



# Liste des annexes

ANNEXE 1. BUILDER: Paramètres des stimuli.....	3
ANNEXE 2. BUILDER: Paramètres des réponses.....	8
ANNEXE 3. Paramètres de durée/d'onset des composants.....	15
ANNEXE 4. Espace des couleurs.....	16
ANNEXE 5. Unités de mesure (fenêtre et stimuli).....	20
ANNEXE 6. Coordonnées à l'écran.....	21
ANNEXE 7. Claviers AZERTY et QWERTY.....	23
ANNEXE 8. Erreurs fréquentes.....	24
ANNEXE 9. Librairie PANDAS.....	26

```
def f(x):  
    return x+5  
  
def dotwrite(ast):  
    nodename = getNodeName()  
    label=symbol.sym_name.get(int(ast[0]),ast[0])  
    print "%s (%s)" % (nodename, label),  
    if isinstance(ast[1], str):  
        if ast[1].strip():  
            print "%s" % ast[1]  
        else:  
            print ""  
    else:  
        print ""  
        children = []  
        for n, child in enumerate(ast[1:]):  
            children.append(dotwrite(child))  
        print "%s" % nodename,  
        for name in children:  
            print "%s" % name,
```

# ANNEXE 1. BUILDER: Paramètres des stimuli

Dans cette annexe, uniquement les stimuli les plus communs sont détaillés. Pour les autres, tous les détails sont fournis sur le site de conception de PsychoPy dans le [manuel d'utilisation](#).

## 1. Texte ('text')

Cette composante peut être utilisée pour présenter du texte à l'écran, pour des stimuli ou des instructions.

### . Onglet Basic

- *Name*: Nom de l'événement. Ce nom doit être unique, et ne peut contenir que des lettres, des chiffres ou des underscores (pas de marques de ponctuation ou d'espace).
- *Start*: Le temps auquel le stimulus apparaît pour la première fois (voir Annexe 3 pour les détails).
- *Stop*: La durée pendant laquelle le stimulus est présenté (voir Annexe 3 pour les détails).
- *Color*: Couleur du stimulus (voir Annexe 4 pour la définition des couleurs).
- *Font*: Police d'écriture pour le texte.
- *Letter height*: Taille de la lettre (voir Annexe 5 pour l'unité de mesure)
- *Position*: Position du stimulus (voir Annexe 6)
- *Text*: Le texte à afficher.

### . Onglet Advanced

- *Color space*: Espace de référencement des couleurs choisi (voir Annexe 4)
- *Flip*: Pour écrire le texte en miroir (miroir horizontal ou vertical).
- *Opacity*: Transparence du texte. Le chiffre est à varier entre 1 (non transparent) et 0 (complètement transparent).
- *Orientation*: Orientation du texte (en degré).
- *Units*: Espace de mesure pour les tailles des stimuli (voir Annexe 5).

## 2. Image

Cette composante peut être utilisée pour présenter des images à l'écran. Un grand nombre de formats est supporté (tif, jpg, bmp, png, ...). Il est généralement conseillé d'avoir une image de bonne qualité et de dimensions très proches du format dans lequel l'image sera présentée sur l'écran (en pixels) pour éviter de faire travailler la carte graphique inutilement.

### . Onglet Basic

- *Name*: Nom de l'événement. Ce nom doit être unique, et ne peut contenir que des lettres, des chiffres ou des underscores (pas de marques de ponctuation ou d'espace).
- *Start*: Le temps auquel le stimulus apparaît pour la première fois (voir Annexe 3 pour les détails).
- *Stop*: La durée pendant laquelle le stimulus est présenté (voir Annexe 3 pour les détails).
- *Image*: Image à afficher. Si l'image est dans le même dossier que le script, il suffit de taper le nom du fichier image avec son extension (e.g., image1.jpg), sinon il faut aussi taper le chemin (e.g., “/Users/Fab/Desktop/image1.jpg” sur un mac)
- *Position*: Position de l'image par rapport à l'écran (voir Annexe 6).
- *Size*: Taille de l'image (voir Annexe 5 pour l'unité de mesure). A laisser vide si l'on veut que l'image apparaisse dans ses dimensions réelles.

- *Orientation*: Orientation du texte (en degré).
- *Opacity*: Transparence de l'image. Le chiffre est à varier entre 1 (non transparent) et 0 (complètement transparent).
- *Units*: Espace de mesure pour les tailles des stimuli (voir Annexe 5)

### . Onglet Advanced

- *Color*: Couleur du stimulus (voir Annexe 4 pour la définition des couleurs).
- *Color space*: Espace de référencement des couleurs choisi (voir Annexe 4)
- *Flip*: Pour présenter l'image en miroir (miroir horizontal ou vertical).
- *Interpolate*: Spécifie comment l'image doit être interpolée en cas de changement de dimensions. Si 'linear' est sélectionné, une interpolation linéaire sera appliquée si l'image doit être redimensionnée en fonction de la taille de l'écran. 'Nearest' sera une approximation.
- *Mask*: Possibilité d'ajouter un masque sur l'image. Rentrer un nom courant de masque (gauss, circle,...) ou un nom de fichier contenant ses caractéristiques.
- *Texture resolution*: Cela est nécessaire uniquement si une texture synthétique est utilisée (par ex., grating sinusoïdal).

### 3. Vidéo ('movie')

Le composant 'movie' permet de faire jouer des fichiers films de différents formats possibles (par ex.: .mpeg, .avi, .mov). La position du film peut être déterminée, il peut être pivoté, mis à l'envers et entendu selon n'importe quelle taille.

**Attention, si le fichier est trop lourd (= vidéo longue), PsychoPy pourrait planter. Par ailleurs, nous avons remarqué que le risque de plantage diminue si l'extension est .mp4 (plutôt que .mov par exemple).**

Enfin, sur les ordinateurs Mac, si Avbin n'a pas été installé, un message d'erreur apparaîtra (*IOError: Caught exception 'AVbin is required to decode compressed media' while loading file 'movie.mov'. It seems that avbin was not installed correctly. Please fetch/install it from http://code.google.com/p/avbin/.*). Il vous faut donc l'installer en suivant pour cela la procédure indiquée dans le eBook du chapitre 1.

#### . Onglet Basic

- *Name*: Nom de l'événement. Ce nom doit être unique, et ne peut contenir que des lettres, des chiffres ou des underscores (pas de marques de ponctuation ou d'espace).
- *Start*: Le temps auquel le stimulus apparaît pour la première fois (voir Annexe 3 pour les détails).
- *Stop*: La durée pendant laquelle le stimulus est présenté (voir Annexe 3 pour les détails). Si une valeur inférieure à la durée de la vidéo est insérée, celle-ci sera coupée.
- *Force end of the routine*: Si coché, signifie que la fin du film engendre la fin de la routine.

- *Backend*: Précise la librairie à utiliser pour la lecture des films (laisser celui indiqué par défaut).
- *Movie file*: Vidéo à afficher. Si la vidéo est dans le même dossier que le script, il suffit de taper le nom du fichier vidéo avec son extension (e.g., video1.mpeg), sinon il faut aussi taper le chemin (e.g., "/Users/Fab/Desktop/video1.mpeg" sur un mac)
- *Opacity*: Transparence de la vidéo. Le chiffre est à varier entre 1 (non transparent) et 0 (complètement transparent).
- *Orientation*: Orientation de la vidéo (en degré).
- *Position*: Position de la vidéo par rapport à l'écran (voir Annexe 6).
- *Size*: Taille de la vidéo (voir Annexe 5 pour l'unité de mesure). A laisser vide si l'on veut que la vidéo apparaisse dans ses dimensions réelles.
- *Units*: Espace de mesure pour les tailles des stimuli (voir Annexe 5)

## 4. Son ('sound')

Le composant 'sound' permet de jouer des sons, qui soit proviennent de fichiers sons (formats .wav, .mid, .ogg), soit sont directement générés par PsychoPy.

