PiCameraGUI Documentation

Release 1.0

Jérémy Talbot-Pâquet

Aug 09, 2018

Contents

Ce programme fournit un interface graphique (GUI) pour le module de caméra de Raspberry Pi. Ce programme est écrit dans le language de programmation Python à l'aide de la librarie Tkinter.

CHAPTER 1

Fonctionnalités

- Aperçu en temps réel
- Aperçu de la photo prise
- Prise de photo, vidéo et de photos en séquence
- Zoom et déplacement à l'intérieur de l'image en temps réel
- Ajout de texte et du temps présent sur la photo
- Différents formats de photo supportés
- Rotation et revirement horizontal et vertical de l'image

CHAPTER 2

Liens

- Tout le code source est disponible depuis la plateforme GitHub
- Toute la documentation est disponible sur ReadTheDocs
- Merci à la documentation complète sur les modules de la *Pi Camera* accesssible sur Read The Docs
- Ce code est licencé sous la Licence BSD

CHAPTER 3

Bienvenue sur la documentation de PiCamera GUI!

Table des matières:

Installation

Pour accéder au logiciel,

- Installer Python 3. Si votre système Raspberry Pi est à jour, vous pouvez ignorer cette section.
- Télécharger le projet disponible ici
- Déplacer le fichier PiCamera `[Desktop Entry]` sur le bureau. Il devrait avoir l'icône du module PiCamera V2
- Modifier le chemin du fichier à exécuter en ouvrant appuyant sur le `[Desktop Entry]`. Voir *Ouvrir l'application*
- Ouvrir le fichier PiCamera

Installation de Python 3

Le plus important est de s'assurer que votre système contient les packages de la picamera. Pour ce faire, entrez la commande suivante dans l'invite de commande de votre Raspberry Pi:

```
$ sudo apt-get update
$ sudo apt-get install python3-picamera
```

Si vous éprouvez des difficultés pour mettre à jour votre Raspberry Pi, consultez cette page bien détaillée de la documentation du package Picamera.

Télécharger le projet

Vous pouvez importer le projet en entrant la commande suivante:

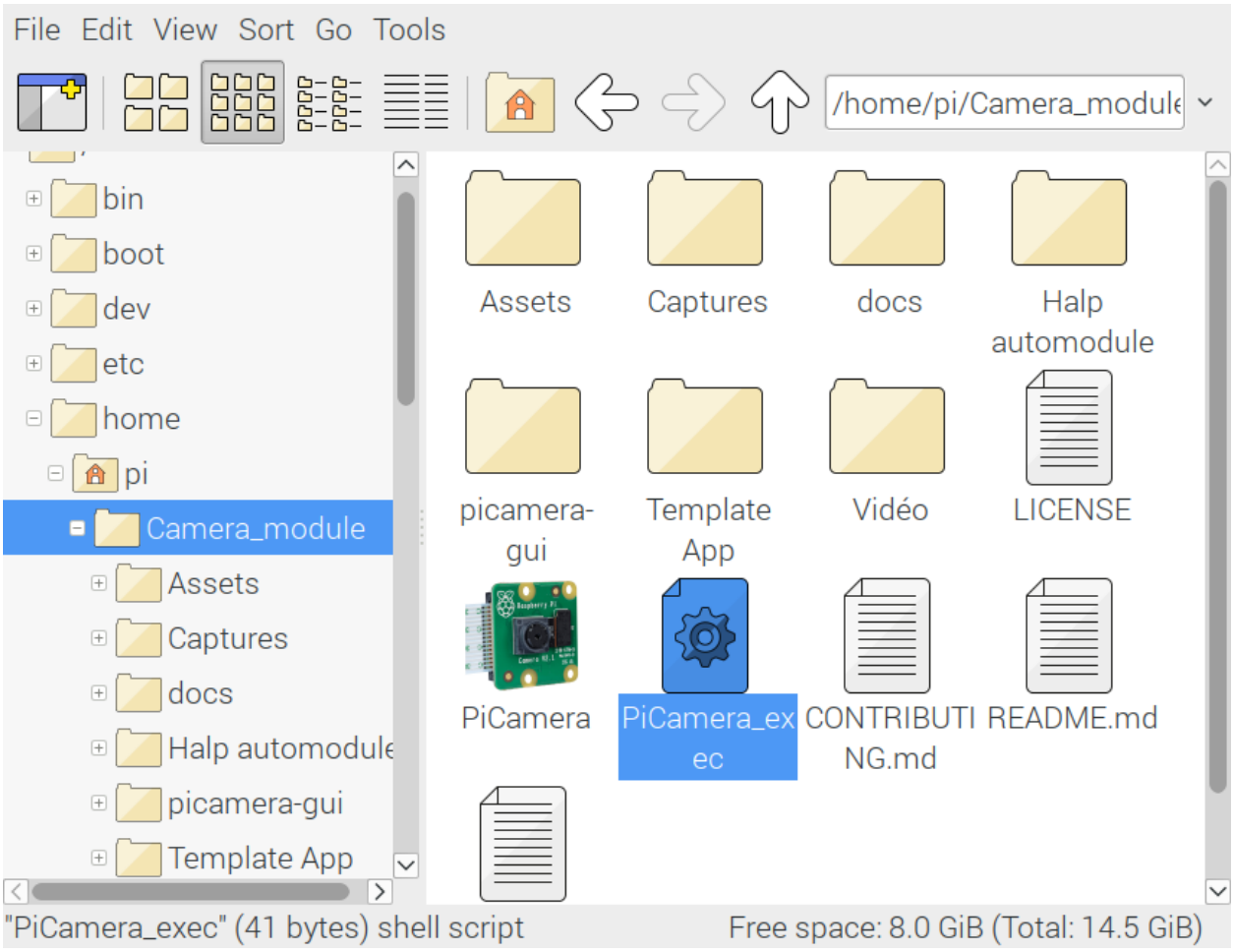
```
$ sudo apt-get install git
$ git clone https://github.com/jtpaquet/PiCamera-GUI
```

