

Questions similaires:

[Comment puis-je augmenter la largeur des cellules du bloc-notes Jupyter / ipython dans mon navigateur?](#) 350

[Comment ajouter du noyau python3 à jupyter \(IPython\).](#) 322

[Comment mettre en commentaire plusieurs lignes dans le bloc-notes Jupyter Ipython? \[fermé\]](#) 269

[Comment convertir un bloc-notes IPython en fichier Python via la ligne de commande?](#) 258

[Utilisation de Python 2.x et Python 3.x dans IPython Notebook](#) 255

# Comment masquer le code des cellules dans un notebook ipython visualisé avec nbviewer?

147



J'ai un notebook ipython / jupyter que je visualise à l'aide de NBviewer.

Comment puis-je masquer tout le code du bloc-notes rendu par NBviewer, de sorte que seule du code (par exemple les graphiques et les tableaux) et les cellules de démarquage soient af

javascript ipython ipython-notebook



10 Il n'y a toujours pas de bouton pour cela dans l'interface utilisateur par défaut (février IMHO c'est vraiment vraiment ennuyeux. C'est sur la liste des fonctionnalités qui serc implémentées: [github.com/jupyter/notebook/issues/534](#) C'est super. J'attends cela av impatience.  
— [stochastique](#)

1 Veuillez jeter un œil ci-dessous à la réponse de Noé. Avec l'inclusion d'un TemplateExporter problème est résolu indépendamment du format de sortie. Au moment de la rédaction de ce article, la réponse de Noah remplace la réponse dure (ce qui était une bonne solution avant TemplateExporter).  
— [MichaelA](#)

## Réponses:

235




```
from IPython.display import HTML

HTML('''<script>
code_show=true;
function code_toggle() {
  if (code_show){
    $('div.input').hide();
  } else {
    $('div.input').show();
  }
  code_show = !code_show
}
$( document ).ready(code_toggle);
</script>
<form action="javascript:code_toggle()"><input type="submit" value="Click here to
toggle on/off the raw code."></form>''')
```


— [dur](#)  
[source](#)


5 A travaillé pour moi sur iPython 3.1.0 si je le mets dans la cellule de code. J'ai remplacé le <form action ... > ... </form> par du HTML simple comme The raw code for this IPython notebook is by default hidden for easier reading.To toggle on/off the raw code, click <a href="javascript:code\_toggle()">here</a>.  
— [akhmed](#)




- 

Merci pour votre réponse! Voir ma réponse si vous avez besoin que le bouton soit masqué et la possibilité de masquer ou d'afficher certains blocs de code comme Rstudio.  
— [jaycode](#)
- 3



Merci, cela fonctionne et avec «enregistrement au format HTML» aussi. Recommandez de le placer dans sa propre cellule en haut du cahier.  
— [Vivek Gani](#)
- 

si vous ajoutez l'attribut accesskey = "h" à l'élément d'entrée, vous pouvez alors faire le show hide avec alt-h (au moins en chrome)  
— [frankc](#)
- 7



Comment changeriez-vous cela pour qu'il n'affiche même pas le bouton, il cache juste le code?  
— [Harlekuin](#)

79



Ceci est désormais possible directement depuis nbconvert à partir de la version [5.2.1](#) : le contenu peut être filtré à l'aide des [options d'exclusion](#) intégrées de l' [exportateur de modèles](#) . Par exemple:

```
jupyter nbconvert --to pdf --TemplateExporter.exclude_input=True my_notebook.ipynb
```

exclura les cellules "code d'entrée", c'est-à-dire le code lui-même. [Des options similaires](#) existent pour exclure les invites, les cellules de démarque ou les sorties, ou à la fois les entrées et les sorties.


(Ces options devraient fonctionner quel que soit le format de sortie.)

