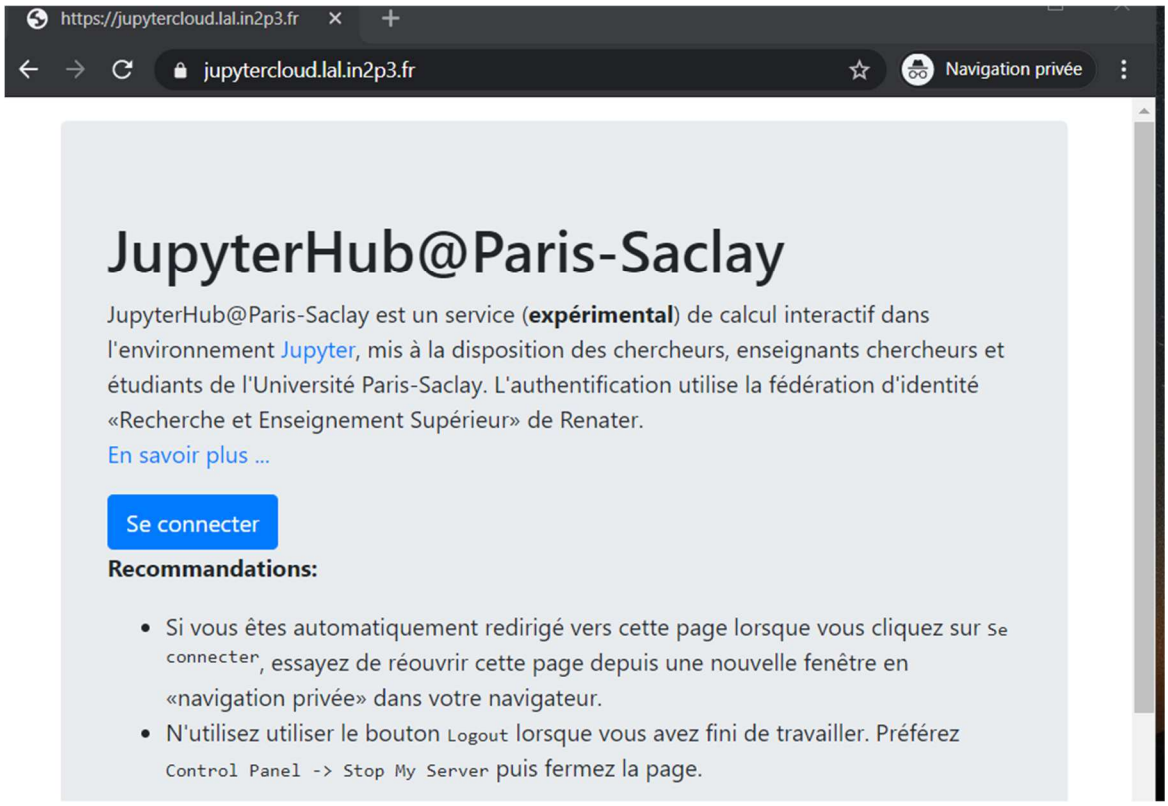
Introduction - Connexion au serveur Jupyter

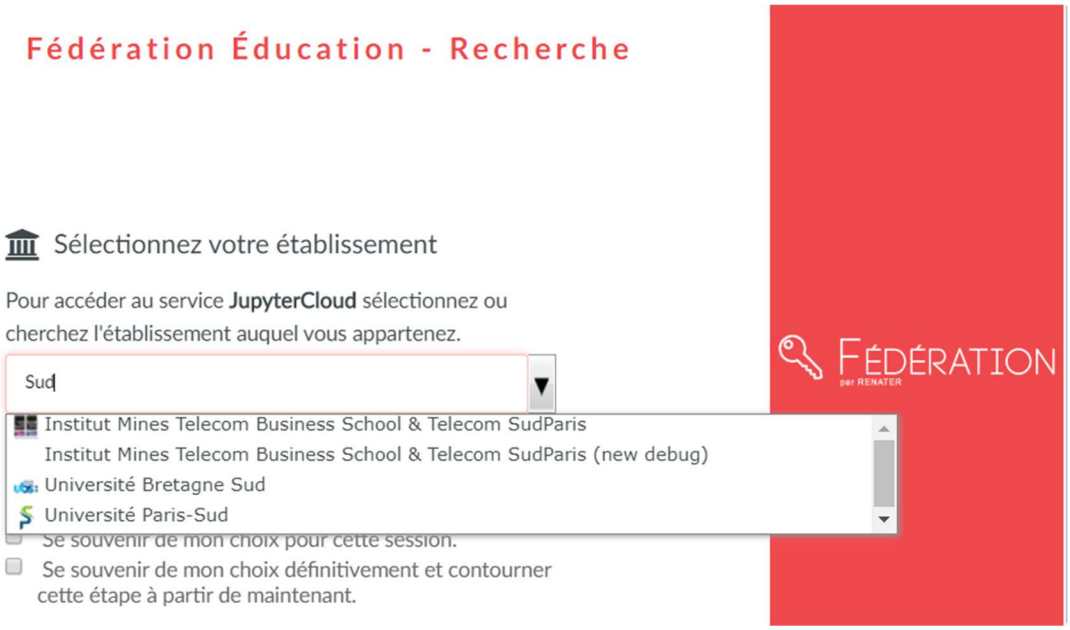
Dans votre navigateur, ouvrez la page dont l'URL est :

https://jupytercloud.lal.in2p3.fr/

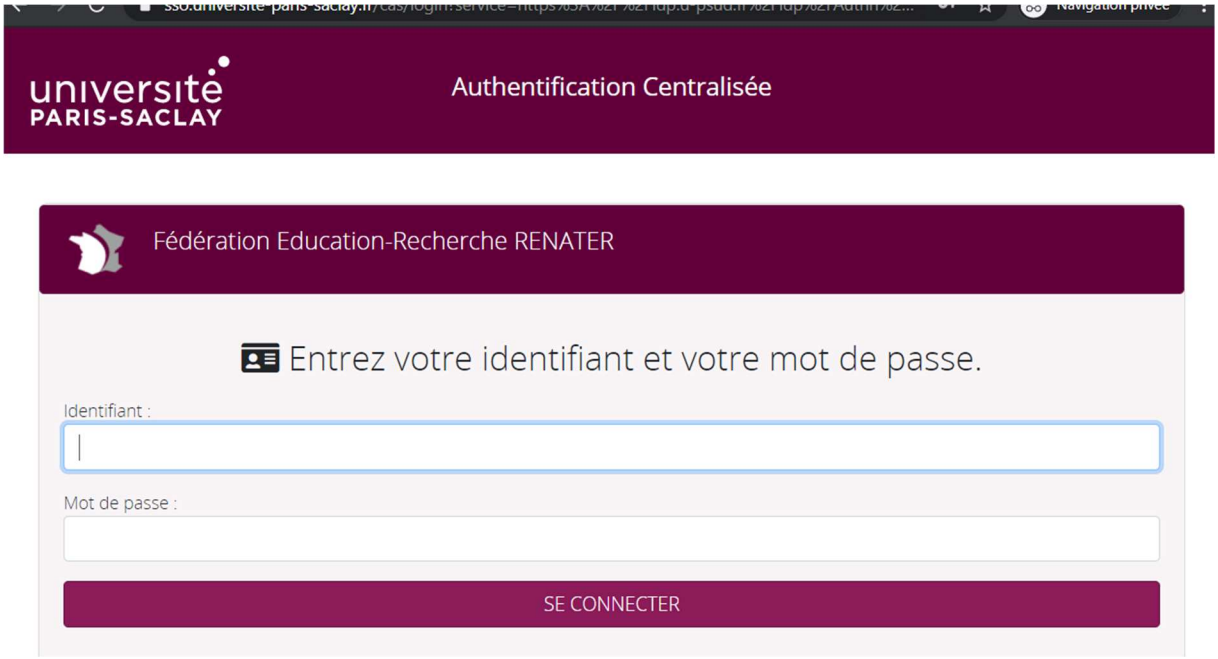
Vous arrivez sur la page d'accueil du serveur Jupyter de Paris-Saclay : cliquez sur le bouton `Se connecter` :



Cas favorable *:* Si tout se passe bien, vous arrivez sur une page de connexion commune à de nombreux établissements d'enseignement et de recherche:

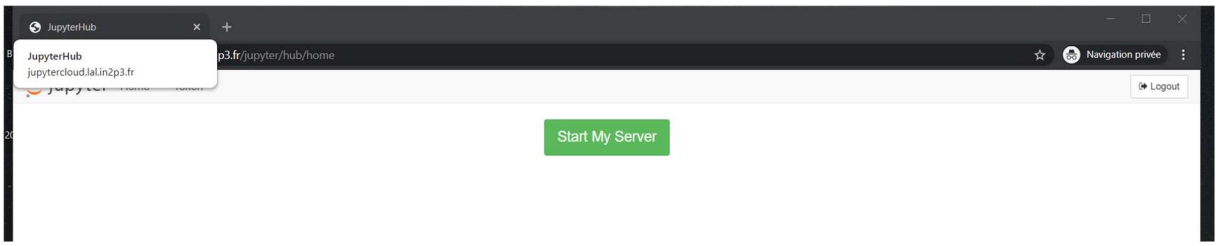


Filtrez la liste en tapant `Sud`, sélectionnez `Université Paris-Sud`, cliquez sur `Sélection` et utilisez vos identifiants Adonis dans la page d'authentification