### . Onglet Basic

- *Name*: Nom de l'événement. Ce nom doit être unique, et ne peut contenir que des lettres, des chiffres ou des underscores (pas de marques de ponctuation ou d'espace).
- *Start*: Le temps auquel le stimulus apparait pour la première fois (voir Annexe 3 pour les détails).
- *Stop*: La durée pendant lequel le stimulus est présenté (voir Annexe 3 pour les détails). Si le champ est laissé vide, le fichier son sera joué en entier. Vous pouvez dans ce cas remplir le champ 'expected duration' (en secs), simplement pour que ce soit visualisable dans la description de la routine. Ce chiffre peut être approximatif, puisque dans tous les cas, PsychoPy jouera le son en entier.
- *Sound*: Son à jouer. Cela peut correspondre à une note (par ex. 1, Bf,... dans le [système anglais](#)), à un son continu d'une certaine fréquence en hertz (par ex., 440), ou à un fichier audio. Si le fichier audio est dans le même dossier que le script, il suffit de taper le nom du fichier avec son extension (e.g., sound1.wav), sinon il faut aussi taper le chemin (e.g., "/Users/Fab/Desktop/sound1.wav" sur un mac).
- *Volume*: Chiffre compris entre 0 et 1 (éventuellement décimal) indiquant le volume auquel le son doit être joué avec 0 comme minimum et 1 comme maximum.

## 5. Forme géométrique ('polygon')

Le composant 'Polygon' permet de présenter une large panoplie de formes géométriques.

### . Onglet 'Basic'

- *Name*: Nom de l'événement. Ce nom doit être unique, et ne peut contenir que des lettres, des chiffres ou des underscores (pas de marques de ponctuation ou d'espace).
- *Start*: Le temps auquel le stimulus apparaît pour la première fois (voir Annexe 3 pour les détails).
- *Stop*: La durée pendant laquelle le stimulus est présenté (voir Annexe 3 pour les détails).
- *Num. vertices*: Nombre entier indiquant le nombre de vertex. Le contrôle de base de la forme se fait en spécifiant le nombre de vertex (*vertices*, en anglais), un vertex étant le point où 2 segments ou plus se rencontrent. Dans PsychoPy, 2 vertex donnent une ligne, 3 donnent un triangle, 4 donnent un rectangle, etc... Un grand nombre de vertex (par ex., 100) approxime un cercle ou une ellipse.
- *Line width*: Taille de la ligne ou du contour de la forme. Nombre entier correspondant à une mesure en pixel.
- *Opacity*: Transparence de la forme. Le chiffre est à varier entre 1 (non transparent) et 0 (complètement transparent).
- *Orientation*: Orientation de la forme (en degré).
- *Position*: Position de la forme par rapport à l'écran. Les coordonnées à indiquer correspondent aux coordonnées

du centre du polygone (voir Annexe 6).

- *Size*: Ce paramètre prend 2 valeurs pour indiquer la taille de la forme. Pour une ligne, seul le premier est pris en compte (utilisez *ori* pour spécifier l'orientation). Pour des triangles et rectangles, *size* spécifie la hauteur et la largeur attendue. Pour les formes ayant 5 côtés ou plus, *size* détermine la hauteur/largeur de l'ellipse sur laquelle les vertex vont tomber plutôt que la hauteur/largeur des vertex eux-mêmes (typiquement légèrement plus petite).
- *Units*: Espace de mesure pour les tailles des stimuli (voir Annexe 5)

### . Onglet 'Advanced'

- *Fill color*: Couleur de remplissage de la forme (voir Annexe 4 pour la définition des couleurs).
- *Fill color space*: Espace de référencement des couleurs choisi pour la couleur de remplissage (voir Annexe 4)
- *Interpolate*: Spécifie comment la forme doit être interpolée en cas de changement de dimensions. Si 'linear' est sélectionné, une interpolation linéaire sera appliquée si la forme doit être redimensionnée en fonction de la taille de l'écran. 'Nearest' sera une approximation.
- *Line color*: Couleur de la ligne ou du contour de la forme (voir Annexe 4 pour la définition des couleurs).
- *Line color space*: Espace de référencement des couleurs choisi pour la couleur de la ligne ou du contour (voir Annexe 4)

## ANNEXE 2. BUILDER: Paramètres des réponses

Dans cette annexe, uniquement les réponses les plus communes sont détaillées. Pour les autres, tous les détails sont fournis sur le site de conception de PsychoPy dans le [manuel d'utilisation](#).

### 1. Clavier ('keyboard')

Le composant Clavier peut être utilisé pour collecter des réponses du participant. En ne stockant pas le Key press et en cochant 'force End Trial', cela peut simplement terminer une routine.

#### . Onglet 'Basic'

- *Name*: Nom de l'événement. Ce nom doit être unique, et ne peut contenir que des lettres, des chiffres ou des underscores (pas de marques de ponctuation ou d'espace).
- *Start*: Le temps auquel le clavier doit être contrôlé la première fois (voir Annexe 3 pour les détails).
- *Stop*: Temps à partir duquel le clavier n'est plus contrôlé (voir Annexe 3 pour les détails).
- *Force end routine*: Si la case est cochée, la routine terminera dès que l'une des touches permises ('allowed key') est pressée.
- *Allowed keys*: Liste des touches de réponse permises (par ex.: ['m','z','1','2']) ou bien le nom d'une variables contenant cette liste. Si le champ est laissé blanc, alors n'importe quelle touche pressée sera lue et considérée comme une touche permise. Seules les touches permises comptent comme touche pressée et l'appui sur toute autre touche pressée

ne sera pas enregistré et ne permettra pas de terminer une routine. Attention: les noms des touches doivent être donnés entre apostrophe (même pour les touches représentant des nombres), séparés par des virgules. Les flèches de contrôle de curseur ont pour nom 'up', 'down',...

**Pour connaître le nom des autres touches spéciales, faites tourner le script "what\_key.py" stocké dans les démos du Coder, appuyez sur une touche et regardez l'output dans la fenêtre d'output du Coder.**

- *Store*: Définit le cas échéant quels appuis de touches doivent être enregistrés: la touche correspondant au premier appui, au dernier appui, à tous les appuis. Si un appui termine la routine, alors ce champ est optionnel: c'est nécessairement la première et dernière touche pressée qui est enregistrée. Le temps va également être enregistré si un appui est enregistré. Ce temps correspond au temps écoulé entre le début de la vérification du clavier et le moment d'appui (par ex., si la vérification du clavier a été initiée 2s après le début de la routine et que la touche a été appuyé au temps  $t = 3.2$  secs, le temps de réponse enregistré est de 1.2 secs).
- *Store correct*: Cette fonction est à cocher si l'on souhaite enregistrer si oui ou non la touche pressée est la correcte. Si c'est le cas, cela nécessite de remplir le champ à côté avec la liste des touches de réponse correcte: par exemple 'left', '1', ou \$corrAns si vous avez une colonne corrAns dans une liste externe.
- *Discard previous*: A cocher pour s'assurer que c'est



uniquement les touches pressées durant cette période de ckeck de clavier qui sont considérées. Si cette option n'est pas sélectionnée, un appui de touche qui a eu lieu avant l'essai (i.e., dans un essai précédent) sera interprété comme le premier appui. Cette option doit être sélectionnée dans la plupart des expériences.

## 2. Souris ('mouse')

Ce composant peut être utilisé de plusieurs manières, notamment pour enregistrer la position du curseur lors d'un appui ou pour contrôler les paramètres de stimuli (par ex., utiliser la souris pour rendre un stimuli plus grand ou plus petit).

