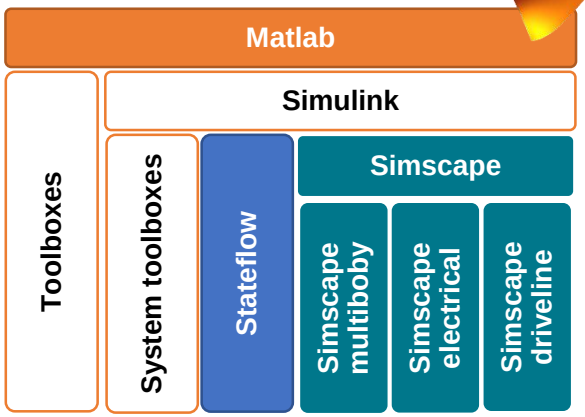
- Select « Python 3.X » version at the installation beginning

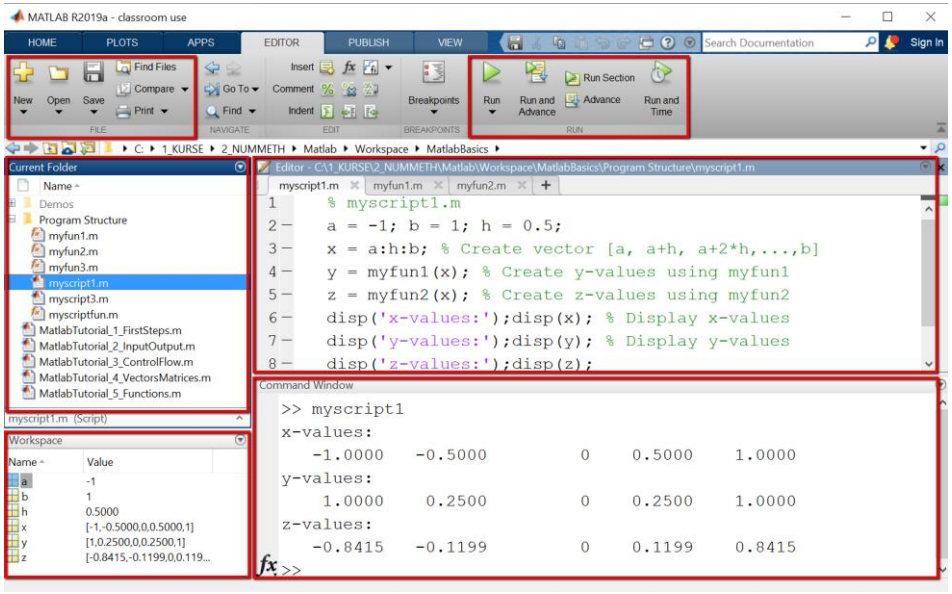
Matlab

Présentation rapide de l'environnement

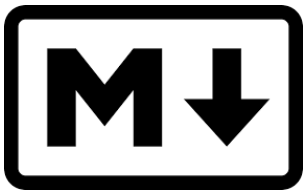
- Matlab est à la fois un langage de programmation et un environnement intégré de développement (IDE)

- L'environnement de Matlab intègre
  - Un **éditeur** de code Matlab
  - Un **interpréteur** de code pour exécuter
  - Un **éditeur graphique** Simulink
  - De nombreuses bibliothèques appelées « **toolbox** »





Markdown language (1/8)



What is Markdown?

Markdown is a lightweight markup language that you can use to add formatting elements to plaintext text documents. Created by John Gruber in 2004, Markdown is now one of the world's most popular markup languages.

Depending on the application you use, you may not be able to preview the formatted document in real time. But that's okay. According to Gruber, Markdown syntax is designed to be readable and unobtrusive, so the text in Markdown files can be read even if it isn't rendered.

For example, to denote a heading, you add a number sign before it

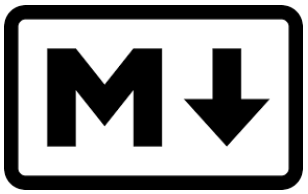
```
# Heading One
```

Or to make a phrase bold, you add two asterisks before and after it

```
**this text is bold**.
```

(extract from www.markdownguide.org)

Markdown language (2/8)



Why use it ?

Markdown is use in GitHub.com and GitLab.com and Paris-Saclay GitLab websites and in Jupyter, Google Colab, Microsoft Azure, Apache Zeppelin and Deepnote notebooks. More over it is platform independent and portable.