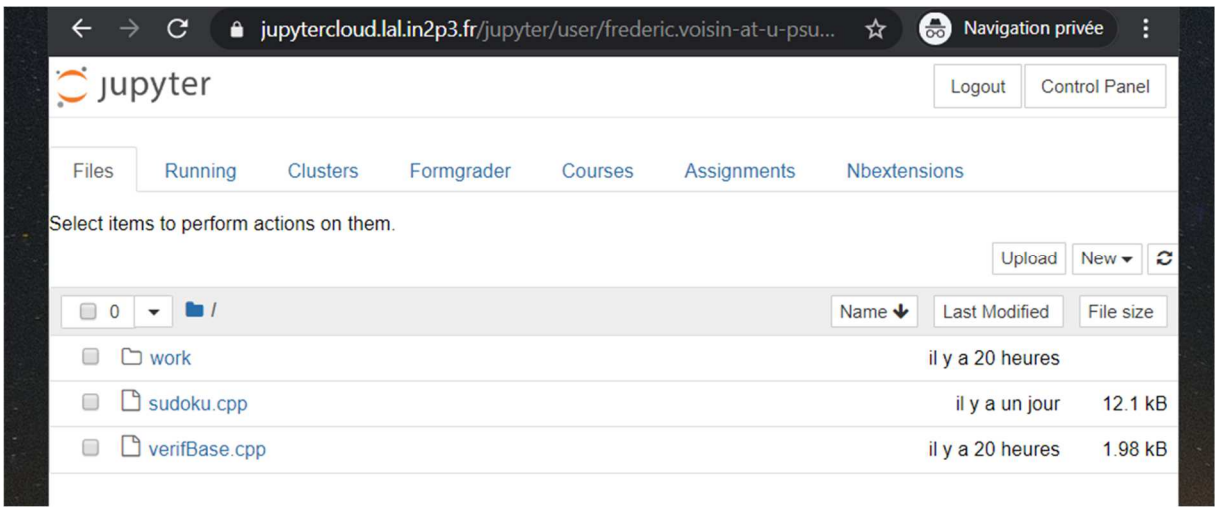
Cas défavorable *:* dans le premier écran, le clic sur « `Se connecter` » boucle sur la même page. Revenez à l'URL de départ, faites un « clic droit » et utilisez « `Ouvrir le lien dans une fenêtre de navigation privée` ». Vous arrivez alors sur la page de recherche des établissements et vous pouvez poursuivre la procédure.

I – Généralités : sur la page d'accueil après connexion, on clique pour démarrer son serveur :



On arrive sur sa « page personnelle » qui comprend :

L'espace personnel de fichiers, qu'on peut organiser avec des sous-répertoires si on le souhaite. Ci-dessous, il contient un sous-répertoire `work` et deux fichiers `sudoku.cpp` et `verifBase.cpp` :



Si vous sélectionnez des éléments, des commandes apparaissent au-dessus de l'arborescence des fichiers : *dupliquer, renommer, déplacer, télécharger, visualiser, éditer, supprimer* (ou leur équivalent en anglais).

Pour créer un nouveau répertoire : placez-vous dans le répertoire où vous voulez faire l'ajout (ici, soit le répertoire d'accueil, soit `work`) puis, dans le menu `New` à droite, sélectionnez `Folder`. Il ne vous reste plus qu'à sélectionner le nouveau répertoire (`Untitled Folder`) et à le renommer.

Des onglets :

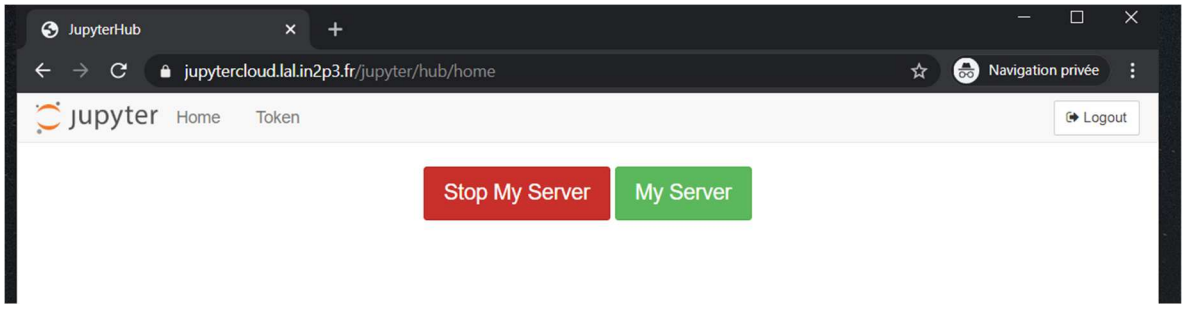
- `Files` : gestion des fichiers et répertoires
- `Running` : les terminaux et « notesbooks » en activité sur votre serveur (voir plus loin)

Nous n'aurons pas l'usage des autres onglets.

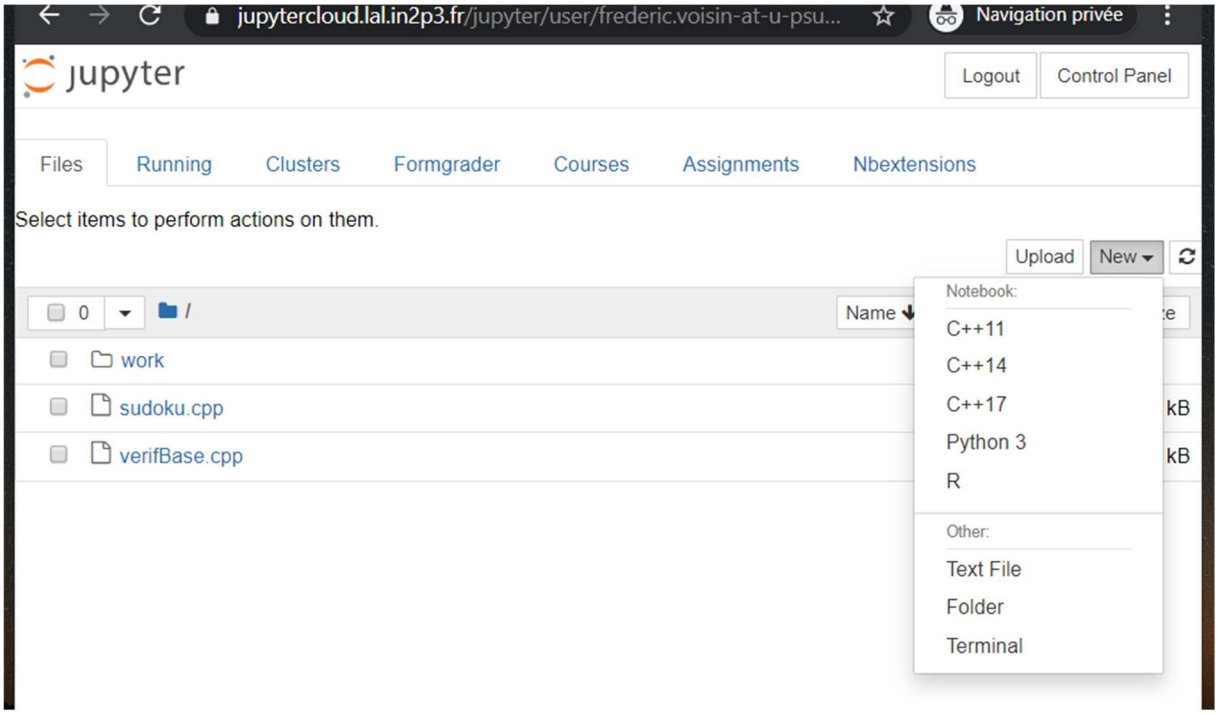
Des boutons :

- Le bouton « `Control Panel` » en haut à droite sert à fermer le serveur avant de quitter Jupyter. *Note* : le bouton « `Logout` » qui apparaît sur ces copies d'écran à gauche de « `Control Panel` » était source d'erreurs et n'apparait plus dans la version actuelle du système !
- Le bouton `Upload` permet de télécharger sur votre serveur un fichier de votre ordinateur. Après avoir sélectionné votre fichier, cliquez sur le bouton « `Téléverser` » qui apparaît sur la page Jupyter pour confirmer le chargement.
- Le menu « `New` » permet de lancer les principaux outils de travail : l'éditeur de texte, le terminal dans lequel nous compilerons et exécuterons nos programmes et les « notebooks C++ ». Nous présentons très rapidement chacun de ces outils dans les rubriques qui suivent.

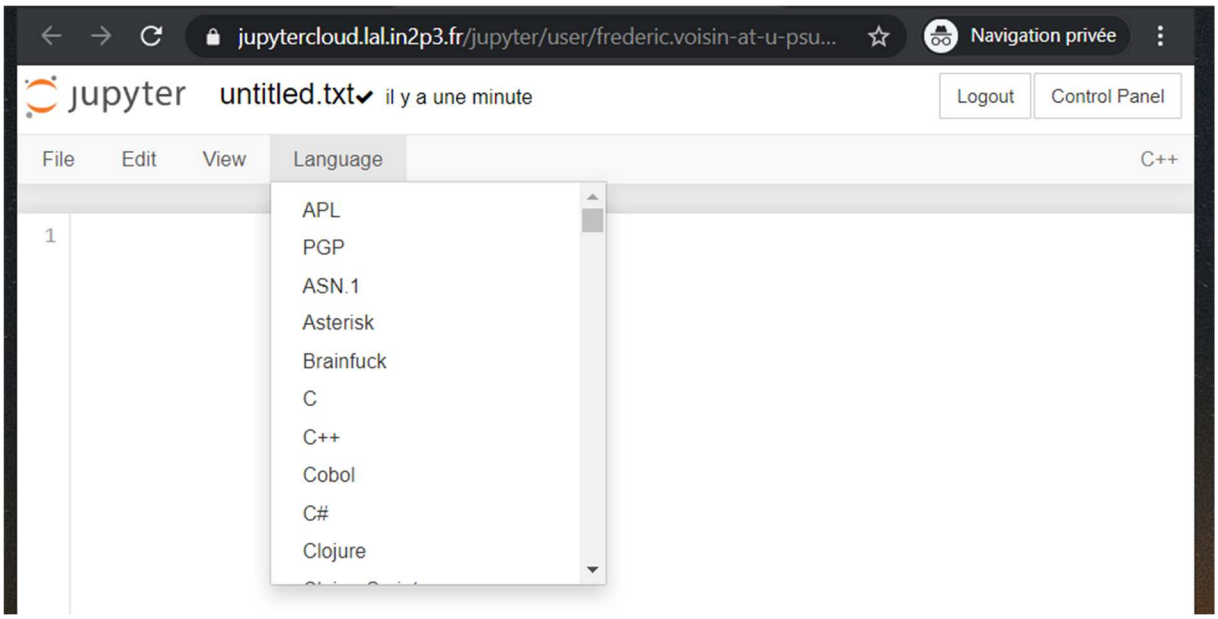
Pour quitter le serveur, cliquez sur « `Panel Control` » de la fenêtre précédente puis sur « `Stop My Server` ». Vous pouvez ensuite vous octroyer un repos bien mérité.