### . Onglet 'Basic'

- *Name*: Nom de l'événement. Ce nom doit être unique, et ne peut contenir que des lettres, des chiffres ou des underscores (pas de marques de ponctuation ou d'espace).
- *Start*: Le temps auquel la souris doit être contrôlée la première fois (voir Annexe 3 pour les détails).
- *Stop*: Temps à partir duquel la souris n'est plus contrôlée (voir Annexe 3 pour les détails).
- *Force end routine*: Si la case est cochée, la routine terminera dès que l'un des boutons de la souris sera pressé.
- *Save mouse state*: Précise la fréquence à laquelle l'état de la souris nécessite d'être sauvegardé: chaque fois qu'un participant a cliqué ? à la fin d'un essai ? à chaque frame ? Notez que l'output de texte pour les cas où vous souhaitez stocker l'output de la souris de façon répétée est compliqué à lire (le mieux est de regarder dans le fichier .psydat), mais cela devrait s'améliorer dans les versions ultérieures de PsychoPy.
- *Time relative to*: Lorsque l'état de la souris est sauvegardé (i.e., lors d'un clic ou de la fin d'un essai), un temps est sauvegardé également. Voulez-vous que ce temps soit relatif par rapport au début de la routine ou au début de l'expérience ?

### 3. Echelle ('rating')

Une échelle est utilisée pour collecter des notations/évaluations ou un choix parmi plusieurs alternatives, via la souris, le clavier ou les deux. A la fois la réponse et le temps écoulé sont enregistrés. Une routine donnée peut contenir une image (patch component) accompagnée d'une échelle de notation pour collecter la réponse.

Trois styles différents sont disponibles dans les paramètres basiques: (1) 'visual analog scale': le participant utilise la souris pour positionner le marqueur sur une ligne non graduée, (2) 'category choices': choix entre plusieurs labels verbaux (catégories: par ex., 'Un peu', 'beaucoup', 'énormément'), (3) 'scale description': à utiliser avec des formats numériques (par ex., scores de 1 à 7). Un control complet de l'affichage est possible via l'onglet 'custom'.

#### . Onglet 'Basic'

- *Name*: Nom de l'événement. Ce nom doit être unique, et ne peut contenir que des lettres, des chiffres ou des underscores (pas de marques de ponctuation ou d'espace).
- *Start*: Le temps auquel le clavier doit être contrôlé la première fois (voir Annexe 3 pour les détails).
- *Stop*: Temps à partir duquel le clavier n'est plus contrôlé (voir Annexe 3 pour les détails).
- *Visual analog scale*: Si cette option est cochée, une ligne sans graduation sera présentée avec un marqueur, et une notation entre 0 et 1.00 sera retournée (quasi-continu). Cela est fait pour faire en sorte que les participants ne pensent pas en terme de nombres, mais qu'ils se focalisent plus sur

la barre visuelle quand ils font leur évaluation. Cela détrône les choix suivants à faire ou *Scale description*.

- *Category choices*: Au lieu d'une échelle numérique, vous pouvez présenter au participant des mots ou des petites phrases sur lesquels il doit faire son choix. Entrez dans ce champ tous les mots au format texte (plus de 6 risque de ne pas être beau visuellement). Des espaces sont supposés séparer les mots. S'il y a des virgules, le texte sera interprété comme une liste de mots ou de phrases (incluant éventuellement des espaces) qui sont séparés par des virgules.
- *Scale description*: Brèves instructions, rappelant comment le participant doit interpréter l'échelle numérique (e.g., "1 = pas du tout, 2 = ..., 7 = extrêmement").
- *Lowest value*: Le chiffre le plus bas de l'échelle. Par défaut: 1.
- *Highest value*: Le chiffre le plus haut de l'échelle. Par défaut: 7.
- *Labels*: Label pour les deux extrémités de l'échelle, au format texte (et non numérique).
- *Marker start*: Position initial du marqueur.
- *Marker type*: Forme du marqueur (triangle, circle,...)

#### . Onglet 'Advanced'

- *Size*: Ce champ contrôle la taille de l'échelle sur l'écran. Un chiffre plus grand que 1 rendra plus grande l'échelle, un chiffre plus petit que 1, la rendra plus petite.
- *Position*: Position du centre du stimulus, dans les unités spécifiées. Par défaut, l'échelle est centrée sur l'axe gauche-

droite et un peu plus basse que le centre vertical ([0, -0.4]).

- *Tick height*: Taille des graduations.
- *Disappear*: Si coché, cache l'échelle lorsque la réponse a été acceptée.
- *Force end routine*: Si l'option est cochée, la routine s'arrête quand le sujet a répondu sur l'échelle.
- *Show accept*: Si coché, le bouton de validation reste visible.
- *Single click*: Si l'option est cochée, le participant ne peut cliquer qu'une seule fois et sa réponse sera stockée. Si l'option n'est pas cochée, le participant doit valider son choix avant qu'il ne soit stocké.
- *Store history*: Si coché, sauvegarde l'historique (réponse, temps)
- *Store rating*: Si coché, sauvegarde la réponse du participant.
- *Store rating time*: Sauvegarde du temps depuis le début de l'essai jusqu'à la réponse du participant.

## . Onglet 'Custom'

- *Customize everything*: Si ce champ n'est pas blanc, il sera utilisé lors de l'initialisation de l'échelle comme si c'était un composant 'code' ([voir ici](#)). Cela permet de customiser l'échelle à l'infini, et de détronner les autres paramètres de la boîte de dialogue (sauf: start, forceEndTrial, duration, storeRatingTime, storeRating)

## 4. Micro

Ce composant est nouveau et est susceptible de changements dans les versions futures. Le composant micro permet d'enregistrer du son pendant une expérience. Pour cela, il suffit de spécifier le départ ('start') par rapport au début de la routine and le temps *stop* (= durée en secs). Si le champ *stop* est laissé blanc, l'enregistrement sera fait pendant 0 sec.

Le fichier son résultant est sauvegardé au format .wac (48,000 Hz, 16 bit), avec un fichier par enregistrement. Ce fichier sera enregistré dans un nouveau sous-dossier (terminant par l'extension .wav) du dossier qui contient le script de l'expérience. Le nom du fichier inclu le temps unix (voir note à la fin) de l'onset de l'enregistrement (e.g., mic-1346437545.759.wav).

Il est possible d'arrêter l'enregistrement en cours de progression en utilisant le composant Code. Dans ce cas, à chaque frame une certaine condition est vérifiée (telle que 'q' ou 'clic de souris') et appelle la méthode *.stop()* du composant micro. A ce moment, l'enregistrement est terminé et sauvegardé. Par exemple, si *mic* est le nom de votre composant micro, faites cela, à chaque frame, dans le composant Code:

```
if event.getKeys(['q']):  
    mic.stop()
```

*Note: Le temps unix correspond au temps (exprimé en secs) qui s'est écoulé entre le moment t et le 1er janvier 1970 aux USA.*

## . Onglet 'Basic'

- *Name*: Nom de l'événement. Ce nom doit être unique, et ne peut contenir que des lettres, des chiffres ou des underscores (pas de marques de ponctuation ou d'espace).
- *Start*: Le temps à partir duquel le micro doit commencer à enregistrer du son (voir Annexe 3 pour les détails).
- *Stop*: La durée pendant laquelle le micro doit enregistrer (voir Annexe 3 pour les détails).
- *Stereo*: Si coché, enregistre le son par deux canaux, sinon par un seul.

## 5. Button Box ioLab

Une boîte de réponse (button box) est un appareil qui permet de collecter les réponses de participants avec une très bonne résolution temporelle, idéalement avec une vraie précision de l'ordre de la milliseconde. A la fois la réponse (le bouton pressé) et le temps de réponse sont enregistrés. Le temps est déterminé par une horloge interne de l'appareil, ce qui rend (en théorie) la précision excellente.

Vous pouvez vérifier le fichier .log pour voir combien de temps cela prend à PsychoPy de remettre à 0 l'horloge interne de la button box. Si cela prend un moment, alors les temps de réaction enregistrés ne seront vraisemblablement pas de haute précision. Il est possible d'obtenir un facteur de correction pour votre set up ordinateur + button box si le délai temporel est constant.

