Septembre 2022
révision été 2023

Matériel

En résumé (sans lire le reste)←——————————————————————Matériel

Indispensable Ordinateur portable sous GNU/linux, cartes type esp32

Facilitateur BYOD, un ou deux routeurs, Raspberry-Pi, cartes micro:bit, visualiseur USB (voir partie « logiciel »)

Ordinateur portable Internet offre une grande variété d'outils, sous forme de sites, qui permettent de ne pas être dépendant d'un ordinateur sous GNU/linux. Néanmoins de nombreux sites nécessitent une inscription préalable, qu'on ne peut pas exiger des élèves. C'est pourquoi l'ordinateur sous GNU/linux fourni par la région reste incontournable.

On peut toutefois regretter que la maintenance de ces postes soit dévolue aux enseignants de NSI. Compte-tenu de l'évolution constante des distributions GNU/linux, cette maintenance peut être envisagée comme annuelle : faire un *master* et déployer celui-ci sur les postes.

Il est à noter que la distribution GNU/linux installée nécessite un travail régulier de mise à jour. Même si on peut considérer comme formateur de déléguer cette tâche aux élèves, cette pratique s'avère trop chronophage et éloignée des préoccupations du quotidien. Ainsi la mise à jour annuelle par l'enseignant est véritablement inévitable.

Dans l'esprit *Bring Your Own Device* (BYOD), on peut proposer aux élèves d'utiliser leur propre ordinateur portable. Cela renforce leur autonomie et permet une appropriation rapide des outils utilisés au-delà des heures de cours.

Objets connectés

Raspberry Pi Le `R-pi` pourrait être une alternative crédible à l'ordinateur portable. Il fournit un environnement sous GNU/linux et est bien adapté à l'utilisation des objets connectés via son interface GPIO. Prêté pendant un an aux élèves, il permet à l'élève de disposer d'un OS libre « à la maison ».

Toutefois cette solution n'évite pas le rafraîchissement annuel de la distribution GNU/linux.

BBC micro:bit Les capteurs intégrés de la `micra bit` (notamment la v2) fournissent une plateforme très intéressante pour une utilisation immédiate. Son système simple de connexion à d'autres capteurs/actionneurs et les nombreuses possibilités de simulation sur Internet font de lui un incontournable pour une sensibilisation aux objets connectés.

Notons que dans la version 1, utilisée par beaucoup d'établissements, le `Bluetooth` ne fonctionne pas avec `Micropyhton`, la communication reste possible entre deux cartes en utilisant le protocole `radio`.

Néanmoins, cette plateforme ne dispose pas d'entrée d'un module communiquant de type WiFi (existant toutefois sous la forme d'une carte d'extension).

esp8266-esp32 Les cartes basées sur ces SoC sont évolutives, polyvalentes et permettent une connexion à un réseau via le WiFi.

Associées au système *Grove*, elles fournissent une base solide, et sans vraie connaissance électronique à acquérir, pour s'approprier les objets connectés.

Routeur Un routeur permet de monter à partir d'objets connectés et des ordinateurs portables en WiFi un réseau indépendant. Deux routeurs permettent de créer une situation plus complexe de communication. La simulation avec un logiciel dédié reste indispensable pour créer des situations plus riches.

Liens

- Site officiel Rapberry Pi (en)
- Site officiel micro:bit (en)
- Doc micropython pour esp8266 (en)
- Doc micropython pour esp32 (en)
- Description du système Grove (en)

Logiciel

Nous ne détaillons pas ici, les sites Internet permettant de simuler une architecture RISC, ou de simuler les fonctions logiques ou etc.
Préambule La crise sanitaire nous a montré à quel point l'accès aux outils logiciels en ligne peut être pratique et permettre une continuité des apprentissages. Ainsi certaines plateformes proposent des **E**nvironnements de **D**éveloppement **I**ntégré dans une grande variété de langages, dont `Python`, `JavaScript`, `html` et `css`.

Python

En résumé ← Python

Indispensable accès à Capytale ou à un serveur Jupyter, pouvoir installer des bibliothèques `Python`, système en ligne de partage simple et efficace de fichiers (de code), outils bureautiques

Facilitateur `Edupython`, `VS Code`, `Edupyter`, `thonny`, `Mu` (pour les objets connectés) etc.

Il est difficile de conseiller une «bonne pratique» pour programmer en Python. Pour l'installation sur les ordinateurs portables, une approche assez simple basée sur la version de `Python` 3 fournie par les dépôts du système semble à privilégier. Les modules et bibliothèques supplémentaires sont installés via les dépôts de la distribution GNU/linux ou l'utilitaire `pip` sous environnement virtuel Python.

En première Disposant de suffisamment de PCs fixes, on peut privilégier leur utilisation pour des raisons de simplicité (les élèves connaissent cet environnement et on a accès au réseaux du lycée et à des outils intéressants comme le verrouillage des postes) et de maintenance (mise à jour et problèmes techniques réalisées par l'informaticien).

Les PCs portables région permettent de traiter la partie GNU/linux, et peuvent servir à quelques projets en fin d'année pour les élèves qui le souhaitent.