II - L'éditeur du texte : il est accessible via `Text File` dans le menu New.

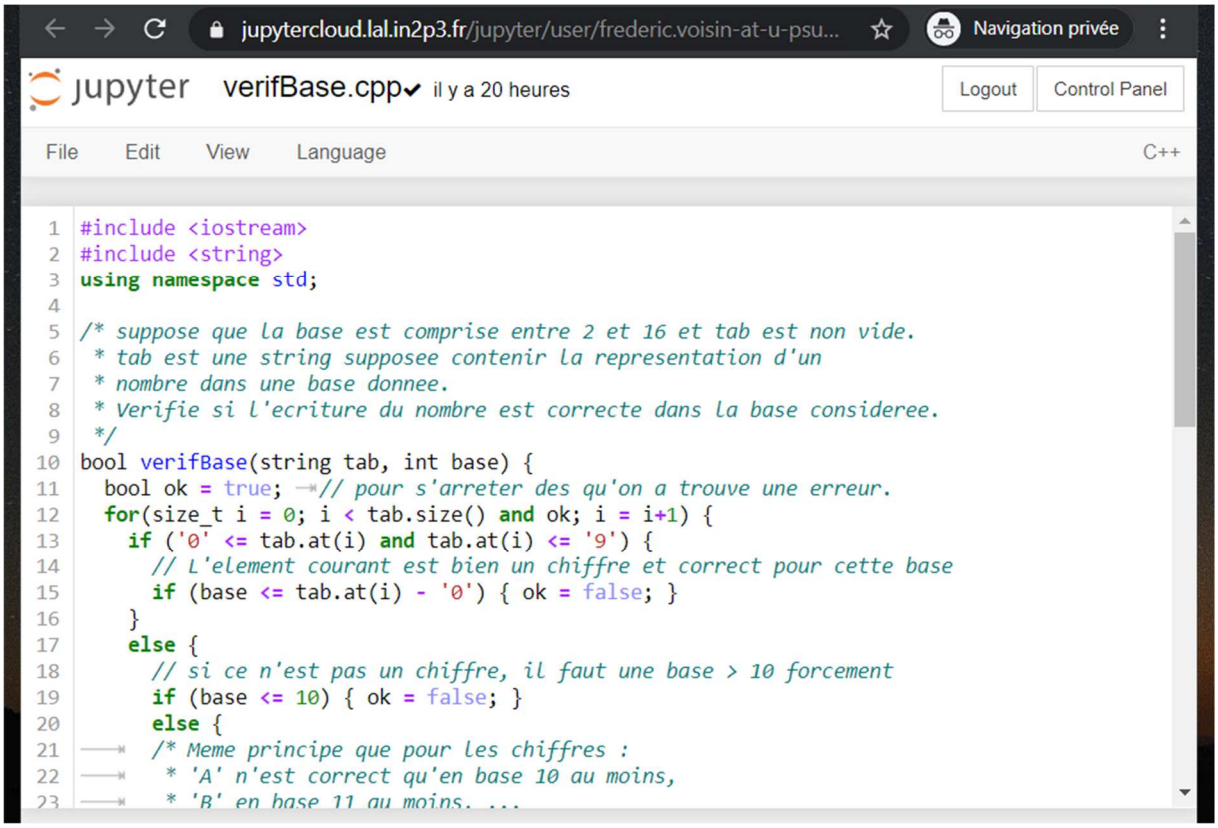


Si vous passez par le menu New de l'espace personnel, vous arrivez sur une page vide. Renommez-la (menu `File` de l'éditeur) et mettez-la en mode C++ (menu `Language`).



Voilà, votre vrai travail commence !

L'éditeur peut être lancé à partir de l'espace de fichiers en sélectionnant le fichier et en utilisant le bouton `Edit` qui apparaît parmi les commandes applicables au fichier. Votre fichier est automatiquement ouvert dans le mode C++ s'il a l'extension `.cpp`



III - Le terminal : accessible à partir du menu `New` de l'espace personnel, il permet de lancer des commandes. Seules deux commandes nous intéressent ici : compiler un programme, et l'exécuter une fois que la compilation a réussi.

Attention, vous n'êtes pas dans un environnement (comme CodeBlocks) avec une recompilation automatique. C'est à vous de penser à recompiler un fichier chaque fois que vous l'avez modifié, sinon le résultat de la compilation contient toujours la version précédente.

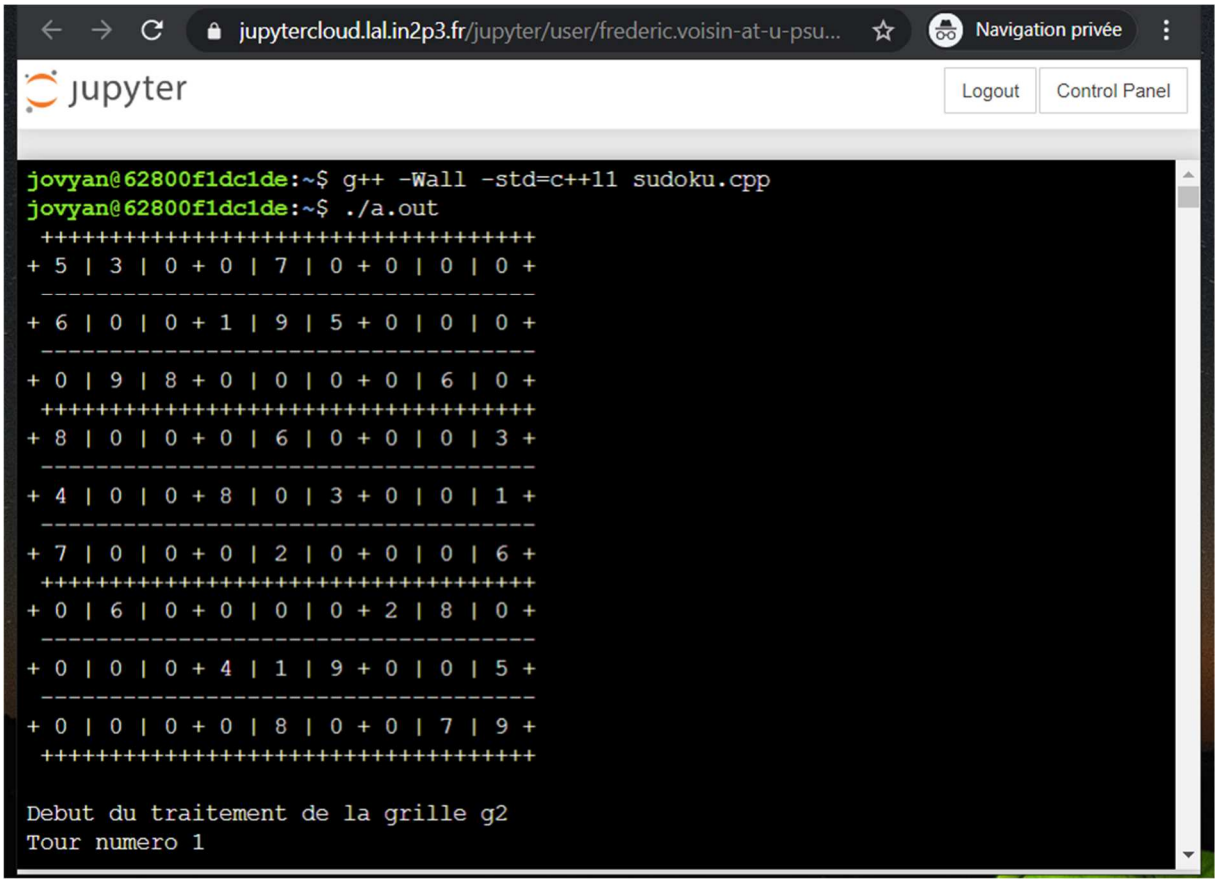
Compiler un programme : le compilateur s'appelle `g++`. Les options d'appel sont les mêmes que pour CodeBlocks : `-Wall` et `-std=c++11`. On fournit en plus (à la fin) le nom du fichier à compiler, par exemple notre fichier `sudoku.cpp`. La commande complète à taper dans le terminal est donc

```
g++ -Wall -std=c++11 sudoku.cpp
```

suivi de la touche *Entrée* pour la lancer. Si tout se passe bien (pas d'erreur de compilation), il n'y a aucun message mais le compilateur produit un fichier exécutable nommé `a.out`. Sinon, on obtient les messages d'erreur du compilateur (attention, s'il existait un fichier `a.out` résultat d'une compilation précédente réussie, celui-ci n'est pas effacé mais n'est plus à jour).

Jupyter n'est pas un environnement de développement intégré, donc pas d'indication dans l'éditeur de texte des lignes avec des erreurs. Il faut suivre les messages du compilateur (et on rappelle qu'on commence **par lire les messages du début**, ceux qui suivent pouvant parfois découler d'une erreur précédente).

Exécuter un programme : Si la compilation a réussi, on lance l'exécution du programme par la commande `./a.out` suivie de la touche `Entrée`. Apparait alors sur le terminal, ce que le programme a prévu qu'il apparaisse !