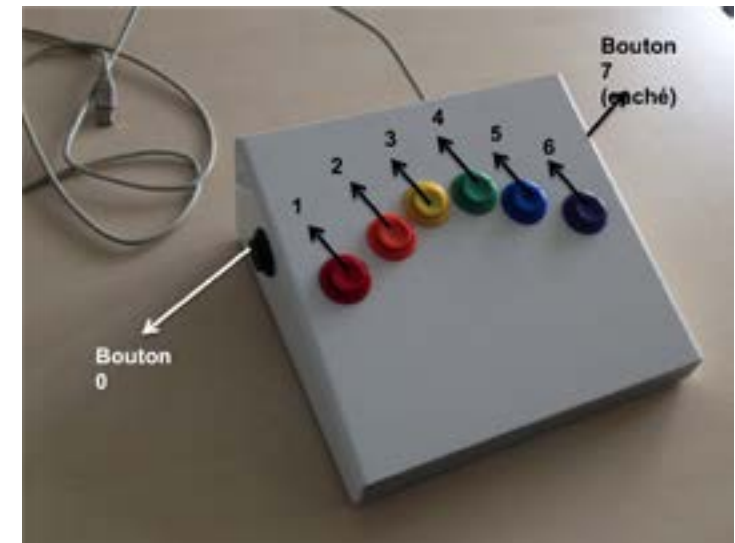
Les button box ioLab ont une clé vocale incluse (= micro), mais PsychoPy n'a pas d'interface pour le prendre en charge. Il vaut mieux dans ce cas utiliser le composant micro.

### . Onglet 'Basic'

- *Name*: Nom de l'événement. Ce nom doit être unique, et ne peut contenir que des lettres, des chiffres ou des underscores (pas de marques de ponctuation ou d'espace).
- *Start*: Le temps auquel la button box doit être contrôlée la première fois (voir Annexe 3 pour les détails).
- *Stop*: Temps à partir duquel la button box n'est plus contrôlée (voir Annexe 3 pour les détails).
- *Force end routine*: Si l'option est cochée, la routine s'arrête quand le sujet a appuyé sur une touche de la button box.
- *Active buttons*: La button box contient 8 boutons (nommés

de 0 à 7). Ce champ sert à spécifier lesquels PsychoPy doit considérer. Les réponses sur des boutons non actifs (= non entrés dans ce champ) ne sont pas considérés par PsychoPy

- *Lights*: Si coché, les LEDs au dessus des lumières seront allumées.
- *Store*: Indique quels appuis de boutons sont à sauvegarder dans le fichier de données. Les événements et la réponse sont sauvés, le temps de réponse étant celui enregistré par la button box et non par PsychoPy.
- *Store correct*: Si coché, l'exactitude de la réponse sera enregistré en fonction de la touche appuyée et de la réponse correcte (par exemple stockée dans une liste externe).
- *Discard previous*: Si coché, toute réponse précédente sera ignorée (c'est typiquement ce que vous voulez).
- *Lights off*: Si coché, toutes les LEDs seront éteintes à la fin de la routine.



## **ANNEXE 3.** Paramètres de durée/d'onset des composants



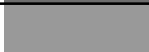
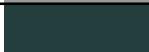
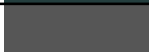

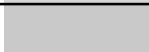

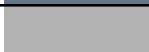















[voir Chapitre 2]

## ANNEXE 4. Espace des couleurs

























Dans PsychoPy, les couleurs peuvent être définies de deux façons différentes:







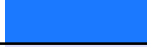

















- Par leur nom: On utilise pour cela les noms conventionnels de couleurs, tels que 'red' pour rouge, 'blue' pour bleu,... Il y a aussi des noms conventionnels pour des teintes moins basiques, comme 'sienna' pour la couleur Terre de Sienne.
- L'autre façon est d'utiliser un code à 3 chiffres qui code les valeurs RGB (red-green-blue). Dans ce cas, ce qui doit être écrit dans le champ *color* ou *color fill* d'un composant est structuré ainsi: [1, 0.5, -1], c'est-à-dire 3 chiffres compris entre -1 et 1 séparés par des virgules et entre crochets. Le premier chiffre correspond à Red, le deuxième à Green et le troisième à Blue selon le format RGB.

























Le tableau suivant donne un échantillon de couleurs avec à la fois le nom conventionnel et le code RGB.


















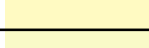
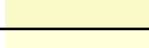





Nom conventionnel	Teinte	R	G	B
Black		-1	-1	-1
Gray		0	0	0
DarkGray		0.33	0.33	0.33
DarkSlateGray		-0.63	-0.38	-0.38
DimGray		-0.18	-0.18	-0.18
Gainsboro		0.73	0.73	0.73
LightGray		0.66	0.66	0.66
LightSlateGray		-0.07	0.07	0.2
Silver		0.51	0.51	0.51
SlateGray		-0.13	0	0.13
BurlyWood		0.74	0.45	0.06
DarkGoldenRod		0.45	0.05	-0.91
Tan		0.65	0.41	0.1
Brown		0.3	-0.67	-0.67
DarkRed		0.09	-1	-1
Maroon		0	-1	-1
Peru		0.61	0.05	-0.51
RosyBrown		0.48	0.13	0.13
SaddleBrown		0.09	-0.46	-0.85
Sienna		0.26	-0.36	-0.65
Green		-1	0	-1
Aquamarine		-0.01	1	0.66
Chartreuse		-0.01	1	-1
DarkGreen		-1	-0.22	-1









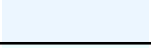

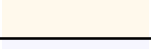
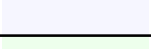





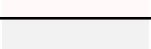
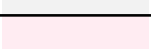



Nom conventionnel	Teinte	R	G	B
DarkKhaki		0.48	0.44	-0.16
DarkOliveGreen		-0.34	-0.16	-0.63
DarkSeaGreen		0.13	0.48	0.13
ForestGreen		-0.73	0.09	-0.73
GreenYellow		0.36	1	-0.63
LawnGreen		-0.03	0.98	-1
LightGreen		0.13	0.87	0.13
LightSeaGreen		-0.75	0.4	0.34
Lime		-1	1	-1
LimeGreen		-0.61	0.61	-0.61
MediumAquaMarine		-0.2	0.61	0.34
MediumSeaGreen		-0.53	0.41	-0.12
MediumSpringGreen		-1	0.96	0.21
Olive		0	0	-1
OliveDrab		-0.16	0.12	-0.73
PaleGreen		0.2	0.97	0.2
SeaGreen		-0.64	0.09	-0.32
SpringGreen		-1	1	-0.01
Teal		-1	0	0
YellowGreen		0.21	0.61	-0.61
DarkCyan		-1	0.09	0.09
CadetBlue		-0.26	0.24	0.26
Blue		-1	-1	1
Cyan		-1	1	1

Nom conventionnel	Teinte	R	G	B
Aqua		-1	1	1
CornflowerBlue		-0.22	0.17	0.86
DarkBlue		-1	-1	0.09
DarkSlateBlue		-0.44	-0.52	0.09
DarkTurquoise		-1	0.62	0.64
DeepSkyBlue		-1	0.5	1
DodgerBlue		-0.77	0.13	1
Lavender		0.8	0.8	0.96
LightBlue		0.36	0.7	0.8
LightCyan		0.76	1	1
LightSkyBlue		0.06	0.62	0.96
LightSteelBlue		0.38	0.54	0.74
MediumBlue		-1	-1	0.61
MediumTurquoise		-0.44	0.64	0.6
PaleTurquoise		0.38	0.87	0.87
SlateBlue		-0.17	-0.3	0.61
PowderBlue		0.38	0.76	0.8
RoyalBlue		-0.49	-0.18	0.77
SkyBlue		0.06	0.62	0.84
SteelBlue		-0.45	0.02	0.41
Turquoise		-0.5	0.76	0.63
Magenta		1	-1	1
Pink		1	0.51	0.59
Purple		0	-1	0