Editors and viewers

Online Markdown editors

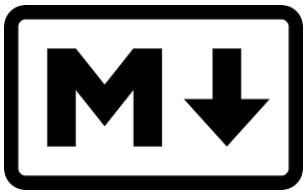
- Dillinger https://dillinger.io/
- StackEdit https://stackedit.io/

Open source Software

- MarkText https://www.marktext.cc/
- ghostwriter https://kde.github.io/ghostwriter/

Extensions availabable of VScode

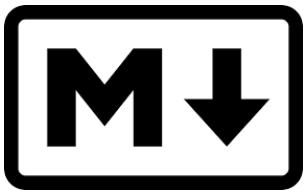
Markdown language (3/8)



Quick reference guide

Markdown	Preview
`**bold text**`	**bold text**
`*italicized text*` or `_italicized text_`	*italicized text*
`` `Monospace` ``	`Monospace`
`~~strikethrough~~`	~~strikethrough~~
`[A link](https://www.google.com)`	A link
`![An image](https://www.google.com/images/rss.png)`	

Markdown language (4/8)



Headings are rendered as titles

```
# Section 1
# Section 2
## Sub-section under Section 2
### Sub-section under the sub-section under Section 2
# Section 3
```

Section 1

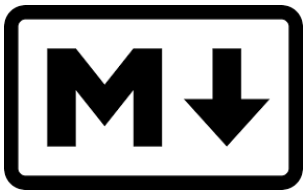
Section 2

Sub-section under Section 2

Sub-section under the sub-section under Section 2

Section 3

Markdown language (5/8)



Block code

```
```python
print("a")
```
```

```python
print("a")
```

Lists

Ordered lists:

```
1. One
1. Two
1. Three
```

1. One
2. Two
3. Three

Unordered lists:

```
* One
* Two
* Three
```

- One
- Two
- Three

Markdown language (6/8)

Tables

```
First column name   | Second column name
--------------------|-------------------
Row 1, Col 1        | Row 1, Col 2
Row 2, Col 1        | Row 2, Col 2
```

First column name	Second column name
Row 1, Col 1	Row 1, Col 2
Row 2, Col 1	Row 2, Col 2

Inline equations

```
$y=x^2$

$e^{i\pi} + 1 = 0$

$e^x=\sum_{i=0}^\infty \frac{1}{i!}x^i$

$\frac{n!}{k!(n-k)!} = {n \choose k}$
```

$y = x^2$

$e^{i\pi} + 1 = 0$

$e^x = \sum_{i=0}^{\infty} \frac{1}{i!} x^i$

$\frac{n!}{k!(n-k)!} = \binom{n}{k}$

Markdown language (7/8)

Equation blocks

```
$$
y=x^2
$$

$$
e^{i\pi} + 1 = 0
$$

$$
e^x=\sum_{i=0}^\infty \frac{1}{i!}x^i
$$

$$
\frac{n!}{k!(n-k)!} = {n \choose k}
$$

$$
A_{m,n} =
 \begin{pmatrix}
  a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\
  a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\
  \vdots  & \vdots  & \ddots & \vdots  \\
  a_{m,1} & a_{m,2} & \cdots & a_{m,n}
 \end{pmatrix}
$$
```

$$y = x^2$$

$$e^{i\pi} + 1 = 0$$

$$e^x = \sum_{i=0}^{\infty} \frac{1}{i!} x^i$$

$$\frac{n!}{k!(n-k)!} = \binom{n}{k}$$

$$A_{m,n} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{pmatrix}$$

Markdown language (8/8)

Example avec Github view

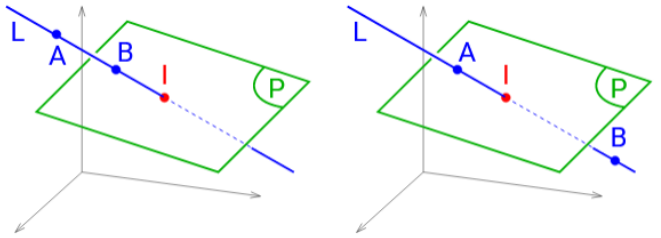
```
### Intersection plan-line
Notations
- $A(x_A, y_A, z_A)$  and  $B(x_B, y_B, z_B)$  are two points
- $L = \overleftrightarrow{AB}$ is a line defined by points A and B
- $P$ is a plan defined by the following equation  $M(x,y,z) \in P$   iff   $ax + by + cz = d$
- $I(x_i, y_I, z_I)$ is the point intersection common to line $L$ and plan $P$
- $S = \overline{AB}$ is a line segment bounded by points A and B

![Fig](img/plan_segment.s
```