On peut choisir le nom du fichier résultat de la compilation, grâce à l'option `-o` suivie du nom de fichier :

```
g++ -Wall -std=c++11 -o verifBase verifBase.cpp
```

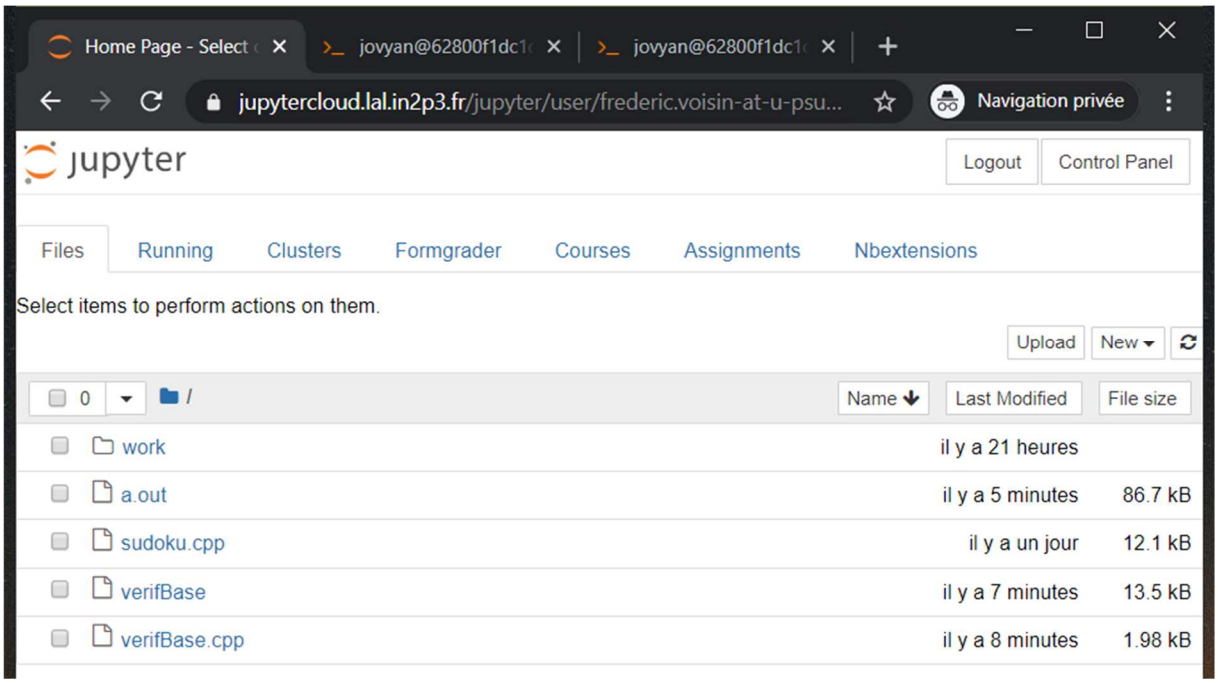
On lance alors l'exécution par : `./verifBase`

Ici, le programme n'est pas très loquace !

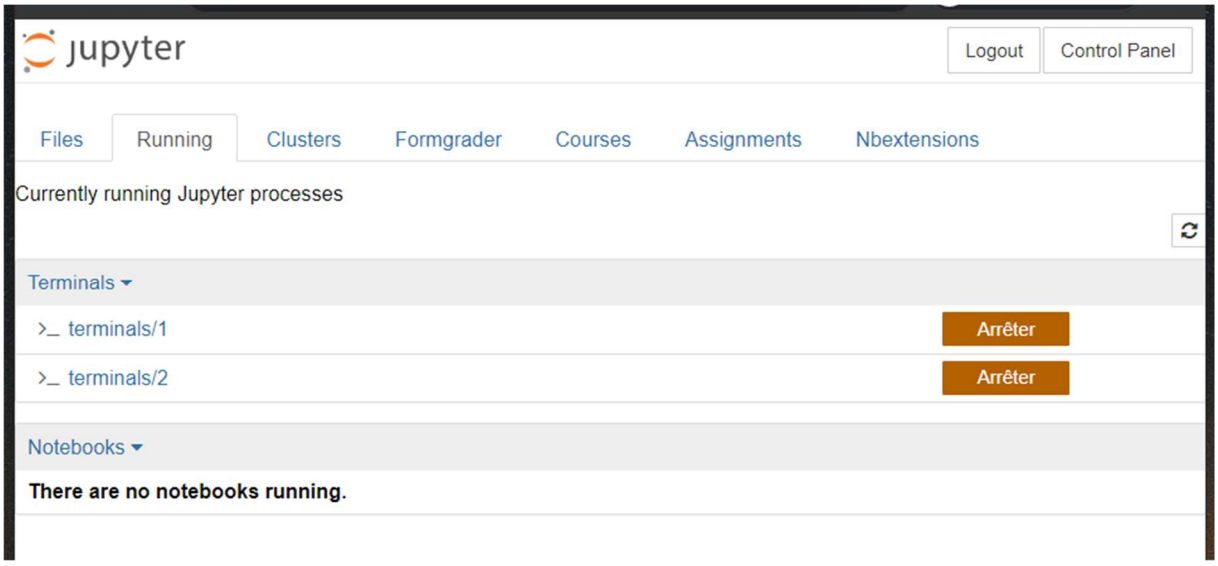
Si le programme boucle, on peut l'interrompre en tapant *Ctrl-C* (i.e. en appuyant simultanément sur les touches *Ctrl* et *C*.

Il n'y a malheureusement pas (encore) de débogueur utilisable (mais voir la rubrique sur les « notebooks » qui peuvent aider à la mise au point).

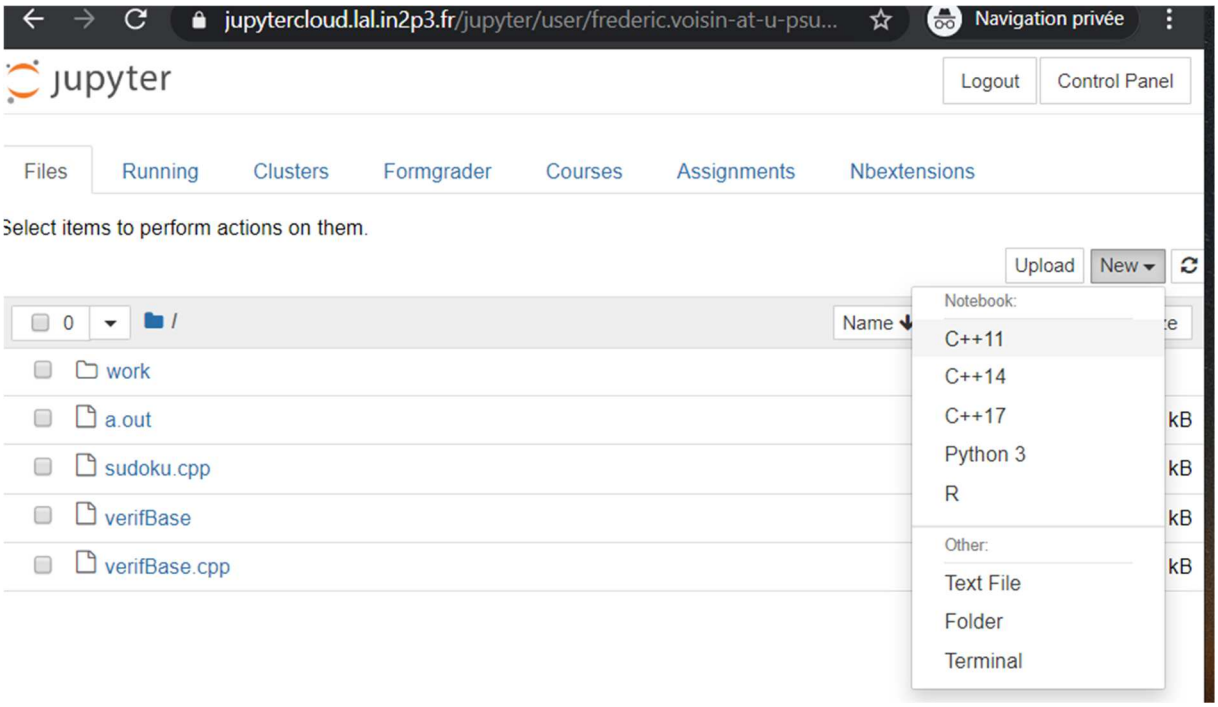
Retour à la fenêtre principale : les fichiers `a.out` et `verifBase` apparaissent maintenant dans l'espace de travail. Dans le navigateur, ici trois fenêtres sont ouvertes : l'espace personnel et deux terminaux :



Pour arrêter proprement un terminal, cliquez sur l'onglet `Running`, puis sur le bouton `Arrêter` du terminal dont vous n'avez plus l'usage (comme pour `Téléverser`, ces boutons sont en français. Jupyter est au moins bilingue, on n'en attendait pas moins de lui !)



IV - Le « notebook C++ » : Le menu New de la fenêtre principale permet de lancer un « notebook » (une « feuille de travail ») qui va être ici un interprète C++ (on choisira dans le menu New la version C++11).