— [Noé](#)


[source](#)

- 3


c'est la meilleure réponse  
— [skurp](#)

- 


Où l'export .pdf est-il enregistré par défaut?  
— [MyopicVisage](#)

- 

Même dossier que le notebook .ipynb. Utilisez l'argument '--output NotebookNoCode' pour renommer le fichier.  
— [MyopicVisage](#)

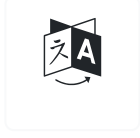
- 

Est-ce censé fonctionner dans le cahier?  
— [lcrmorin](#)

- 

@were\_cat non, il s'agit d'une commande shell utilisée pour exporter le fichier notebook .ipynb; dans cet exemple, il est converti en pdf  
— [Noah](#)

19



J'utiliserais à `hide_input_all` partir de **nbextensions** ( <https://github.com/ipython-contrib/IPython-notebook-extensions> ). Voici comment:

1. Découvrez où se trouve votre répertoire IPython:

```
from IPython.utils.path import get_ipython_dir
print get_ipython_dir()
```

2. Téléchargez **nbextensions** et déplacez-les dans le répertoire IPython.

3. Modifiez votre fichier *custom.js* quelque part dans le répertoire IPython (le mien était dans *profile\_default / static / custom* ) pour qu'il soit similaire au *custom.example.js* dans le répertoire *nbextensions* .


4. Ajoutez cette ligne à *custom.js* :

```
IPython.load_extensions('usability/hide_input_all')
```


IPython Notebook aura désormais un bouton pour basculer les cellules de code, quel que soit le classeur.

— [user394430](#)  
[source](#)

- 6




J'ai juste essayé ceci - cela semble aider à cacher les cellules de code lors de l'édition d'un bloc-notes, bien que lors de l'enregistrement du bloc-notes au format html (c'est-à-dire le rendu dans nbviewer), les cellules de code apparaissent toujours.

— [Vivek Gani](#)
- 


@VivekGani juste une note rapide que vous pouvez garder les cellules cachées cachées dans le html exporté en utilisant le modèle fourni avec le même référentiel, voir la [page de documentation correspondante](#) (voir également [cette question pertinente](#) )

— [glS](#)

15



La dernière version du notebook IPython ne permet plus d'exécuter javascript dans les cellules markdown, donc l'ajout d'une nouvelle cellule markdown avec le code javascript suivant ne fonctionnera plus pour masquer vos cellules de code (reportez-vous à [ce lien](#) )




Modifiez ~ / .ipython / profile\_default / static / custom / custom.js comme ci-dessous:


```
code_show=true;
function code_toggle() {
  if (code_show){
    $('div.input').hide();
  } else {
    $('div.input').show();
  }
  code_show = !code_show
}

$([IPython.events]).on("app_initialized.NotebookApp", function () {
  $("#view_menu").append("<li id=\"toggle_toolbar\" title=\"Show/Hide code cells\"><a href=\"javascript:code_toggle()\">Toggle Code Cells</a></li>")
});
```

— [Yangshun Tay](#)  
[source](#)

- 


exactement ce que je recherche!

— [lucky1928](#)
- 

Étrangement, cette solution n'a pas fonctionné pour moi car le menu d'affichage iPython reste inchangé. (iPython 3.1.0) Votre solution m'a inspiré à chercher plus loin et à trouver une solution très similaire par [p3trus](#) qui a ajouté un bouton au lieu d'un menu et cela a fonctionné.

— [akhmed](#)


- 1



@akhmed Vous pouvez peut-être vous référer à [stackoverflow.com/a/29851084/1914781](#) .

C'est une question de différence mais elle vous est utile!

12



J'ai écrit un code qui accomplit cela et ajoute un bouton pour basculer la visibilité du code.

Ce qui suit va dans une cellule de code en haut d'un bloc-notes:



```
from IPython.display import display
from IPython.display import HTML
import IPython.core.display as di # Example: di.display_html('<h3>%s:</h3>' % str,
raw=True)

# This line will hide code by default when the notebook is exported as HTML
di.display_html('<script>jQuery(function() {if (jQuery("body.notebook_app").length ==
0) { jQuery(".input_area").toggle(); jQuery(".prompt").toggle();}});</script>',
raw=True)


# This line will add a button to toggle visibility of code blocks, for use with the
HTML export version
di.display_html(''<button onclick="jQuery('.input_area').toggle();
jQuery('.prompt').toggle();">Toggle code</button>'', raw=True)
```

Vous pouvez voir [un exemple de ce à quoi cela ressemble dans NBviewer ici](#) .


**Mise à jour:** Cela aura un comportement amusant avec les cellules Markdown dans Jupyter, mais cela fonctionne bien dans la version d'exportation HTML du notebook.