Nom conventionnel	Teinte	R	G	B
BlueViolet		0.09	-0.66	0.77
DarkMagenta		0.09	-1	0.09
DarkOrchid		0.2	-0.61	0.6
DarkViolet		0.16	-1	0.66
DeepPink		1	-0.84	0.16
Fuchsia		1	-1	1
HotPink		1	-0.18	0.41
Indigo		-0.41	-1	0.02
LightPink		1	0.43	0.52
MediumOrchid		0.46	-0.34	0.66
MediumPurple		0.16	-0.13	0.72
MediumSlateBlue		-0.04	-0.19	0.87
MediumVioletRed		0.56	-0.84	0.05
MidnightBlue		-0.8	-0.8	-0.13
Navy		-1	-1	0
Orchid		0.71	-0.13	0.68
PaleVioletRed		0.72	-0.13	0.16
Plum		0.73	0.26	0.73
RebeccaPurple		-0.2	-0.6	0.2
Thistle		0.7	0.5	0.7
MistyRose		1	0.79	0.77
Violet		0.87	0.02	0.87
Orange		1	0.3	-1
Red		1	-1	-1

Nom conventionnel	Teinte	R	G	B
Chocolate		0.65	-0.18	-0.77
Coral		1	-0.01	-0.38
Crimson		0.73	-0.84	-0.53
DarkOrange		1	0.1	-1
DarkSalmon		0.83	0.18	-0.05
FireBrick		0.4	-0.73	-0.73
GoldenRod		0.71	0.3	-0.75
IndianRed		0.61	-0.28	-0.28
LightCoral		0.88	0	0
LightSalmon		1	0.26	-0.05
OrangeRed		1	-0.46	-1
Salmon		0.96	0	-0.11
SandyBrown		0.91	0.29	-0.25
Tomato		1	-0.23	-0.45
Yellow		1	1	-1
Gold		1	0.69	-1
Khaki		0.88	0.8	0.1
LemonChiffon		1	0.96	0.61
LightGoldenRodYellow		0.96	0.96	0.65
LightYellow		1	1	0.76
AntiqueWhite		0.96	0.84	0.69
Beige		0.92	0.92	0.73
Bisque		1	0.79	0.54
BlanchedAlmond		1	0.84	0.61

<b>Nom conventionnel</b>	<b>Teinte</b>	<b>R</b>	<b>G</b>	<b>B</b>
Cornsilk		1	0.95	0.73
Linen		0.96	0.88	0.8
Moccasin		1	0.79	0.42
NavajoWhite		1	0.74	0.36
PaleGoldenRod		0.87	0.82	0.34
PapayaWhip		1	0.88	0.67
PeachPuff		1	0.71	0.45
Wheat		0.92	0.74	0.41
White		1	1	1
AliceBlue		0.88	0.95	1
Azure		0.88	1	1
FloralWhite		1	0.96	0.88
GhostWhite		0.95	0.95	1
HoneyDew		0.88	1	0.88
Ivory		1	1	0.88
MintCream		0.92	1	0.96
OldLace		0.98	0.92	0.8
SeaShell		1	0.92	0.87
Snow		1	0.96	0.96
WhiteSmoke		0.92	0.92	0.92
LavenderBlush		1	0.88	0.92

## ANNEXE 5. Unités de mesure (fenêtre et stimuli)

Un des avantages de PsychoPy est que les stimuli peuvent être décrits dans une large variété de mesures, certaines étant indépendantes des appareils (écrans).

Pour rappel, vous avez tout intérêt à choisir le type de mesure choisir dans les paramètres de l'expérience (*Experiment settings*).

Pour toutes les unités, le centre de l'écran est représentée par les coordonnées (0,0), avec les valeurs négatives à gauche et en bas, et les valeurs positives à droite et en haut. Cela est expliqué en détail dans l'annexe suivante, dans le cas des pixels.

### 1. Unités 'height'

Cet espace de mesure ('height') permet de tout spécifier relativement à la hauteur de la fenêtre (attention, hauteur de la fenêtre et non de l'écran).

Les dimensions d'un écran en format standard 4:3 auront des dimensions qui vont des coordonnées [-0.6667,-0.5] dans le coin bas gauche aux coordonnées [+0.6667, +0.5] dans le coin haut droite. Pour un écran 16:10, les coordonnées du coin bas gauche seront [-0.8, -0.5] et [+0.8, +0.5] dans le coin haut droite.

### 2. Unités normalisées ('normalised units')

Dans le système d'unités normalisées ('norm'), la fenêtre va de -1 à 1 à la fois pour les x et les y. Le coin haut droit de la fenêtre a comme coordonnées [1,1], le coin bas gauche a [-1,-1].

### 3. Centimètres sur l'écran

Permet d'indiquer la taille et la localisation du stimulus en centimètres sur l'écran.

### 4. Degrés d'angle

Utilise les degrés d'angle visuel pour ajuster la taille et position des stimuli. C'est totalement dépendant de la distance de à laquelle le participant est assis (par rapport à l'écran).

### 5. Pixels sur l'écran

Permet de spécifier la localisation et la taille en pixel. Cela a le désavantage que la taille et position des objets est spécifique de l'écran. Cela a l'avantage d'être intuitif et simple d'utilisation.

Cette description est sommaire à certains égards. Vous trouverez plus d'informations en suivant [ce lien](#).

## ANNEXE 6. Coordonnées à l'écran

Cette annexe présente le système de mesure sur un écran dans l'espace des pixels, afin d'encoder correctement la position des stimuli.

### 1. Principe de base

Admettons que votre écran ait une résolution de 1280 x 1024 pixels. Par convention, le premier chiffre indique toujours la largeur (autrement dit l'étendue sur l'axe des x) et le deuxième la hauteur (i.e., l'étendue sur l'axe des y).

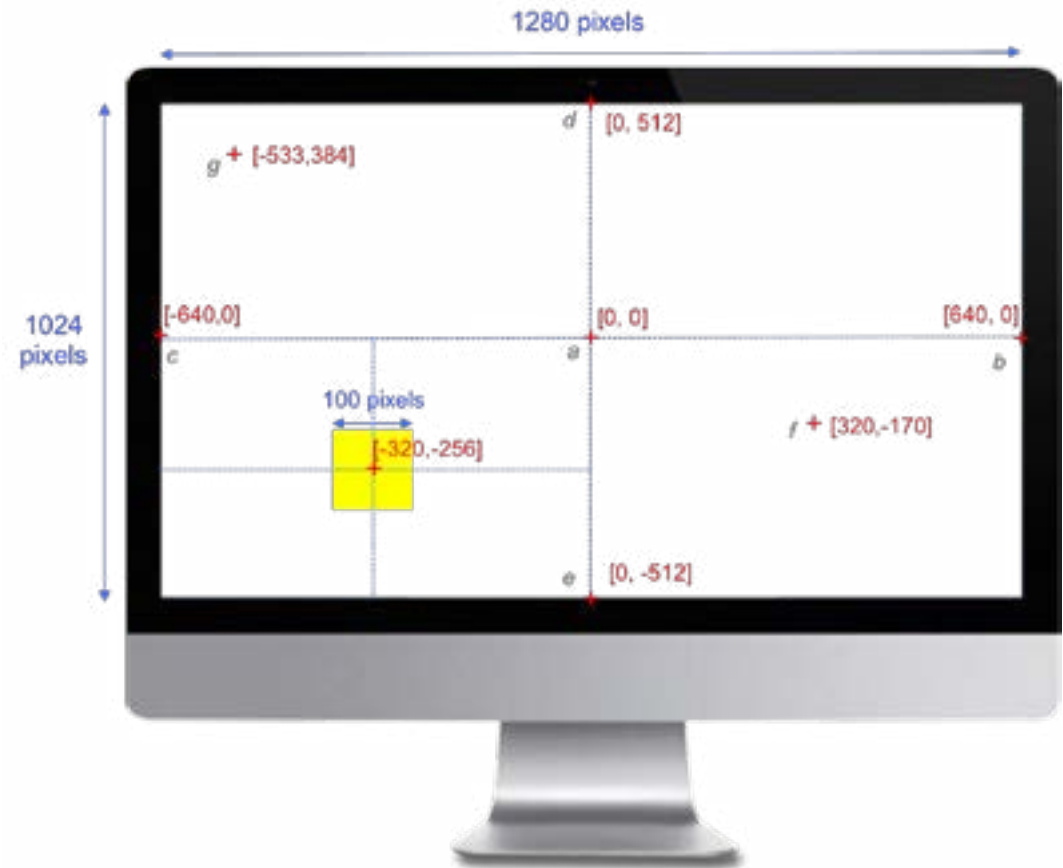
Dans PsychoPy, le centre de l'écran (point *a* sur l'image) a pour valeur  $[0,0]$ , signifiant que la coordonnée x vaut 0 et la coordonnée y vaut 0 aussi. Dans ce cas, la valeur maximale de x est 640 (correspondant à  $1280/2$ ) et la valeur minimale vaut -640 (soit  $-1280/2$ ). Le même raisonnement est utilisé pour les valeurs maximale et minimale sur l'axe des y. Par conséquent, tous les points dans le quart bas gauche ont des coordonnées x et y négatives, tous les points dans le quart haut droit ont des coordonnées x et y positives, etc...