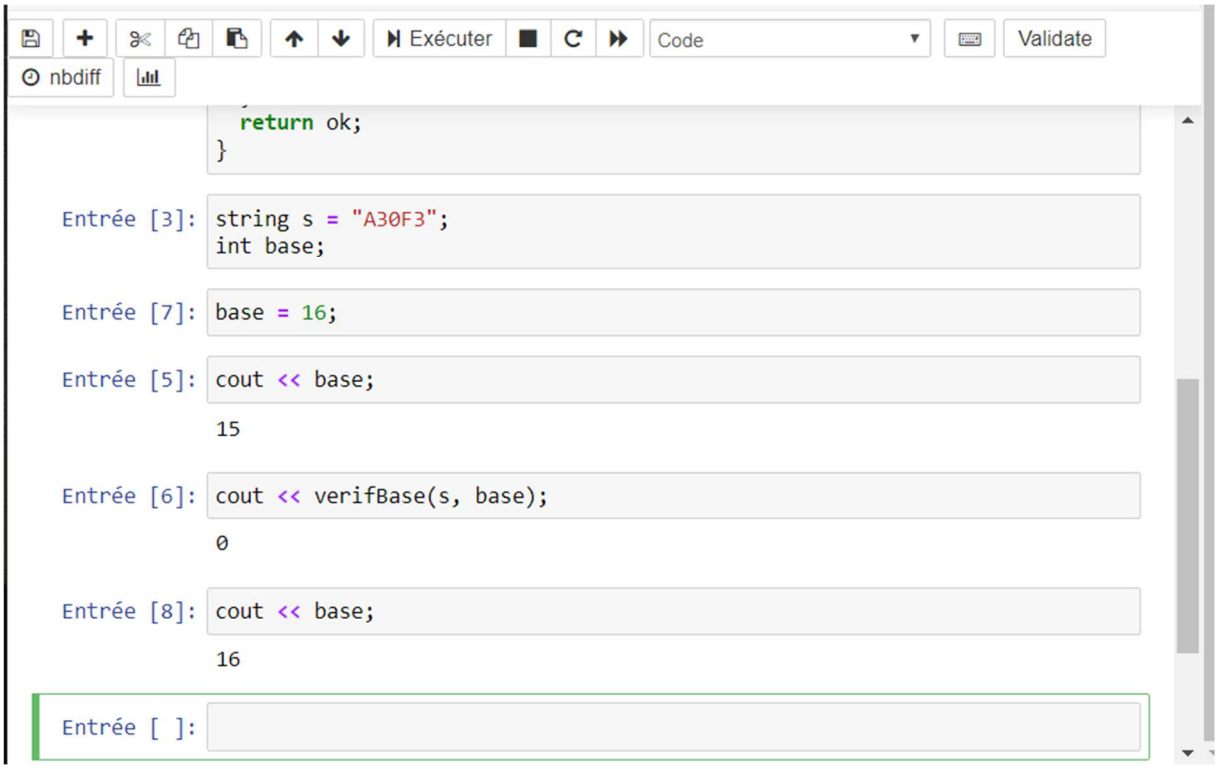


Vous pouvez avoir plusieurs « notebooks », les effacer, les reprendre pour les compléter, etc. ***Faites simple au début pour ne pas vous y perdre***. Chaque notebook s'ouvre dans une fenêtre différente et est associé à un « fichier » dans votre espace personnel. C'est juste une sorte de « fichier » très actif car associé à un processus d'exécution.

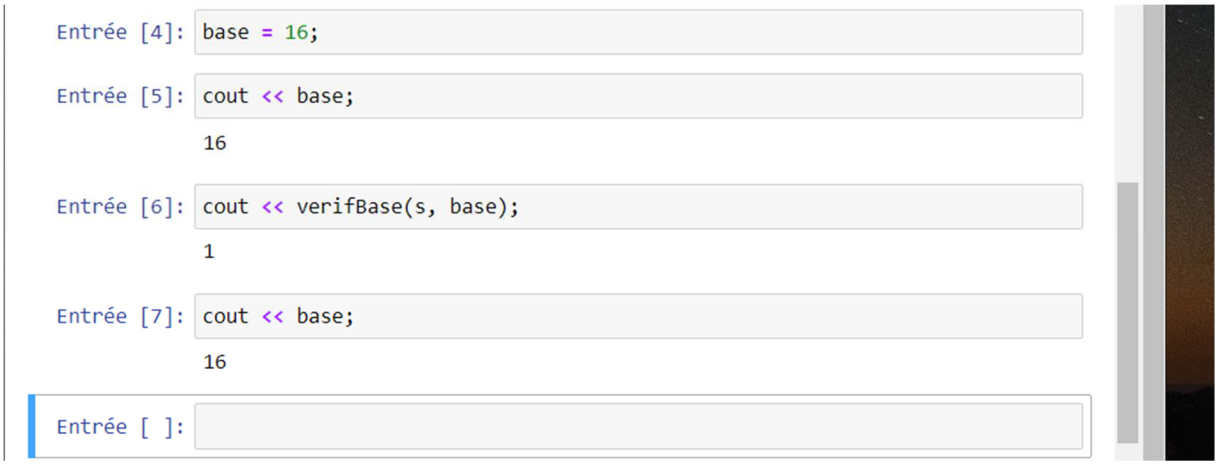
À la différence d'un compilateur qui prend en entrée un fichier (ou un ensemble de fichiers) avant de le(s) traduire en un fichier exécutable sur la machine, un interprète va vérifier, traduire et exécuter les instructions ou directives d'un programme au fur et à mesure qu'il les rencontre. Ceci peut être très utile pour mettre au point un programme, surtout pour de petits programmes : on peut tester directement l'effet de ses instructions. Si vous avez fait un peu de `python`, vous avez l'habitude. L'ordre dans lequel on exécute les instructions importe puisque chaque action change l'état de l'interprète (ajout d'une nouvelle fonction, changement de valeur d'une variable globale, etc.).

On tape ses directives dans une cellule (par exemple des `#include`, `using namespace std`, ou une déclaration de variable, le code d'une fonction, etc.) puis on valide par `Maj-Entrée` (i.e. la touche pour les majuscules plus la touche `Entrée`). Le résultat (ou l'indication d'erreur) est indiqué en-dessous. Les cellules restent affichées pour qu'on puisse bien distinguer le texte du programme et les sorties. On peut même sélectionner les cellules pour les effacer ou les déplacer (mais attention, ça ne change **pas** pour autant l'état courant de l'interprète, **on ne modifie pas le passé** : il faudra faire rejouer l'ensemble de la nouvelle séquence d'actions pour être dans un état cohérent). Le contenu d'une cellule est aussi éditable.

Si vous éditez le contenu d'une cellule qui a déjà été évaluée puis vous l'exécuter à nouveau, le résultat apparaîtra bien en dessous **mais le numéro dans la cellule correspond à l'ordre dans lequel les cellules ont été exécutées**. Dans l'exemple ci-dessous, la cellule numérotée `[4]` a été exécutée, puis la `[5]` (ce qui montre qu'à cet instant la variable `base` valait `15`) et la cellule`[6]`, avant que la cellule `[4]`soit modifiée et re-exécutée. La cellule `[4]` n'apparaît alors plus et est remplacée par la cellule `[7]` avec son résultat. Sans cette numérotation fidèle à l'ordre d'exécution, on ne comprendrait plus les résultats des cellules qui suivent et qui avaient été exécutées dans l'ordre initial.



Une commande (icône semblable au « défilement rapide d'une vidéo ») permet de rejouer les instructions dans l'ordre où elles apparaissent actuellement dans la feuille de travail ; celle-ci pourra donc maintenant afficher des résultats différents. Ceci est illustré sur la partie significative de l'exemple : la cellule `[5]` a été exécutée dans le contexte de la (nouvelle) cellule `[4]`:



On a bien une « feuille de travail » puisque interactions et résultats sont conservés sur la page qu'on peut éditer, sauvegarder, rejouer.

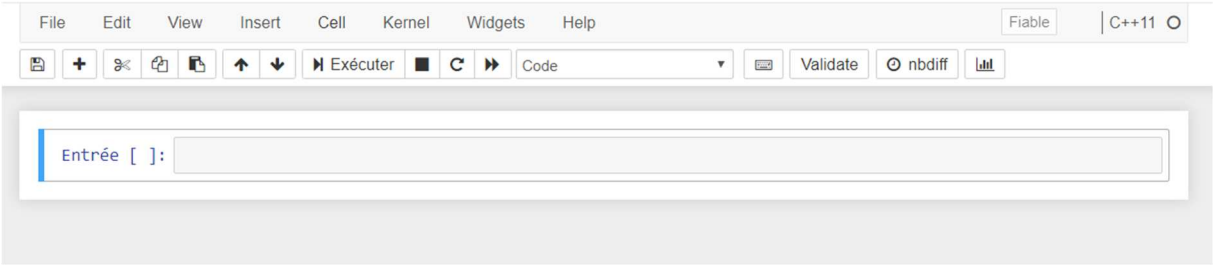
Notez que l'avant-dernière interaction de l'exemple (correspondant à la cellule `[9]` ci-dessous) ne laisse pas de trace visible. L'interprète a bien évalué la variable mais vous n'avez pas demandé d'imprimer le résultat produit. Ceci est corrigé dans la cellule suivante :

La version courante de l'interprète présente encore quelques défauts de jeunesse : notamment ne rentrez pas plusieurs définitions de fonctions dans la même cellule et pour redéfinir une fonction ou une variable globale, il faut relancer l'interprète sur toute la feuille (bouton « avance rapide »). Remettez vos instructions dans le bon ordre puis re-exécuter la feuille de travail pour être dans un état compréhensible.

Principales commandes pour le notebook :

Terminologie : ce qui est appelé « kernel » dans cette page est le processus qui contient l'interprète qui évalue vos cellules et le système de gestion des cellules et de l'affichage.

La plupart des icônes illustrent bien l'action associée et les bulles d'aide fournissent le complément d'aide.