— [Max Masnick](#)  
[source](#)

- 3

 Cela fonctionne sur les cellules de code, mais si vous avez des cellules de démarque, cela fait quelque chose d'étrange. Il montre le **démarquage en tant que démarque** , puis affiche le même contenu - mais formaté - ci-dessous.

— [Scott H](#)
-  Je viens de comprendre que la seule chose qui ne va pas est la spécification du nœud. Au lieu de '.input\_area' et '.prompt' , utilisez 'div.input' et cela fonctionne comme un charme! Donc, pour récapituler, remplacez-le jQuery("div.input").toggle(); par jQuery('.input\_area').toggle(); jQuery('.prompt').toggle(); . @Max Masnick, pourriez-vous corriger votre réponse?

— [Scott H](#)
-  Utiliser le "div.input" comme sélection de nœuds fonctionne tant que vous n'interagissez pas avec les cellules de démarquage, mais j'ai juste compris que si vous interagissez avec les cellules de démarque, vous pourriez avoir un comportement génial. Par exemple, si vous double-cliquez sur une cellule de démarque, elle est entièrement masquée. En l'état, ma modification de la solution de Max convient parfaitement pour générer du HTML à partager avec d'autres, mais pas pour interagir trop avec lui par la suite.


— [Scott H](#)
-  Ouais, alors j'ai remarqué la même chose que vous avez faite avec les cellules de Markdown qui se sont toutes redressées. Cela fonctionne bien dans l'exportation HTML, où je l'utilise. Je modifierai la réponse pour noter ceci.

— [Max Masnick](#)
- 1

 Pour supprimer l'espace de droite restant de la suppression de ".prompt", ajoutez simplement ce code à la fin du code ci-dessus. CSS = ""#notebook div.output\_subarea { max-width:100%;"" HTML('<style>{</style>' .format(CSS)) . Ceci est très utile pour l'impression.

— [Little Bobby Tables](#)