Intersection plan-line

Notations

- $A(x_A, y_A, z_A)$ and $B(x_B, y_B, z_B)$ are two points
- $L = \overleftrightarrow{AB}$ is a line defined by points A and B
- P is a plan defined by the following equation $M(x, y, z) \in P$ iff $ax + by + cz = d$
- $I(x_i, y_I, z_I)$ is the point intersection common to line L and plan P
- $S = AB$ is a line segment bounded by points A and B



Jupyter notebook (1/2)

What is a notebook ?

It is a documents containing

- Code (Python, Matlab, …)
- Equations (Markdown)
- Visualizations (results of code execusion)
- Narrative text (Markdown)

What is a Jupyter ?

The Jupyter Notebook is an open-source web application for creating and sharing a notebook

Jupyter notebook (1/2)

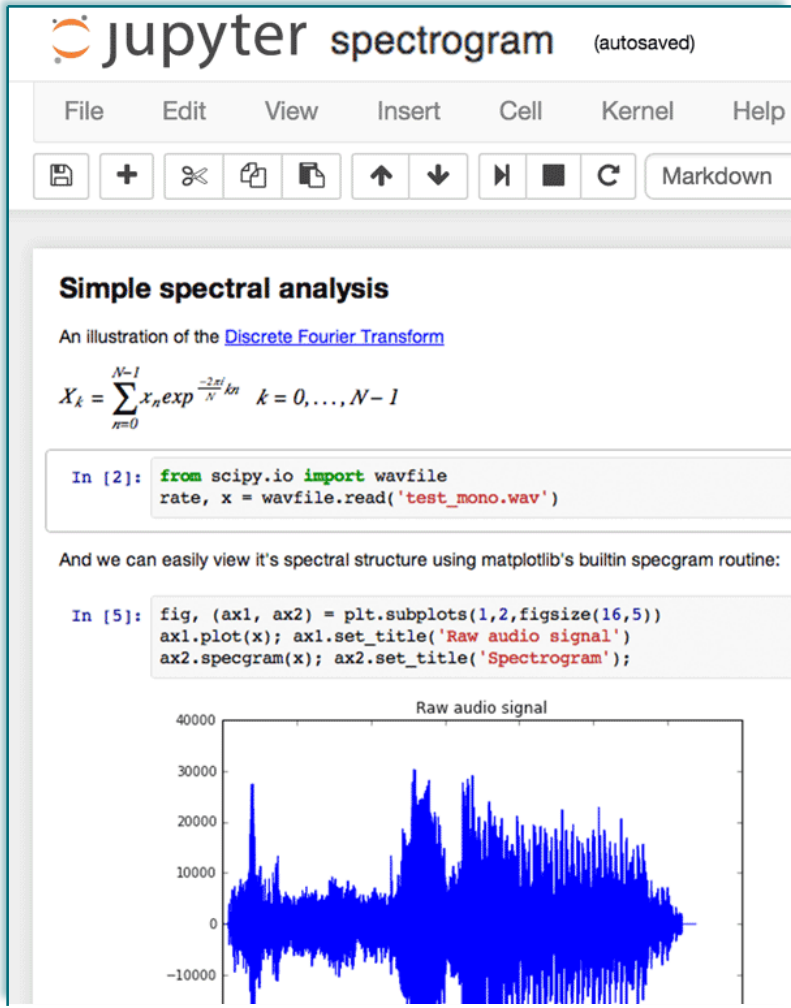
What is a notebook ?

It is a documents containing

- Code (Python, Matlab, …)
- Equations (Markdown)
- Visualizations (results of code execusion)
- Narrative text (Markdown)

What is a Jupyter ?