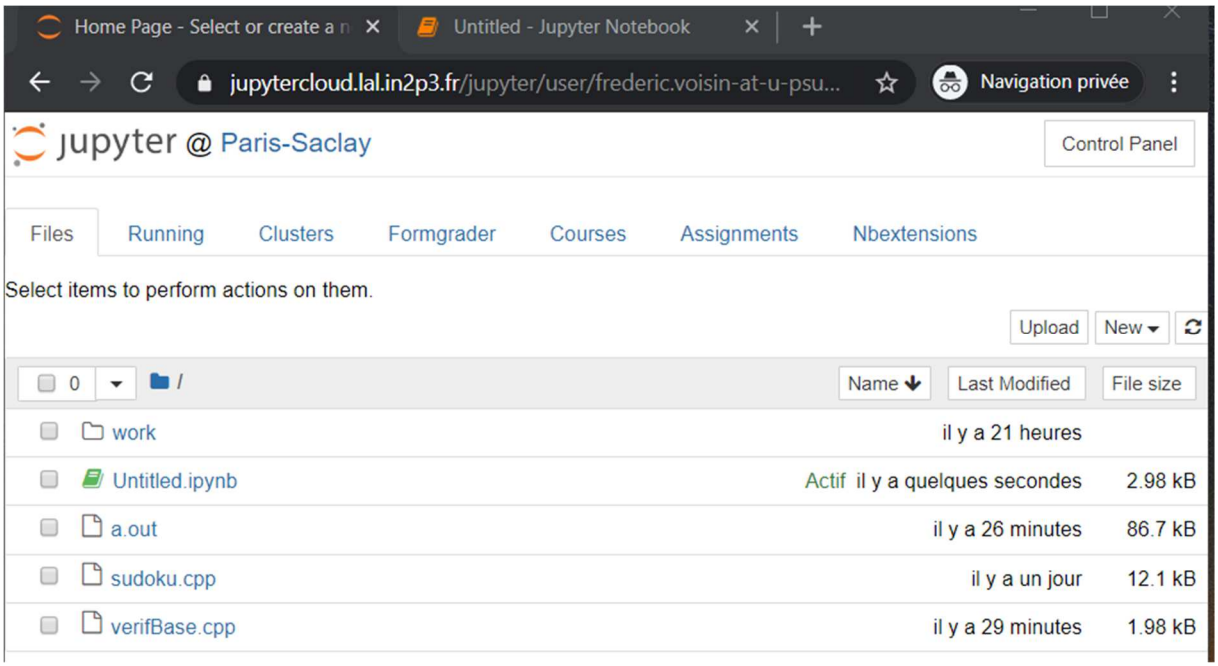
Dans l'onglet `Kernel` vous trouverez notamment les commandes pour interrompre un programme qui boucle, relancer l'interprète, fermer la feuille de travail.

Parmi les raccourcis sous forme de boutons :

- L'icône « paire de ciseaux » supprime les cellules sélectionnées
- Les icônes « flèche montante/descendante » déplacent la cellule courante dans la feuille de calcul
- L'icône « pause vidéo » (carré noir) interrompt l'exécution
- L'icône « avance rapide » relance la feuille en lui faisant évaluer, dans l'ordre actuel, le contenu des cellules.

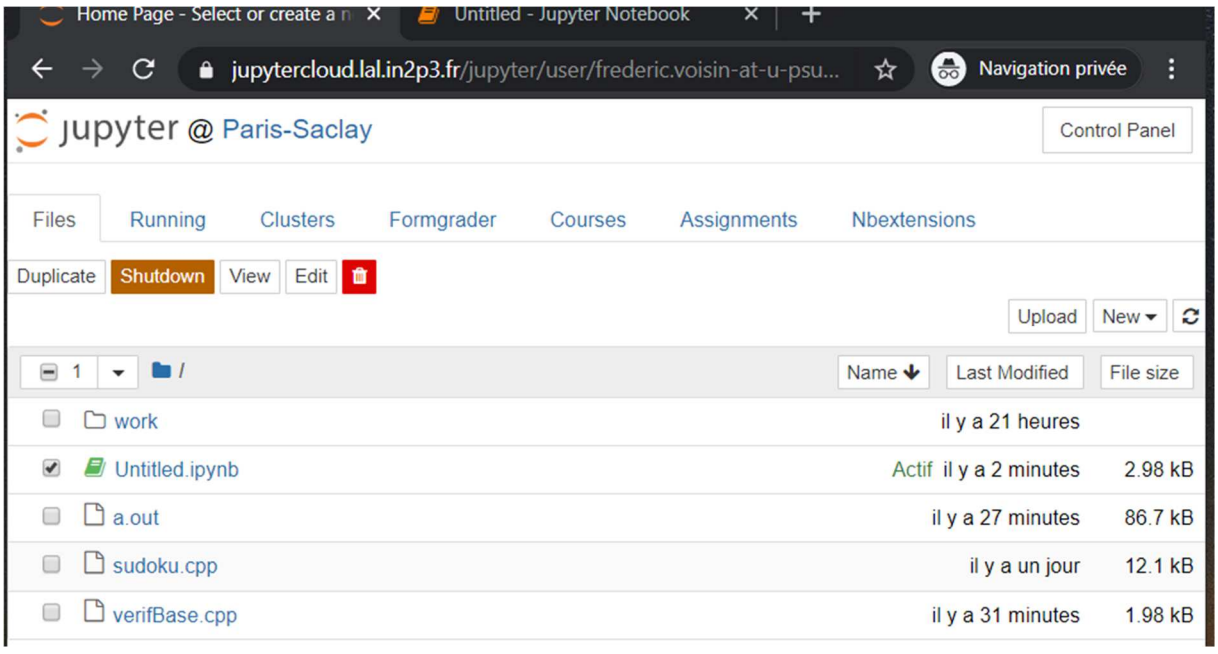
Gestion des « fenêtres Jupyter » avec les notesbooks

On revient sur la fenêtre principale Jupyter :



On y voit le fichier, ici nommé `Untitled.ipynb`, qui est associé au contenu d'un notebook. Vous pouvez avoir plusieurs tels fichiers si vous avez lancé plusieurs notebooks. Le fichier apparaît dans le répertoire dans lequel vous étiez au moment où vous avez lancé le notebook. Si vous cliquez sur un tel fichier, vous retournez dans le notebook correspond pour y travailler.

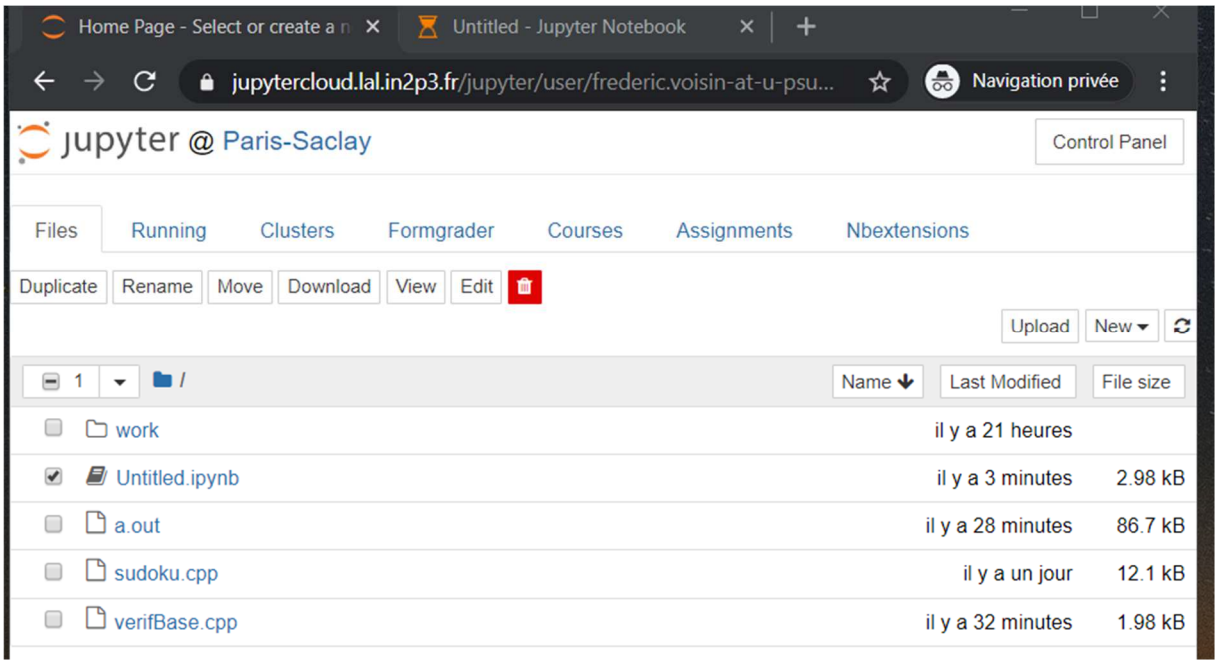
Pour arrêter ce notebook, cliquez sur la case à sa gauche puis sur le bouton `Shutdown` qui apparaît.



Vous pouvez aussi passer par l'onglet `Running` de la page principale et les boutons `Arrêter`, comme pour le terminal.