### 2. Parler en position relative

Une des limites d'utiliser l'espace des pixels comme espace de mesure à l'écran est qu'il est fort probable que vous changiez d'écran. Prenons un cas extrême. Admettons que vous avez préparé votre expérience sur un écran 1280 x 1024 pixels et que vous voulez présenter un point de fixation au 3/4 à droite de l'écran et au 2/3 en bas de l'écran. Cela revient à placer le point en coordonnées x

à la moitié de la partie droite. Donc la coordonnée est  $640/2 = 320$ . Pour les coordonnées y, cela revient à placer le point à un tiers dans la partie basse, donc la coordonnée y est  $-512/3 = -170$ . Les coordonnées du point sont donc  $[320,-170]$ .

Imaginons maintenant que vous testiez un participant sur un écran de résolution 2560 x 1440. Le point sera toujours affiché à la position  $[320, -170]$ , or avec cette nouvelle résolution, 320 ne correspond pas sur tout au 3/4 de l'axe des x (la bonne valeur serait



640 avec cette nouvelle résolution). Il se produit la même chose pour la valeur -170.

Bref, vous l'avez peut-être compris: il vaut mieux toujours parler en coordonnées relatives. Ce n'est pas possible dans le Builder, mais ça l'est dans le Coder.

Admettons que vous créez au début de votre script 2 variables pour la résolution de votre écran:

```
xRes = 1280
```

```
yRes = 1024
```

Les coordonnées du point f seraient alors:

```
[ xRes/4, -yRes/6 ]
```

Si vous changez d'écran, il vous suffit d'indiquer la nouvelle résolution si elle est différente.

De la même manière, si vous voulez dessiner un carré qui aurait 128 pixels de côté, de sorte à ce qu'il occupe un certain espace de l'écran, il vaut mieux indiquer comme taille: **xRes/10**.

### 3. Définir la position des objets

Très souvent, ce que l'on veut afficher est plus qu'un point, mais une forme géométrique, une image, du texte...

La position est un des paramètre à indiquer. Les coordonnées à indiquer seront utiliser pour définir le centre de l'objet. Par exemple, indiquer comme position [-320, -256] pour un carré jaune de 100 pixels de côté revient à le placer au centre du quart bas de l'écran (cf. image). C'est le même principe pour les mots, les images, les vidéos,...

### 4. Taille réelle des objets

Parfois la taille physique d'un écran est la même, mais la résolution est différente. La conséquence immédiate est que sur deux écrans 'identiques', le même script ne fera pas apparaitre les objets à la même taille. En fonction de la situation, vous pourriez alors envisager d'utiliser un autre système de mesure (voir Annexe 5).

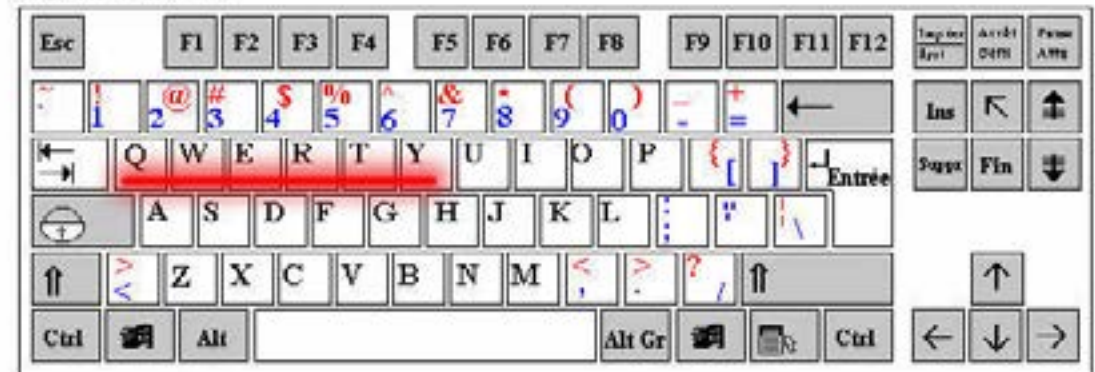
## ANNEXE 7. Claviers AZERTY et QWERTY

Le terme clavier AZERTY ou QWERTY provient de la combinaison des premières lettres d'un clavier en haut à gauche. Dans certains pays, ce sont les claviers QWERTY qui sont le plus utilisés (USA, Suisse, ...) alors que dans d'autres ce sont plutôt des claviers AZERTY (Belgique, France,...).

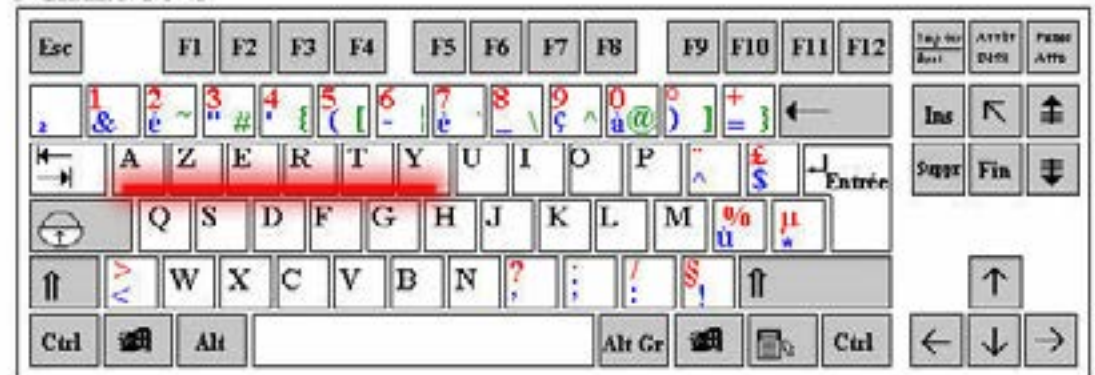
Etrangement, le Builder reconnaît un clavier AZERTY, mais le Coder considère que tous les claviers utilisés sont des QWERTY. Il y a pas mal de discussion sur les forums et des propositions de solutions. Nous considérons néanmoins que dans le cadre de ce cours, chercher la correspondances entre les deux types de claviers suffit. Voici donc une comparaison des deux de claviers (Mac à gauche, Windows à droite). Vous voyez par exemple que quand vous appuyez sur la touche A, le Coder va considérer que vous avez appuyé sur la touche Q.



### QWERTY



### AZERTY



## ANNEXE 8. Erreurs fréquentes

Voici un récapitulatif des erreurs fréquentes sous PsychoPy. Avant de vous arracher les cheveux et de gagner du temps, vérifiez si votre erreur ni serait pas ici.