The Jupyter Notebook is an open-source web application for creating and sharing a notebook



Jupyter notebook (2/2)

Installation guide

Jupyter note book is included in Anaconda distribution

- To use Jupyter with Python : no extra installation after Anaconda installation

- To use Jupiter with Matlab
  - Matlab use its own notebook format (included with Matlab installation) but it can not be render by Github.com (proprietary format)
  - Access Matlab directly from Jupyter install a dedidated Python module

    ```
    > python3 -m pip install jupyter-matlab-proxy
    ```

    (see https://www.mathworks.com/products/reference-architectures/jupyter.html)

Plan

1. Introduction
2. Qu'est-ce que l'on entend par « preuve de concept »
3. Compétences attendues à l'issue du module
4. Présentation des outils utilisés
5. Présentation des ressources

Ressources à disposition

Toutes les informations serons disponibles sur eCampus (lien)

- Slides
- Liens pour les formations Python, Matlab, Git
- Exercices d'application et solutions
  - Séance 3, 4 et 5
  - Exemple type d'évaluation

Exemple de projet sur GitHub

- Projet présenté par Bruno Denis et étudié lors de la séance 4
- Projet en libre accès dès la fin de la séance 4

Formation Python

Installation guide

- Download Anaconda distribution from https://www.anaconda.com/products/distribution
- Select « Python 3.X » version at the installation beginning

Resources

- Tutorials for beginners
  - https://docs.python.org/3/tutorial/index.html (free access, many languagues)
  - https://www.learnpython.org/
    - For Numpy → (https://www.learnpython.org/en/Numpy_Arrays)

Formation Matlab

Installation

- Matlab version R2023a
- Attention l'installation de Matlab nécessite d'avoir une adresse mail ens-paris-saclay.fr et donc de s'être inscrit.

Ressources à disposition

- Formation en ligne (faut un compte matworks et donc une adresse mail ens)
  - https://matlabacademy.mathworks.com/fr/
  - MATLAB Onramp, MATLAB Fundamentals, MATLAB for Data Processing and Visualization
  - Pour aller plus loin MATLAB Programming Techniques
- Autres
  - Document sur les bonnes pratiques
  - Fichier .mlx présentant quelques bases
  - Site matlab.developpez.com (lien 1) (lien 2)
  - Matlab Centrale
  - Chaine YouTube de Matlab

Formation Git

Installation

- Installer la version d'évaluation de Fork qui est gratuite et à durée illimitée.
- Cette version permet d'avoir accès à un dépôt externe privé gratuitement contrairement à bon nombre d'interfaces graphique Git.