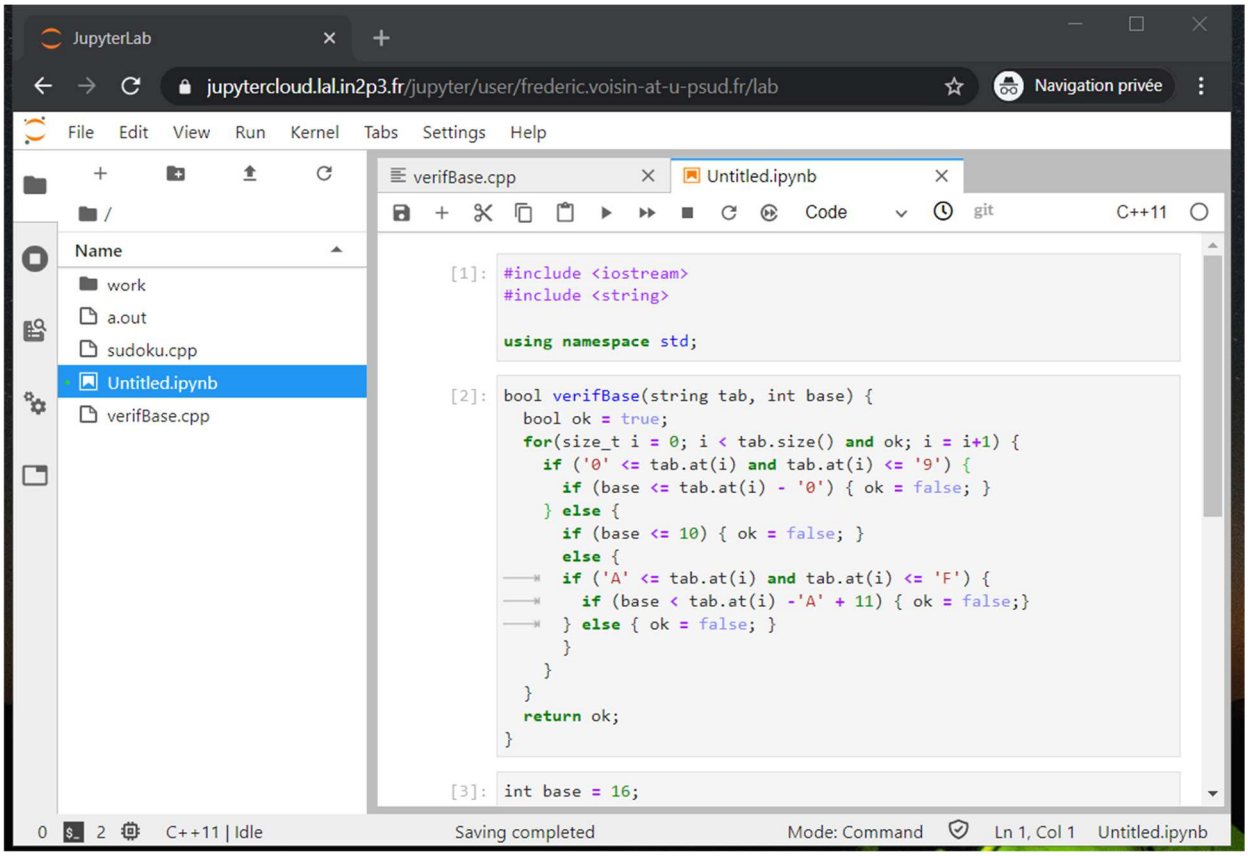
En guise de conclusion :

Renommer vos notebooks avec `Rename`, après sélection de l'élément, pour vous y retrouver. Et faites le ménage dans ce qui ne vous sert plus.



Utilisez « `Control Panel` » / « `Stop My Server` » pour quitter le système.

V – Une interface alternative : quand vous arrivez sur votre espace personnel, allez dans la barre de titre de votre fenêtre et modifiez la fin de l'URL de cette page : remplacez le mot « `tree` » par « `lab` » puis validez. Vous arrivez sur une fenêtre avec un look différent :

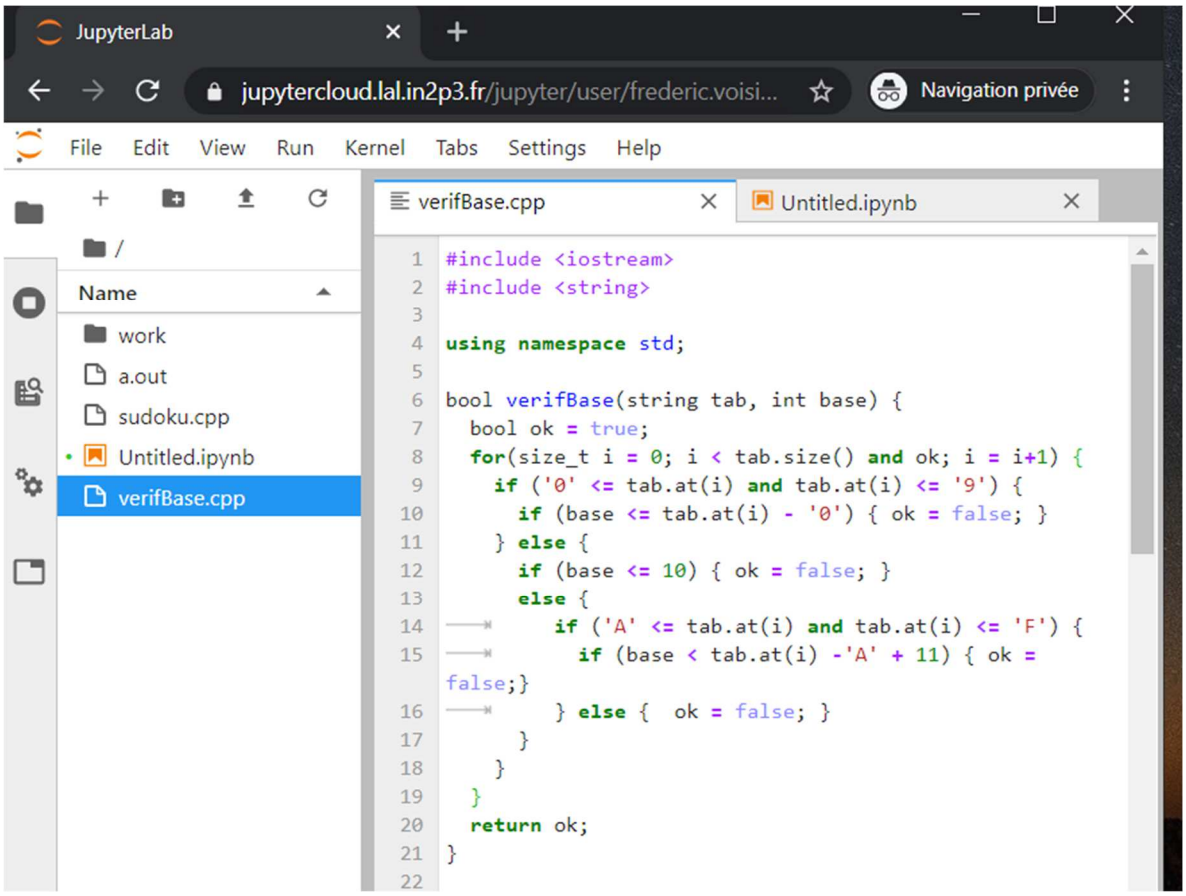


La fenêtre présente les différents outils sous forme d'onglets (« tab » dans le jargon local). La partie gauche contient une barre de boutons et l'espace personnel (qu'on peut cacher ou non grâce à un des boutons). La partie droite contient les onglets : ici un onglet d'édition pour le fichier `verifBase.cpp` et un notebook. On passe de l'un à l'autre en sélectionnant les onglets et on peut bien sûr ajouter/supprimer des onglets. C'est plus pratique d'avoir la fenêtre d'édition juste à côté du terminal de compilation ou du notebook.

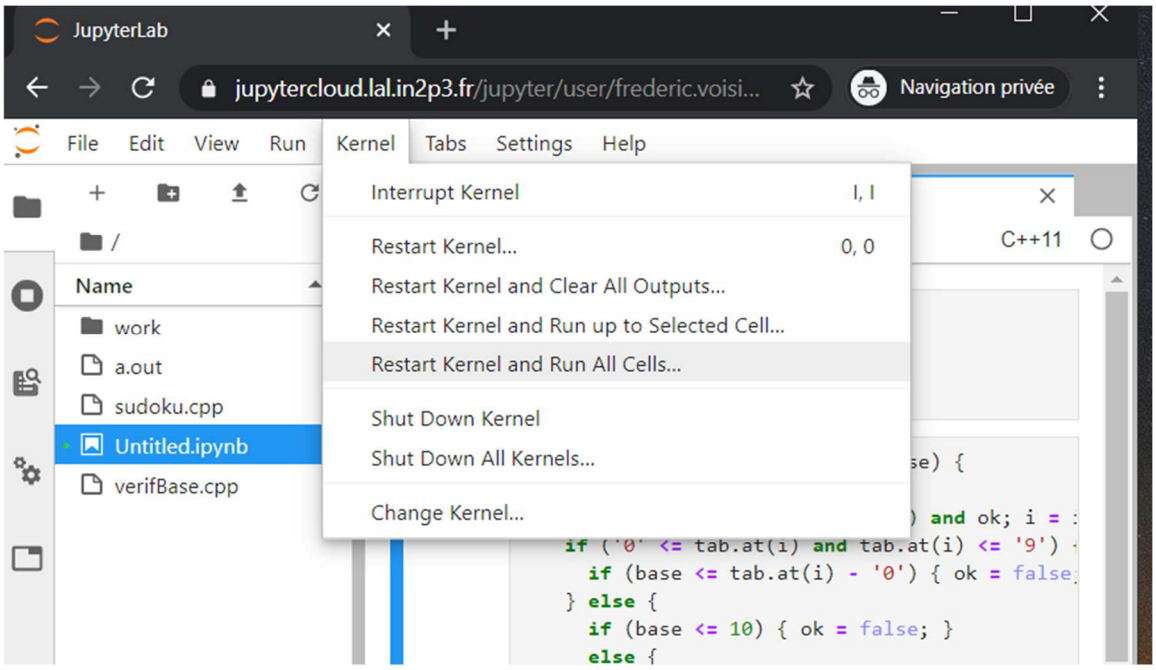
Dans l'image ci-dessus, remarquez dans la fenêtre du notebook (celle nommée `Untitled.ipynb`) la barre de boutons, dont les plus utiles :

- Icone « défilement rapide » pour exécuter toutes les cellules dans l'ordre du texte (mais **sans** redémarrer le kernel, contrairement à son fonctionnement dans l'interface classique, ce qui peut entraîner l'apparition de messages d'erreurs en cas de redéfinition de fonctions ou variables)
- Icone « rechargement » pour redémarrer le kernel mais sans exécuter les cellules
- Icone « défilement rapide dans rechargement » pour combiner les deux actions.