### 1. Générales

- Utiliser une virgule pour la décimal au lieu d'un point
- Mettre des accents (ou autre marque diacritique) dans un des dossiers où se trouve le script ou dans le nom du script lui même
- Mettre des accents (ou autre marque diacritique) dans un om de variable
- Ne pas utiliser la bonne mesure. Par exemple, vous parlez en pixels (e.g., pour la hauteur de lettre: 30), mais vous avez oublié de le définir et PsychoPy est en unités Norm (où le maximum possible est 1). 30 dépassant 1, votre stimulus ne sera pas affiché.

### 2. Spécifiques au Builder

[voir Chapitres 1 et 2 où les erreurs fréquentes ont été renseignées]



### 3. Spécifiques au Coder

- Oublier de spécifier la fenêtre lors de l'utilisation d'une fonction pour créer un objet. Par exemple écrire :

```
visual.TextStim(text='hello') au lieu de  
visual.TextStim(w, text='hello')
```

- Oublier les parenthèses à la fin de l'écriture d'une fonction telles que **win.flip()**
- Inverser le nom des arguments. Par exemple écrire **colorFill=** au lieu de **fillColor=**
- Oublier une majuscule dans le nom d'une fonction. Par exemple écrire **textStim()** au lieu de **TextStim()** (type de message d'erreur :

*TypeError: 'module' object is not callable*)

- Inverser les morphèmes dans le nom d'une fonction. Par exemple écrire **event.keysGet()** au lieu de **event.getKeys()**
- Oublier les deux points après l'utilisation d'un contrôleur de flux
- Oublier que le comptage et l'indexation commencent à 0
- Oublier la variable pour indexer dans un script !

```
for i in range(10):
```

```
    ob = visual.TextStim(text = item) au lieu de
```

```
for i in range(10):
```

```
    ob = visual.TextStim(text = item[i])
```



## ANNEXE 9. Librairie PANDAS

Cette annexe a pour but de vous présenter plus en détail la librairie Panda. En effet, vous pourriez avoir besoin de manipuler plus en profondeur un tableau de données. Voici l'essentiel des recettes.

### 1. Introduction

Cette [librairie](#) permet de générer et manipuler de façon efficace et flexible des structure de données. Dans le cadre de la programmation PsychoPy, elle va surtout nous servir à importer et exporter des listes externes, et à manipuler des tableaux pour enregistrer des données.

Voici tout un ensemble d'information, dont vous pourriez avoir besoin. Pensez-à vous y référer quand vous voulez manipuler un tableau de données.

### 2. Création d'un tableau de données

Voyons un premier exemple. Créons deux listes et essayons d'en faire un tableau. Nous avons vu la dernière fois qu'on peut utiliser la fonction `zip()`, mais le rendu n'est pas très lisible:

```
# -*- coding: utf-8 -*-  
  
nom = ['alain', 'fabienne', 'mari']  
age = [21, 18, 32]  
  
d = zip(nom, age)  
print d  
  
> [('alain', 21), ('fabienne', 18), ('mari', 32)]
```

Rappelez vous ! La ligne `# -*- coding: utf-8 -*-` permet à Python de gérer les accents (dans vos commentaires ou dans vos

stimuli). Elle est donc indispensable !

Utilisons maintenant **pandas** pour voir la différence:

```
# importation du package (on cree un alias):  
import pandas as pd  
  
# creation d'un tableau de donnees:  
df = pd.DataFrame(data = d, columns = ['nom',  
    'age'])  
  
print df  
  
>      nom      age  
0     alain      21  
1  fabienne      18  
2     mari      32
```

Remarquez la structure utilisée pour la fonction `DataFrame()`: on indique les données, puis les titres de colonnes.

### 3. Importation d'un tableau de données

Admettons à présent que vous ne voulez pas créer le tableau dans PsychoPy, mais que vous avez déjà toutes vos données que vous souhaitez importer (cas d'une liste externe). Dans cette liste externe, vous avez des mots à afficher et la condition à laquelle ils appartiennent (Figure 2).

La façon de faire que nous allons suivre est d'avoir des listes au format texte, avec séparateur tabulation (.txt). Ce type de format est le plus flexible, le plus simple à importer et le plus portable (n'importe quel éditeur de texte lit des fichiers au format texte mais pas des fichiers au format excel). Vous pouvez utiliser également le format csv (.csv) que vous connaissez déjà qui utilise la virgule (ou

le point virgule) comme séparateur de texte.

Il est néanmoins fort probable que vous prépariez vos listes de stimuli dans un tableur (Excel, Numbers, OpenOffice). Les Vidéo 1, Vidéo 2 et Vidéo 3 montrent donc comment préparer votre liste externe au format texte ou csv.

Voici les commandes pour importer un fichier qui se trouve dans le même dossier que votre script :

```
# -*- coding: utf-8 -*-

# Importer la librairie pandas
import pandas as pd

# importer un fichier .txt
d = pd.read_table('liste.txt', encoding='utf-8')

# importer un fichier .csv
d = pd.read_csv('liste.csv', encoding='utf-8')

print d
```

	Items	Conditions
>		
0	poule	animal
1	lapin	animal
2	panda	animal
3	lynx	animal
4	marteau	objet
5	chaise	objet
6	tuba	objet
7	sacoche	objet

Attention ! L'argument `encoding='utf-8'` est essentiel si votre liste externe contient des marques diacritiques. Elle spécifie que la liste est au format utf-8 (tout comme la première ligne, un format qui permet de gérer les marques diacritiques entre autre, voir [ici](#)). Attention, vous devez ajouter `sep = ';'`  si votre séparateur est un point virgule.

## 4. Manipulation d'un tableau de données

Vous pourriez vouloir faire plusieurs choses sur ce tableau de données. Voici les principales, nous en verrons d'autres au fur et à mesure.

- **Avoir des infos sur le tableau de données :**

```
# Afficher les dimensions
print d.shape
> (8,2)

# Afficher les 5 premières lignes
print d.head()

>      Items Conditions
0     poule      animal
1     lapin      animal
2     panda      animal
3      lynx      animal
4  marteau      objet

# Afficher les 5 dernières lignes
print d.tail()

>      Items Conditions
3      lynx      animal
4  marteau      objet
5   chaise      objet
6     tuba      objet
7  sacoché      objet

# Afficher des détails
print d.info()

> <class 'pandas.core.frame.DataFrame'>
Int64Index: 8 entries, 0 to 7
Data columns (total 2 columns):
Items      8 non-null values
Conditions  8 non-null values
dtypes: object(2)None
```

```
# Afficher le nom des colonnes
print d.columns
print d.columns[1]

> Index([u'Items', u'Conditions'], dtype=object)
> 'Condition'
```

- **Supprimer des colonnes:**

```
# Supprimer une colonne par son nom
d = d.drop('Items', axis = 1)
print d

> Conditions
0    animal
1    animal
2    animal
3    animal
4    objet
5    objet
6    objet
7    objet

# Supprimer une colonne par sa position
d = d.drop(d.columns[0],axis=1)
```

`drop()` est la fonction pour supprimer. `axis = 1` sert à spécifier que vous êtes en train de considérer les colonnes.

Soit vous supprimez via le nom que vous indiquez (exemple 1), soit vous supprimez via la position (exemple 2). Pour cela vous indiquez la position dans `d.columns[]` de façon à ce que PsychoPy retrouve le nom.

Notre tableau de données ne s'y prête pas, mais vous pouvez supprimer plusieurs colonnes:

```
d = d.drop(d.columns[2:3],axis=1)
d = d.drop(d.columns[[1,3,4:5]],axis=1)
```

Dans le premier cas, on indique dans `d.columns[]` l'intervalle de colonnes à supprimer. Dans le deuxième cas, on crée un vecteur indiquant les positions des colonnes à supprimer `[1,3,4:5]` qu'on insère ensuite dans `d.columns[]`.