10



Cela peut être fait en utilisant un IPython `ToggleButton` widget et un peu de JavaScript. Le code suivant doit être placé dans une cellule de code en haut du document:



```
import ipywidgets as widgets
from IPython.display import display, HTML

javascript_functions = {False: "hide()", True: "show()"}
button_descriptions = {False: "Show code", True: "Hide code"}

def toggle_code(state):

    """
    Toggles the JavaScript show()/hide() function on the div.input element.
    """

    output_string = "<script>$(\"div.input\").{}/script>"
    output_args = (javascript_functions[state],)
    output = output_string.format(*output_args)

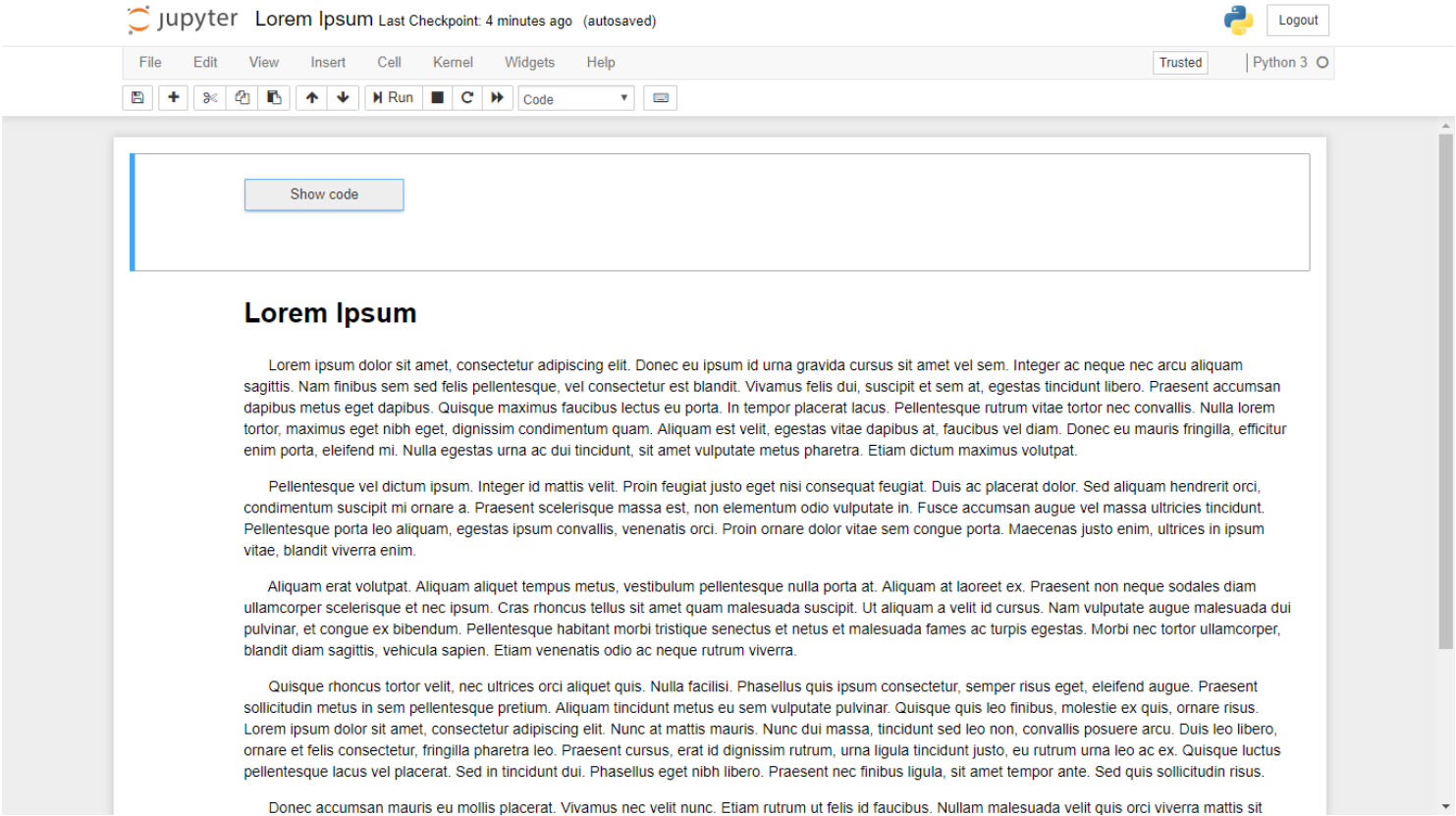
    display(HTML(output))

def button_action(value):

    """
    Calls the toggle_code function and updates the button description.
    """

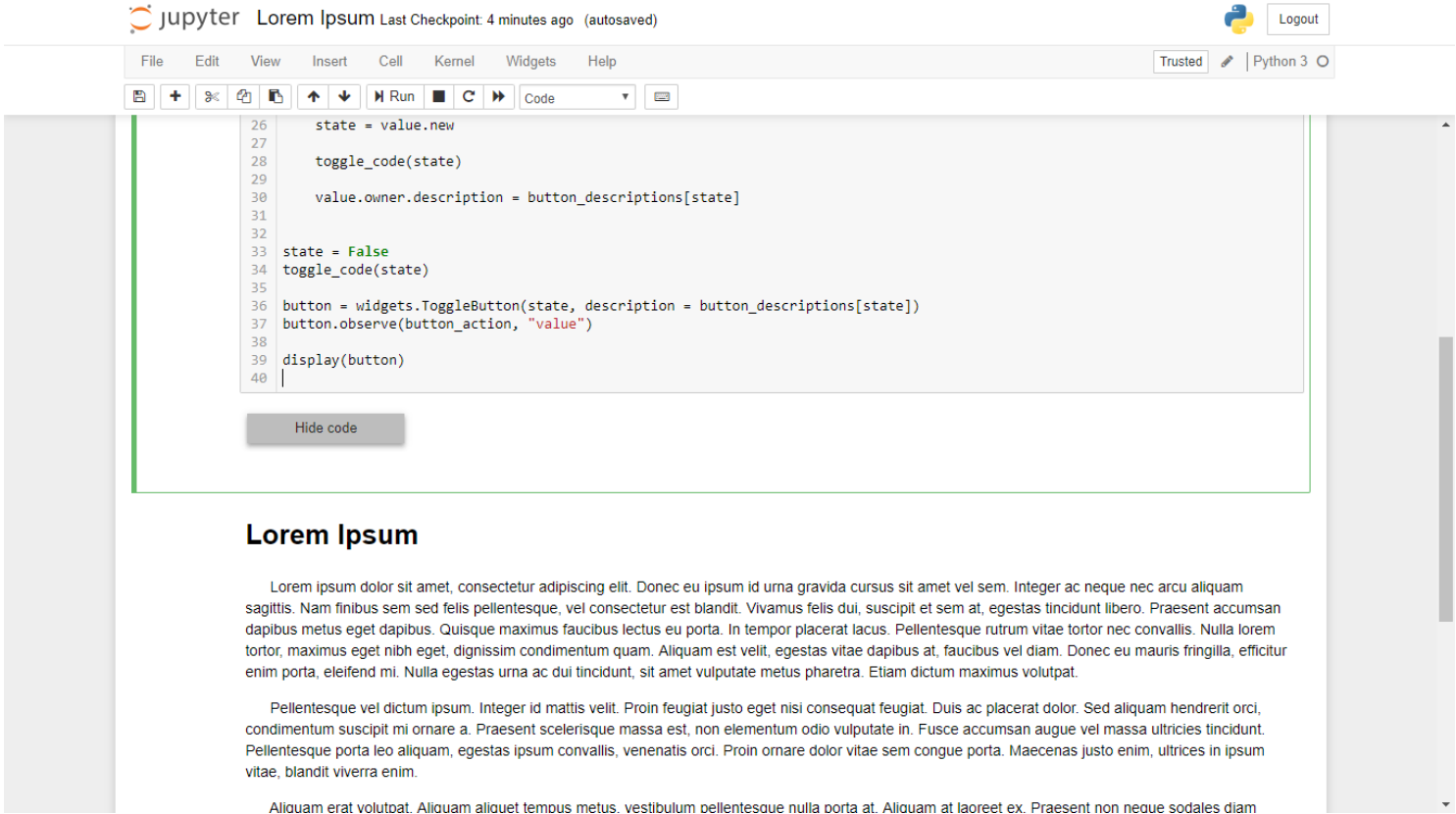
    state = value.new
```