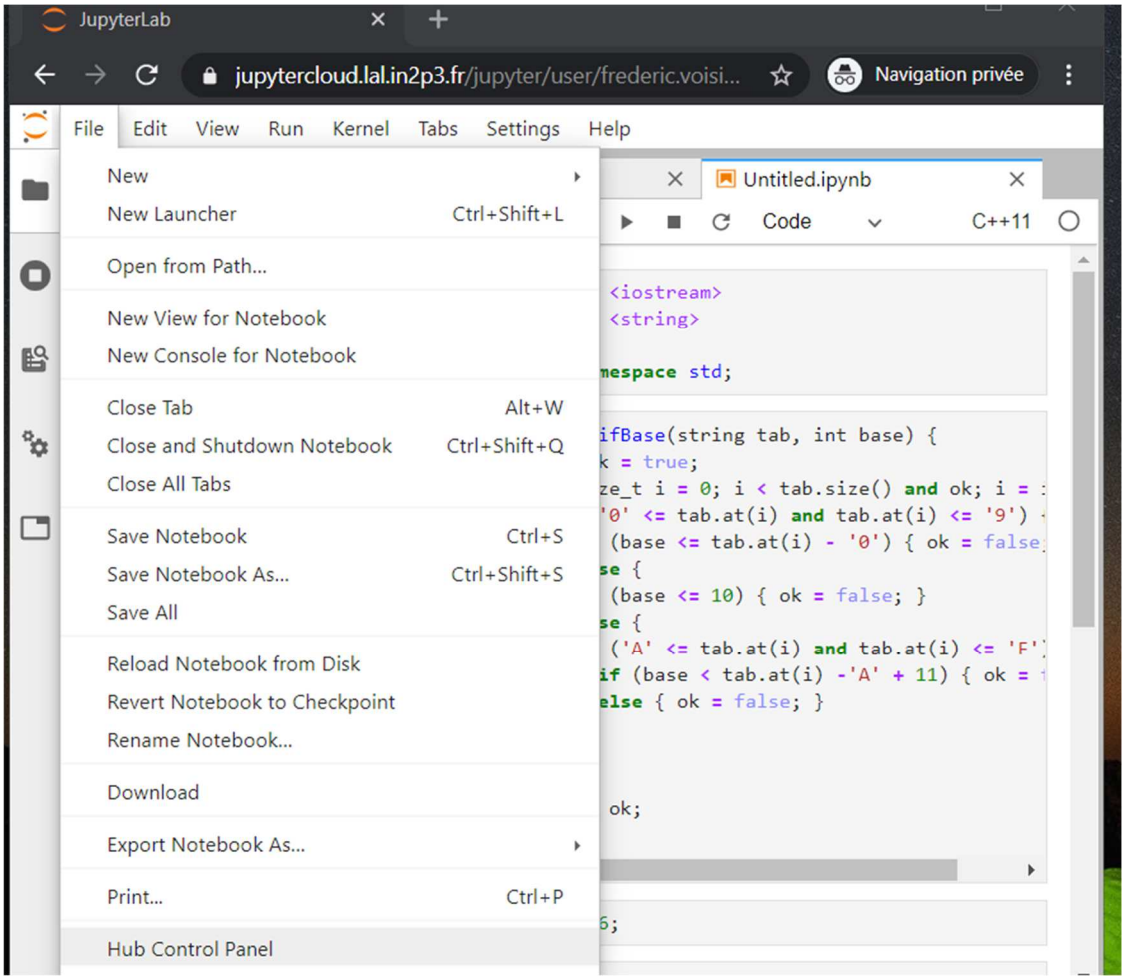
D’un clic on sélectionne par exemple l’onglet d’édition déjà ouvert :



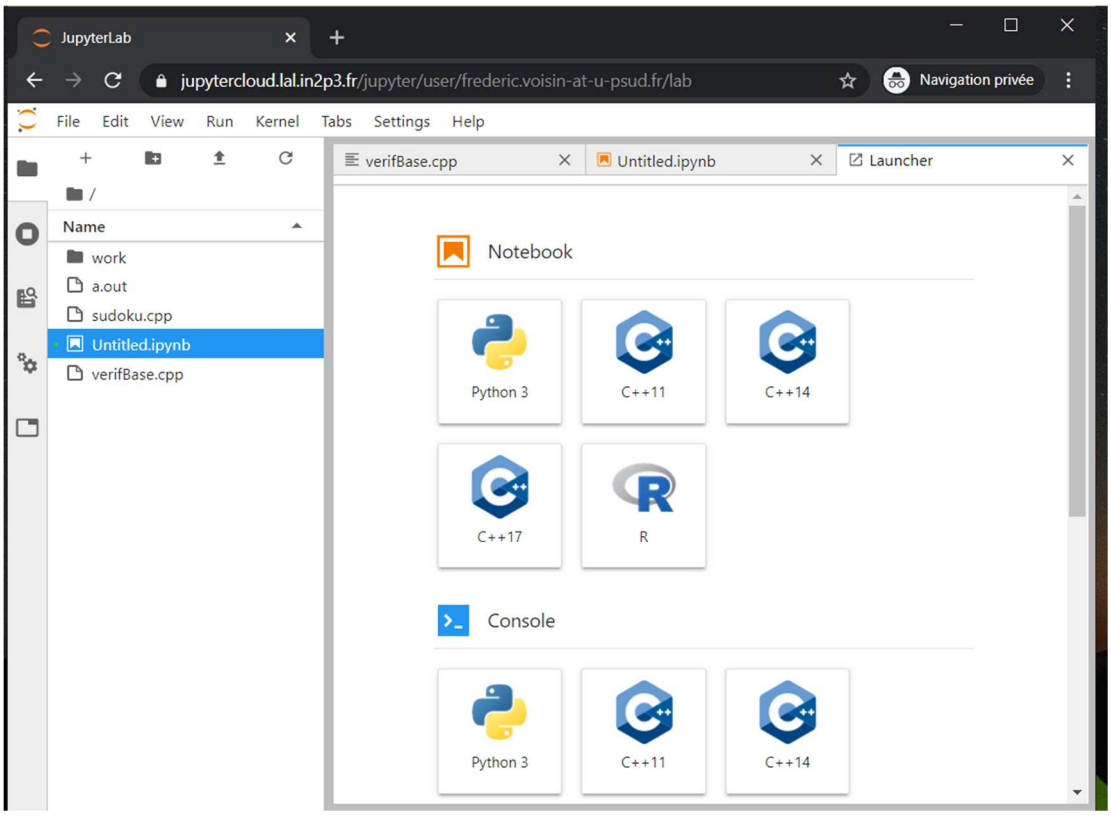
La fenêtre principale contient les menus habituels, dont `Kernel` pour gérer les kernels :

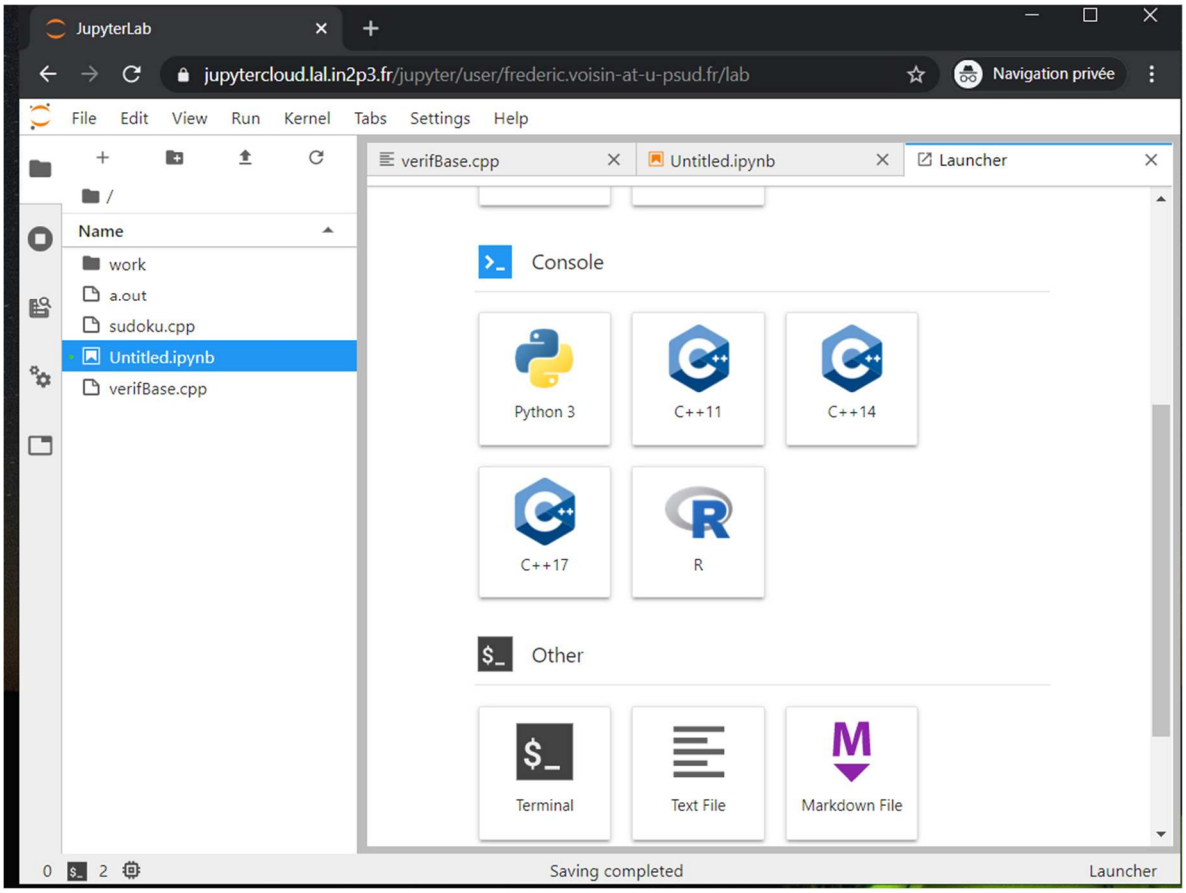


Le bouton « `Control Panel` » se trouve dans le menu `File` :



Le bouton étiqueté + (barre en haut à de gauche dans la fenêtre principale) permet de lancer un nouveau « lanceur » (« `Launcher` ») pour ouvrir un nouveau notebook ou un autre outil :





That's all, Folks, enjoy !