- **Ajouter des colonnes :**

```
# Ajout d'un colonne de texte
v = ['poule','lapin','panda','lynx','marteau',
     'chaise','tuba','sacoche']

d['Items'] = v
print d

> Conditions      Items
0    animal      poule
1    animal      lapin
2    animal      panda
3    animal      lynx
4    objet      marteau
5    objet      chaise
6    objet      tuba
7    objet      sacoche
```

```
# Ajouter une colonne de nombres
v = [30,42,2,0.5,38.5,69,3.4,2.7]
d['Fq'] = v
print d
```

```
> Conditions      Items      Fq
0    animal      poule      30.0
1    animal      lapin      42.0
2    animal      panda      2.0
3    animal      lynx      0.5
4    objet      marteau     38.5
5    objet      chaise     69.0
6    objet      tuba      3.4
7    objet      sacoche     2.7
```

```
# Ajouter une colonne a une position (ici :1)
d.insert(1, 'New', range(8))
```

```
> Conditions  New    Items    Fq
0    animal    0    poule    30.0
1    animal    1    lapin    42.0
2    animal    2    panda    2.0
3    animal    3    lynx     0.5
4    objet     4    marteau  38.5
5    objet     5    chaise   69.0
6    objet     6    tuba     3.4
7    objet     7    sacoché  2.7
```

- **Supprimer des lignes :**

```
# Une ligne (2eme position)
d = d.drop(d.index[1])
print d
```

```
> Conditions  New    Items    Fq
0    animal    0    poule    30.0
2    animal    2    panda    2.0
3    animal    3    lynx     0.5
4    objet     4    marteau  38.5
5    objet     5    chaise   69.0
6    objet     6    tuba     3.4
7    objet     7    sacoché  2.7
```

```
# plusieurs lignes
d = d.drop(d.index[[1,3]])
print d
```

```
> Conditions  New    Items    Fq
0    animal    0    poule    30.0
2    animal    2    panda    2.0
4    objet     4    marteau  38.5
6    objet     6    tuba     3.4
7    objet     7    sacoché  2.7
```

`d.index[]` sert ici à indexer la position des lignes concernées.

- **Changer la dernière ligne :**

```
d[-1:] = ['animal', 1, 'lapin', 42.0]
print d
```

```
> Conditions  New    Items    Fq
0    animal    0    poule    30.0
2    animal    2    panda    2.0
4    objet     4    marteau  38.5
6    objet     6    tuba     3.4
7    animal    1    lapin    42.0
```

`[-1:]` fait référence à la sélection de toutes les colonnes de la dernière ligne. La commande signifie donc *‘remplace toute la dernière ligne par...’*

- **Sélectionner une sous partie des données :**

Pour sélectionner des données (= indexer des positions dans un tableau), **pandas** nécessite d'utiliser des fonctions pour indexer: `loc()`, `iloc()`, et `ix()`.

`loc()` permet d'indexer sur base du label, `iloc()` sur base de la position et `ix()` sur un mix des deux. Nous n'allons pas rentrer dans le détail (si vous voulez tout savoir, [cliquez ici](#)), car c'est `iloc()` que nous allons utiliser, qui est le plus intuitive.

Voici des exemples à comprendre pour récupérer des données dans un tableau. Petit rappel, **les crochets servent à indexer. Comme dans R, on commence toujours par nommer les lignes, puis nommer les colonnes:**

1. Si vous écrivez `[2,3]` vous faites référence à la donnée au croisement de la ligne en 2ème position et de la colonne en 3ème position.

2. Si vous écrivez `[:,2]` vous faites référence à toutes les lignes

(:) de la colonne en 2ème position

3. Si vous écrivez `[4,:]` vous faites référence aux valeurs de toutes les colonnes pour la ligne en 4ème position

4. Très important ! **N'oubliez pas que lorsqu'on indexe en Python avec un intervalle, la dernière des valeurs est exclue !** Si vous écrivez `[:,3:8]`, vous faites référence à toutes les colonnes de la position 3 à la position 8 **exclue !**

- Sélectionner des colonnes :

```
# methode 1: utiliser le nom de la colonne
print d.Items

> 0 poule
2 panda
4 marteau
6 tuba
7 lapin
Name: Items, dtype: object

# methode 2: selection sur base de la position
print d.iloc[:,0]

> 0 animal
2 animal
4 objet
6 objet
7 animal
Name: Conditions, dtype: object

# Plusieurs colonnes qui se suivent(col. 2-3):
print d.iloc[:,1:3]

> New Items
0 0 poule
2 2 panda
4 4 marteau
```

```
6 6 tuba
7 1 lapin

# Plusieurs colonnes separees (col. 2/4)
print d.iloc[:,[1,3]]

> New Fq
0 0 30.0
2 2 2.0
4 4 38.5
6 6 3.4
7 1 42.0
```

- Sélectionner des lignes :

C'est le même principe qui est appliqué ici:

```
# Selection d'un ligne (2eme pos)
print d.iloc[1,:]

>
Conditions animal
New 2
Items panda
Fq 2
Name: 2, dtype: object

# Selection d'un intervalle de lignes
print d.iloc[1:3,:]

> Conditions New Items Fq
2 animal 2 panda 2.0
4 objet 4 marteau 38.5

# selection de plusieurs lignes separees
print d.iloc[[1,2,4],:]

> Conditions New Items Fq
2 animal 2 panda 2.0
4 objet 4 marteau 38.5
7 animal 1 lapin 42.0
```

- **Sélectionner des valeurs au croisement ligne/colonne:**

```
# Methode 1: nom de la colonne + ajout de la ligne
print d.Conditions[2]

> animal

# Methode 2: le nom de la colonne n'est pas donné
print d.iloc[1,2]

> panda

# Selection de plusieurs lignes de plusieurs
# colonnes
print d.iloc[[1,5],[0,2]]

> Conditions  Items
2      animal  panda
7      animal  lapin
```

- **Changer des valeurs :**

Comme nous allons le voir, vous allez souvent changer les valeurs d'un tableau de données, ne serait-ce que parce que les temps de réponse vont s'incrémenter. Comment faire cela ? Ce n'est pas très compliqué, il suffit de suivre les procédures précédentes !

```
# Changer une valeur d'une colonne (2 façons) :
d.Items[0] = 'vache'
d.iloc[0,3] = 61.7

> Conditions  New      Items      Fq
0      animal      0      vache      61.7
2      animal      2      panda      2.0
4      objet      4      marteau    38.5
6      objet      6      tuba       3.4
7      animal      1      lapin     42.0
```

Le principe est le même pour changer toutes les valeurs d'une ligne ou d'une colonne, sauf que ce qui est à droite du signe = doit

être de la longueur de la colonne ou la ligne.

## 5. Exportation d'un tableau de données

Le principe que nous allons suivre dans les scripts à venir est d'importer une liste externe sous forme de dataframe, puis de modifier ce tableau au fur et à mesure des essais (ajout du temps de réponse, de l'exactitude, des metadata,...), et enfin d'exporter ce tableau.

Le tableau exporté contiendra donc la liste des stimuli, les conditions expérimentales que vous jugez importantes, et les données enregistrées. Autrement dit, vous aurez un fichier de résultats brut tout à fait prêt pour l'importer sous R et commencer à traiter les données. Voici la commande très simple d'exportation que nous allons utiliser:

```
d.to_csv('data.csv', index=False, header=True,
sep="\t")
```

- **to\_csv()** est la fonction qu'on utilise pour exporter **d**
- **index=False** signifie que vous ne voulez pas que les chiffres qui indexent les lignes soient marqués dans le fichier de sortie
- **header=True** signifie que vous voulez que les noms des colonnes soient sauvegardés en tant que tels
- **sep="\t"** est un argument qui permet d'utiliser des tabulations comme séparateurs de données dans votre fichier de sortie