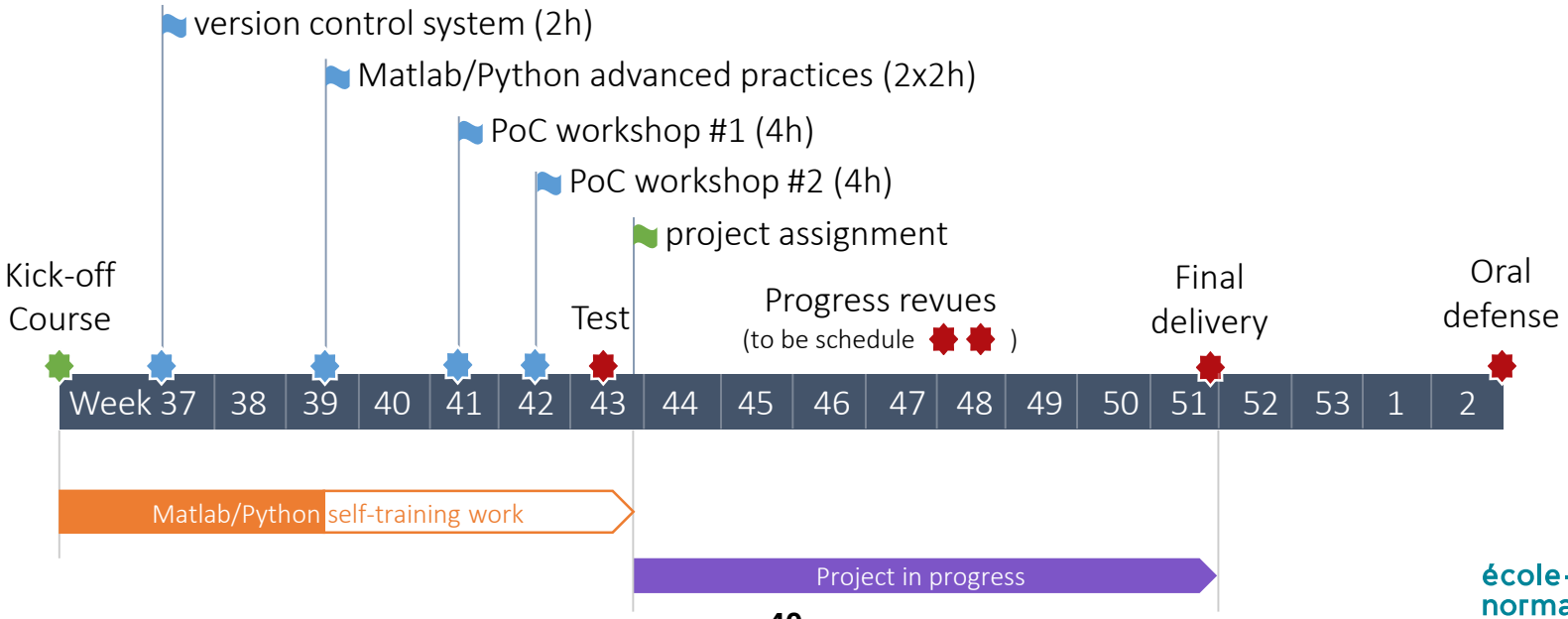
Ressources à disposition

- Séance 2
- Formation git (en ligne de commande) : https://grafikart.fr/formations/git
- Documentation git https://git-scm.com/book/en/v2

Conclusion

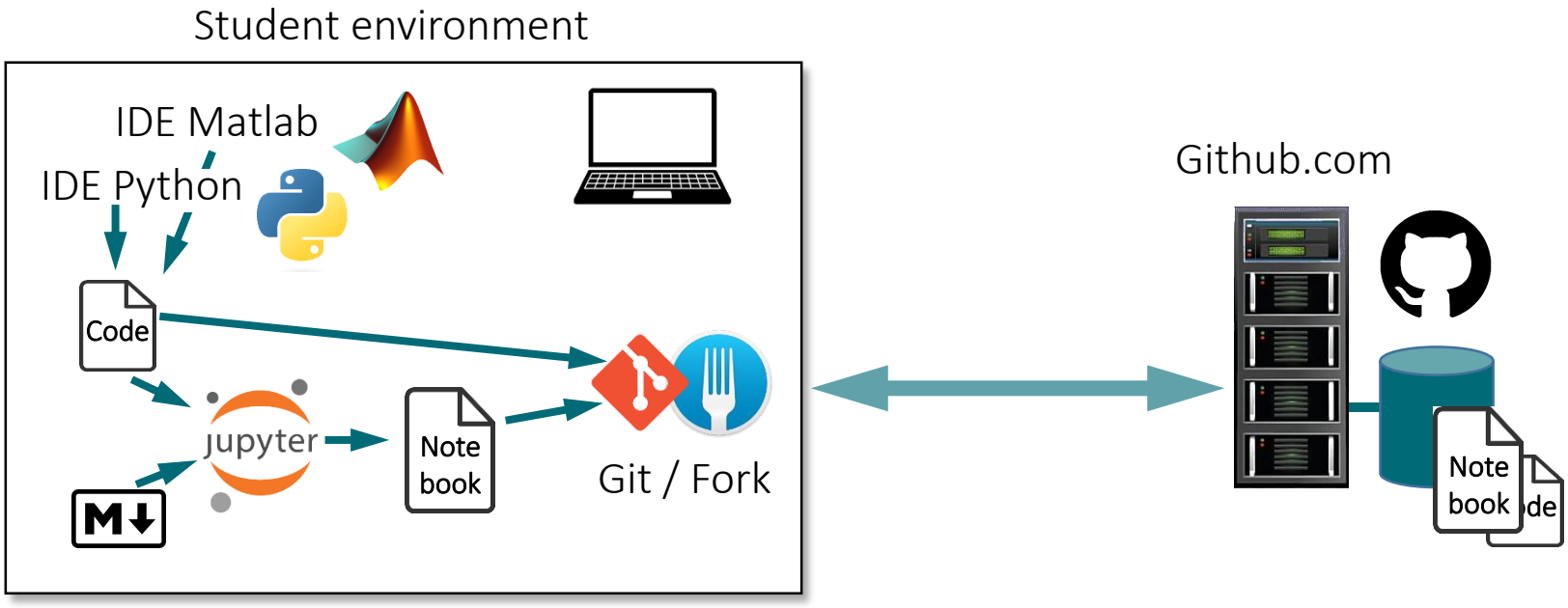
Schedule

- Kick-off teaching module
- Versioning tutorial
- Matlab/Python advanced practices
- 2 x PoC Workshops
- Programming test
- Presentation and project assignment
- Progress review session
- Project delivery
- Project oral defense