Cela crée le bouton suivant pour afficher / masquer le code du bloc-notes Jupyter, réglé par défaut à l'état «masquer»:

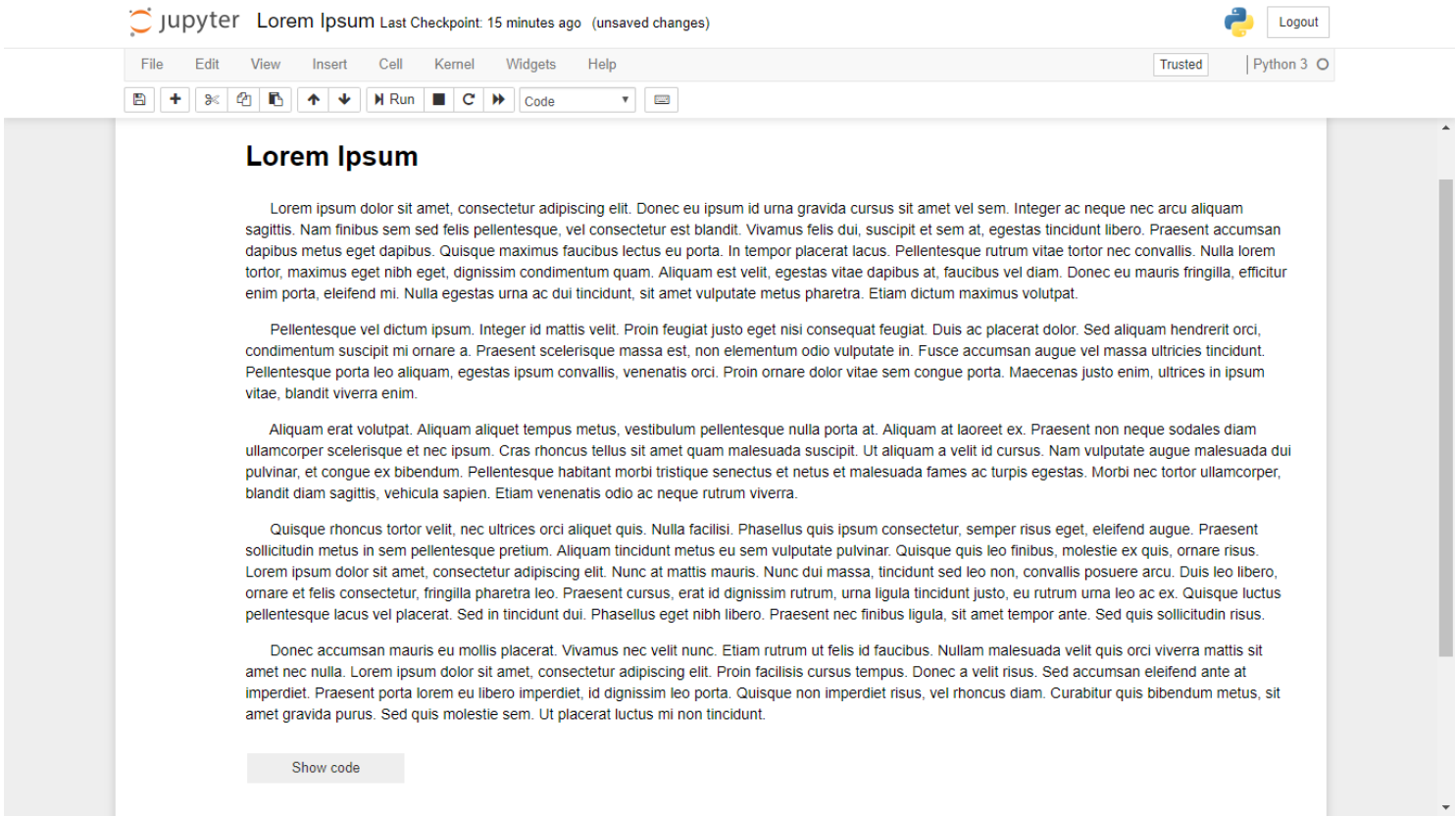


Lorsqu'il est défini sur l'état «show», vous pouvez alors voir le code du bloc-notes Jupyter:





En passant, alors qu'une grande partie de ce code doit être placée au début du notebook, l'emplacement du bouton bascule est facultatif. Personnellement, je préfère le garder au bas du document. Pour ce faire, déplacez simplement la `display(button)` ligne vers une cellule de code distincte en bas de la page:



— [Erick Shepherd](#)  
[source](#)

9

Il existe une solution intéressante fournie [ici](#) qui fonctionne bien pour les blocs-notes exportés au format HTML. Le site Web renvoie même ici à cet article SO, mais je ne vois pas la solution de Chris ici! (Chris, où es-tu?)

C'est fondamentalement la même solution que la réponse acceptée de harshil, mais elle a l'avantage de cacher le code de bascule lui-même dans le HTML exporté. J'aime aussi que cette approche évite le besoin de la fonction HTML IPython.

Pour implémenter cette solution, ajoutez le code suivant à une cellule 'Raw NBConvert' en haut de votre notebook:

```
<script>
function code_toggle() {
  if (code_shown){
    $('div.input').hide('500');
    $('#toggleButton').val('Show Code')
  } else {
    $('div.input').show('500');
    $('#toggleButton').val('Hide Code')
  }
  code_shown = !code_shown
}

$( document ).ready(function(){
  code_shown=false;
  $('div.input').hide()
});
</script>
<form action="javascript:code_toggle()">
  <input type="submit" id="toggleButton" value="Show Code">
</form>
```


Ensuite, exportez simplement le notebook au format HTML. Il y aura un bouton bascule en haut du cahier pour afficher ou masquer le code.

Chris fournit également un exemple [ici](#).

Je peux vérifier que cela fonctionne dans Jupyter 5.0.0

**Mise à jour** : Il est également pratique d'afficher / masquer les `div.prompt` éléments avec les `div.input` éléments. Cela supprime le `In [##]:` et `Out: [##]` texte et réduit les marges sur la gauche.


— [Ken](#)  
[source](#)



serait-il possible d'utiliser ce code pour masquer sélectivement les sorties en cliquant sur un bouton? IE `$( 'div.output' ).next().hide( '500' );` pour masquer la prochaine sortie? J'ai essayé moi-même mais je n'arrive pas à faire fonctionner cela.