En Terminale
- les exercices utilisent souvent un serveur `Jupyter`, utilisé en ligne (Capytale) ou sur les ordinateurs portables. L'ouverture de `Capytale` est une vraie plus-value dans la pratique, notamment pour la possibilité donné à l'enseignant de commenter le notebook et/ou de le noter ;
- les élèves sont libres d'utiliser l'EDI qu'ils souhaitent. Dans la pratique, certains restent attachés au logiciel qu'ils ont découvert en classe de première : `EduPython`, d'autres utilisent *Visual Studio Code* sous GNU/linux ou encore `Jupyter` ;
- la réalisation des projets nécessite parfois des bibliothèques ou modules un peu *exotiques* souvent impossibles à installer dans des réseaux d'établissement très sécurisés. Donc les besoins très spécifiques passent par le BYOD, l'ordinateur portable sous GNU/linux ou des sites (collaboratifs) comme `repl.it` ;
- les projets de groupe des élèves nécessitent de pouvoir partager aisément du code, de pouvoir y ajouter un carnet de bord, voire de pouvoir modifier le code directement sur l'espace de partage ; une solution basée sur un

`git` en ligne avec une interface graphique peut être privilégiée. De plus la présentation des projets exige un diaporama composé avec les outils bureautiques usuels ;
- lors de l'épreuve pratique, les élèves retrouvent leur environnement favori (hors site internet) pour composer.

Liens

- Aide officielle Capytale (fr)
- Site officiel EduPython (fr)
- Site officiel Visual Studio Code (en)
- Démarrer avec GitHub (fr)
- Livre gratuit sur Git

Sur les entrées du programme

Histoire de l'Informatique rien de spécifique, outils bureautiques

Structures de données

En résumé←Structures de données

Indispensable rien

Facilitateur installation du langage `dot` sur l'ordinateur portable

L'appropriation des arbres et des graphes peut passer par une conception visuelle à l'aide du langage `dot`, par exemple. L'installation sur l'ordinateur portable de ce langage permet avec un noyau spécifique de `Jupyter` de l'utiliser agréablement. Néanmoins des sites internet sans inscription permettent de programmer en `dot`.

Base de données

En résumé←Base de données

Indispensable rien

Facilitateur une installation de `MySQL`, voire d'un serveur `lamp`

Un SGBD de type `MySQL` ou `MariaDB` permet de disposer d'une ligne de commande SQL, celle-ci n'est toutefois pas indispensable pour la pratique, des sites internet sans inscription préalable offrant l'opportunité de tester des clauses SQL.

On peut mener la plupart des projets avec `sqlite3` faisant partie de la bibliothèque standard de `Python`. Pour les plus rares projets utilisant un serveur Web, `lamp` peut être installé.

Architectures matérielles, système d'exploitation et réseaux

Composants intégrés sur puce

En résumé←Composants…

Indispensable R-Pi et/ou esp32 et/ou micro:bit (voir partie « Matériel »)

Facilitateur visualiseur USB (voir partie « Matériel »), serveur `Jupyter`, éditeur `Mu`

une caméra USB type visualiseur permet de videoprojeter en gros plans des SoC, de cartes mères etc. en direct.
Pour les objets connectés type cartes esp, on peut prévoir d'installer un noyau `MicroPython` pour `Jupyter` dans l'ordinateur portable ou d'utiliser l'éditeur `Mu`.

Gestion de processus et des ressources par un système d'exploitation

En résumé ← Système Exploitation

Indispensable terminal sous GNU/linux

Facilitateur rien

Protocoles de routage

En résumé ← Protocoles de routage

Indispensable Filius

Facilitateur un ou deux routeurs (voir partie « Matériel »)

Langages et programmation

En résumé ← Langages et programmation

Indispensable terminal sous GNU/linux

Facilitateur rien

Un terminal type **sh** prend du sens dans cette entrée : « notion de programme en tant que donnée ».

L'accès à d'autres langages permet de découvrir des langages de paradigmes différents de `Python`, de nombreux sites proposent des interprétations et des compilations dans quasiment tous les langages, néanmoins l'installation directe sur l'ordinateur portable peut permettre une découverte plus sensée.

Algorithmique rien de spécifique

Liens

- Site officiel de graphviz/langage dot (en)
- Graphviz en ligne
- Doc officielle sqlite3 (en)
- Site officiel de l'éditeur Mu (en)
- Installation de lamp sur Ubuntu (fr)
- Site sur les visualiseurs
- Le shell Bash (fr)
- Prise en main de filius sur l'académie de Bordeaux (fr)

Autres

Sur l'ordinateur portable, des outils complémentaires peuvent être installés. En voici une liste non exhaustive :

bluefish, spyder, thonny, geany, pyzo etc. divers éditeurs

Arduino Site officiel Arduino (en)

Wireshark logiciel d'analyse de protocoles Site officiel (en)

Un exemple de reverse-engineering sur USB avec Wireshark (fr)

Freeplane conception de cartes heuristiques Doc Ubuntu (fr)

KeePassXC gestionnaire de mots de passe Site officiel (en)

GeoGebra logiciel de géométrie dynamique Site officiel GeoGebra (fr)

Dia conception de diagrammes Site officiel (en)