Conclusion

IT organization overview



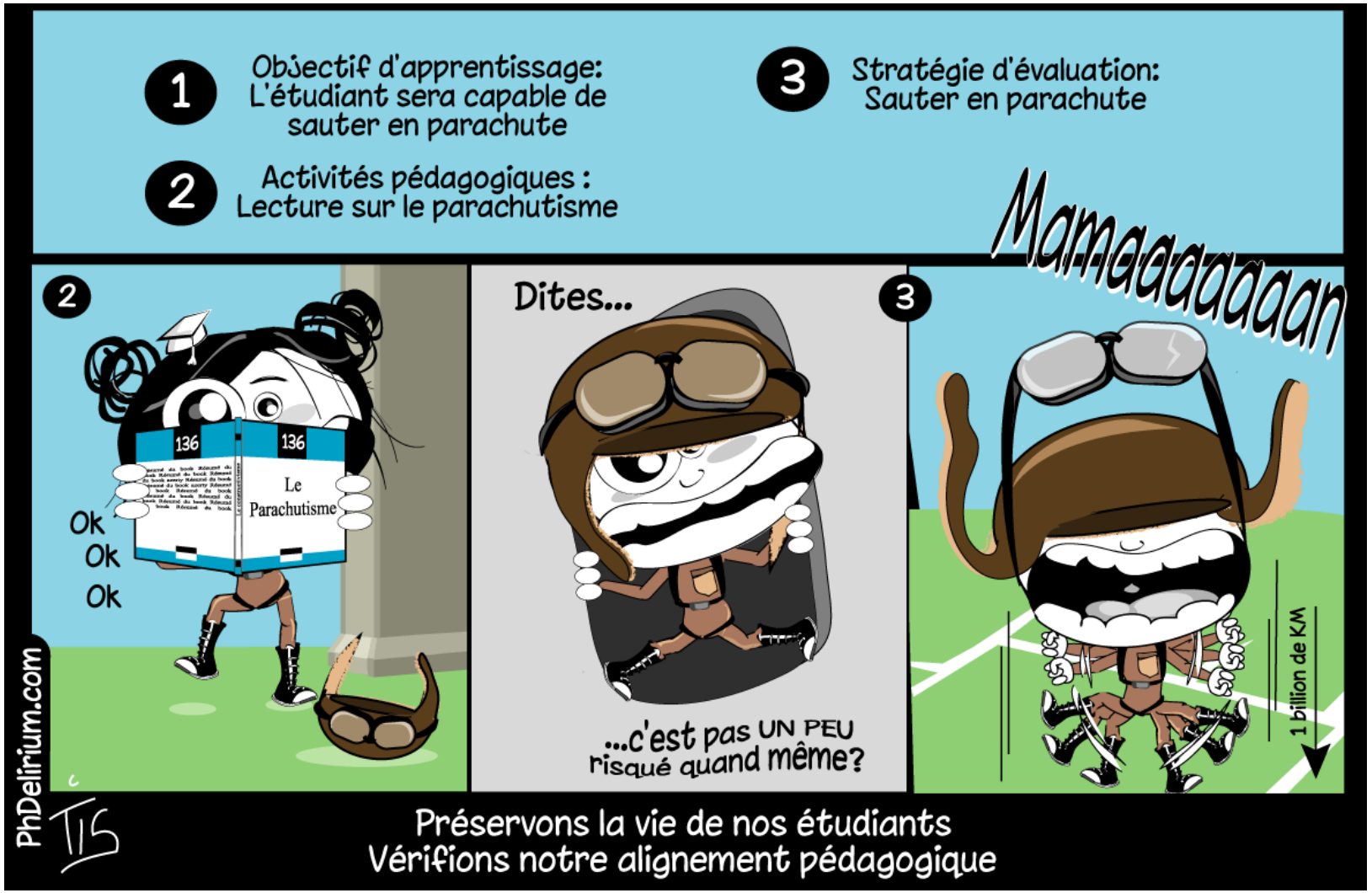
Conclusion

Pour la séance du 15/09/2023 (obligatoire)