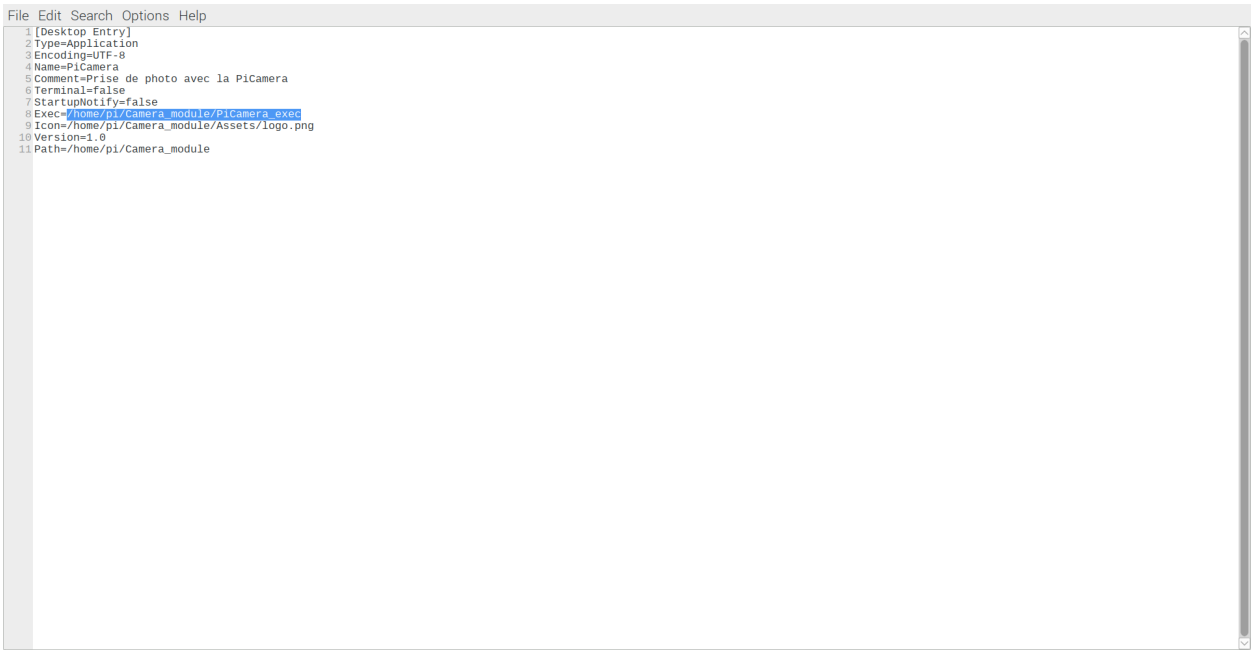
Sinon, vous pouvez télécharger le projet en tant que fichier compressé .zip et le déplacé dans le répertoire voulu.

Ouvrir l'application

Déplacer le fichier ayant cette icône sur le bureau.

Copiez le chemin du fichier (*right click* > *copy path*) du fichier `PiCamera_exec` et coller le chemin dans le fichier avec l'icône déplacé sur le bureau.