— [Brian Keith](#)

7



Pour un meilleur affichage avec un document imprimé ou un rapport, nous devons également supprimer le bouton et la possibilité d'afficher ou de masquer certains blocs de code. Voici ce que j'utilise (copiez-collez simplement ceci dans votre première cellule):



```
# This is a cell to hide code snippets from displaying
# This must be at first cell!

from IPython.display import HTML

hide_me = ''
HTML(''
```

Puis dans vos prochaines cellules:

```
hide_me
print "this code will be hidden"
```

et

```
print "this code will be shown"
```

— [Jaycode](#)  
[source](#)


- 


Je suppose que cela ne fonctionne pas pour les versions les plus récentes / python 3?  
— [baxx du](#)
- 

Fonctionne avec jupyter version 4.3.0 avec Python version 3.6.1.  
— [Alma Rahat](#)
- 

Merci! Heureux de dire que cela fonctionne également avec le notebook Jupyter **5.3.1** . J'utilise Python version **3.6.1**  
— [Amitrajit Bose](#)

4






Cela rendra une sortie de notebook IPython. Cependant, vous noterez être en mesure de voir le code d'entrée. Vous pouvez copier un bloc-notes, puis ajouter ce code si nécessaire pour le partager avec quelqu'un qui n'a pas besoin d'afficher le code.

```
from IPython.display import HTML

HTML(''
```

— [Chase Wright](#)  
[source](#)

- 1
- 

@Rocketq use this - from IPython.display import HTML HTML(''

— [fixxxer](#)



4

Convertir la cellule en Markdown et utiliser HTML5 `<details>` balise comme dans l'exemple en joyrexus :

<https://gist.github.com/joyrexus/16041f2426450e73f5df9391f7f7ae5f>



```
## collapsible markdown?

<details><summary>CLICK ME</summary>
<p>

#### yes, even hidden code blocks!

```python
print("hello world!")
```

</p>
</details>
```

— [Valentas](#)  
[source](#)

1

Voici une autre solution proposée par [p3trus](#) :



```
$([IPython.events]).on('notebook_loaded.Notebook', function(){
    IPython.toolbar.add_buttons_group([
        {
            'label'   : 'toggle input cells',
            'icon'    : 'icon-refresh',
            'callback': function(){$('.input').slideToggle()}
        }
    ]);
});
```

Comme décrit par [p3trus](#) : "[Il] ajoute un bouton à la barre d'outils du notebook ipython pour masquer / afficher la cellule de code d'entrée. Pour l'utiliser, vous devez placer le fichier custom.js dans votre .ipython\_profile/static/custom/ dossier, où est le profil ipython utilisé. "

Mes propres commentaires: j'ai vérifié cette solution et elle fonctionne avec iPython 3.1.0.

— [Akhmed](#)  
[source](#)

1

La solution acceptée fonctionne également dans julia Jupyter / IJulia avec les modifications suivantes:




```
display("text/html", """<script>
code_show=true;
function code_toggle() {
  if (code_show){
    $("div.input").hide();
  } else {
    $("div.input").show();
  }
  code_show = !code_show
}
$( document ).ready(code_toggle);
</script>
<form action="javascript:code_toggle()"><input type="submit" value="Click here to
toggle on/off the raw code."></form>""")
```

notez en particulier:


- utiliser la `display` fonction
- échapper au `$` signe (sinon vu comme une variable)


— [gozzilli](#)  
[source](#)



Je ne sais pas comment faire fonctionner cela. Une déclaration d'importation est-elle nécessaire et quel type de boîte doit être le bloc? Un brut ou une boîte de code?  
— [J Spen](#)

1





[Voici](#) un bel article (le même que celui que @Ken a publié) sur la façon de peaufiner les blocs-notes Jupyter (le nouvel IPython) pour les présenter. Il existe d'innombrables façons d'étendre Jupyter à l'aide de JS, HTML et CSS, y compris la possibilité de communiquer avec le noyau python du notebook à partir de javascript. Il existe des décorateurs magiques pour `%%HTML` et `%%javascript` vous pouvez donc faire quelque chose comme ça dans une cellule seule:


```
%%HTML
<script>
  function code_toggle() {
    if (code_shown){
      $('div.input').hide('500');
      $('#toggleButton').val('Show Code')
    } else {
      $('div.input').show('500');
      $('#toggleButton').val('Hide Code')
    }
    code_shown = !code_shown
  }


  $( document ).ready(function(){
    code_shown=false;
    $('div.input').hide()
  });
</script>
<form action="javascript:code_toggle()"><input type="submit" id="toggleButton"
value="Show Code"></form>
```