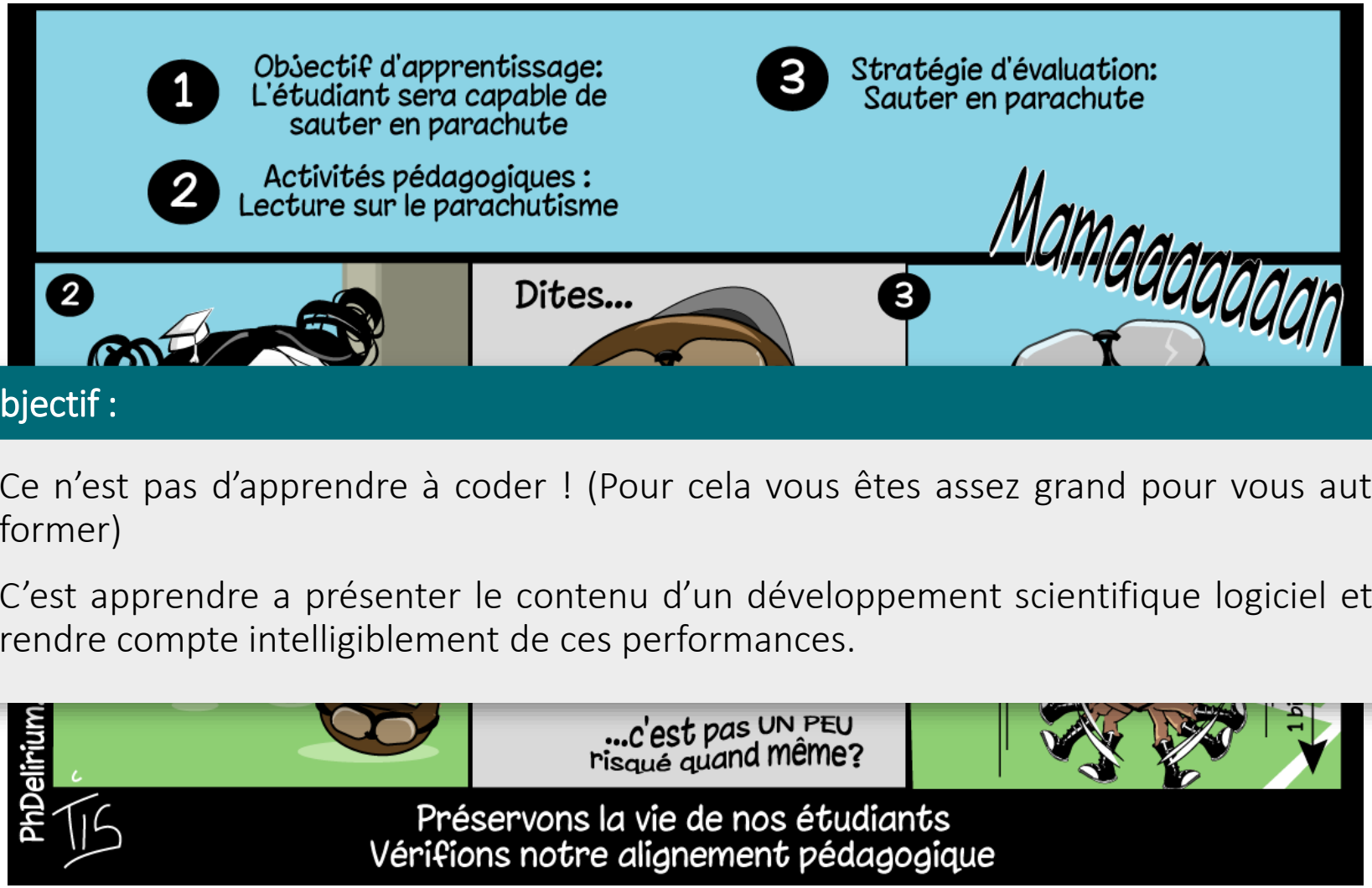
1. Installer Fork (lien)
2. Créer un compte github (lien)
3. Envoyer votre login github à l'adresse mail kevin.godineau@ens-paris-saclay.fr

Questions ?

Pourquoi tout cela !



Pourquoi tout cela !



Objectif :

- Ce n'est pas d'apprendre à coder ! (Pour cela vous êtes assez grand pour vous auto-former)
- C'est apprendre a présenter le contenu d'un développement scientifique logiciel et à rendre compte intelligiblement de ces performances.