Je peux également garantir que les méthodes de Chris fonctionnent dans jupyter 4.XX

— [ThisGuyCantEven](#)  
[source](#)

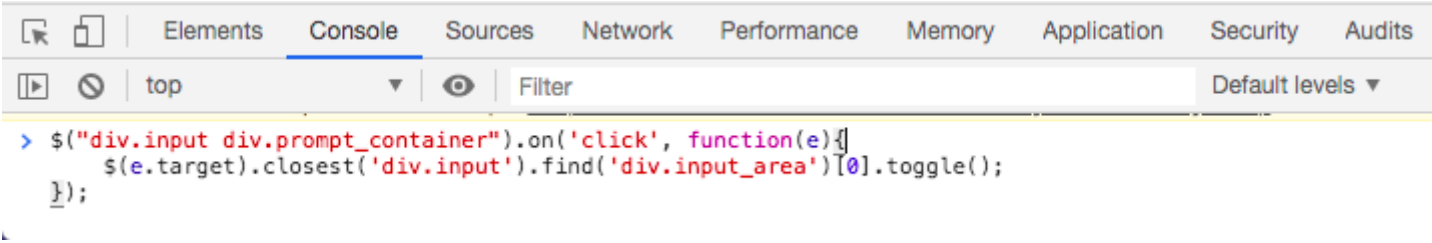
1





Solution très simple en utilisant la console du navigateur. Vous copiez ceci dans la console de votre navigateur et appuyez sur Entrée:

```
$("#div.input div.prompt_container").on('click', function(e){
  $($ (e.target).closest('div.input').find('div.input_area')[0]).toggle();
});
```





Ensuite, vous basculez le code de la cellule en cliquant simplement sur le numéro de la cellule d'entrée.

In [1]:

— [Matěj M](#)  
[source](#)

0





**(Papier) Impression ou enregistrement au format HTML**

Pour ceux d'entre vous qui souhaitent imprimer sur papier les résultats, les réponses ci-dessus ne semblent pas à elles seules donner une belle sortie finale. Cependant, prendre le code de @Max Masnick et ajouter ce qui suit permet de l'imprimer sur une page A4 complète.

```
from IPython.display import display
from IPython.display import HTML
import IPython.core.display as di

di.display_html('<script>jQuery(function() {if (jQuery("body.notebook_app").length == 0) { jQuery(".input_area").toggle(); jQuery(".prompt").toggle();}});</script>',
raw=True)

CSS = """"#notebook div.output_subarea {max-width:100%;}"""" #changes output_subarea
width to 100% (from 100% - 14ex)
HTML('<style>{}</style>'.format(CSS))
```

La raison du retrait est que la section d'invite supprimée par Max Masnick signifie que tout se déplace vers la gauche en sortie. Cela n'a cependant rien fait pour la largeur maximale de la sortie qui était limitée à `max-width:100%-14ex;` . Cela change la largeur maximale de `output_subarea` en `max-width:100%;` .

— [Petites tables Bobby](#)  
[source](#)

- 0
- Avec toutes les solutions ci-dessus, même si vous cachez le code, vous obtiendrez toujours le `[<matplotlib.lines.Line2D at 0x128514278>]` merde au-dessus de votre chiffre, ce que vous ne voulez probablement pas.
- Si vous voulez réellement vous débarrasser de l'entrée plutôt que de simplement la cacher, je pense que la solution la plus propre est de sauvegarder vos chiffres sur le disque dans des cellules cachées, puis d'inclure simplement les images dans les cellules Markdown en utilisant par exemple `![Caption](figure1.png)` .

— [maxymoo](#)  
[source](#)

- 3

Vous pouvez mettre `_ = plt.plot()` pour ne pas l'avoir impression de `[<>]` merde

— [jonnybazookatone](#)
- 3

Placer un point-virgule après les commandes de traçage matplotlib a supprimé la sortie indésirable pour moi.

— [DakotaD](#)

0

jupyter nbconvert testing.ipynb --to html --no-input

— [gocode](#)  
[source](#)



0

jupyter nbconvert yourNotebook.ipynb --no-input --no-prompt

— [Naveen Kumar](#)  
[source](#)



- 6

Veuillez ajouter quelques explications au lieu de simplement répondre avec une commande.

— [Fabian Bettag le](#)