Cela modifie le chemin du fichier à exécuter en ouvrant appuyant sur le `[Desktop Entry]`. Cela exécute un script qui effectue la ligne de commande:

```
$ python3 picamera-gui/main.py
```

En ouvrant ce fichier, l'interface graphique devrait s'ouvrir et vous devriez pouvoir l'utiliser. Assurez vous que la PiCamera soit bien branchée. Si l'erreur persiste, essayez de redémarrer le Rapsberry Pi.

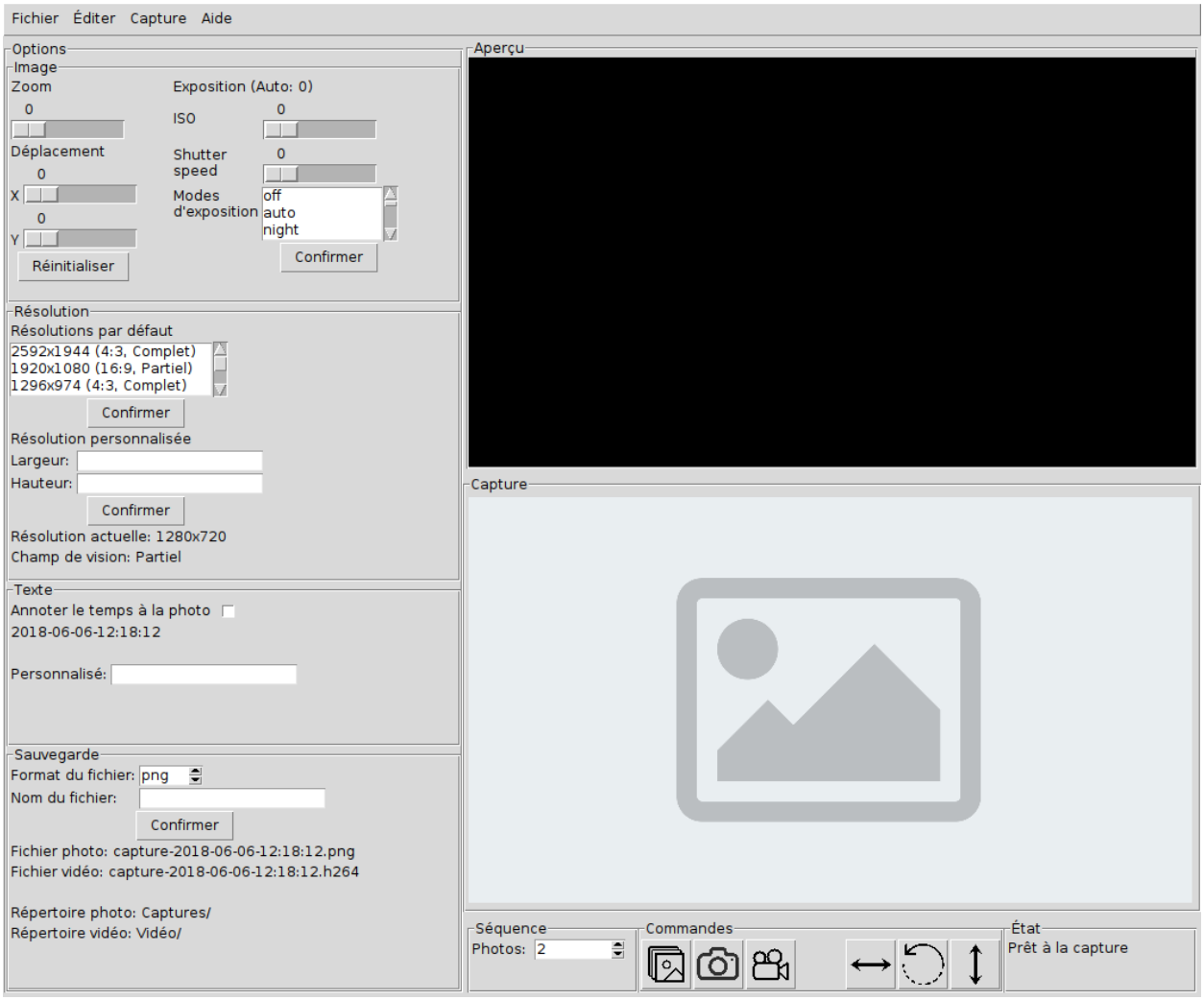
Sinon, vous pouvez ouvrir l'invite de commande (F4) dans le répertoire principal dans lequel se trouve les fichiers et entrer la ligne de commande:

```
$ python3 picamera-gui/main.py
```

Guide de l'utilisateur

À l'initialisation du logiciel, l'aperçu en temps réel devrait apparaître. Vous avez alors plusieurs fonctionnalités à votre disposition.

Note: Le prévisionnement est un Overlay, ce qui signifie qu'il est généré «par-dessus» l'écran. C'est pour cela qu'en déplaçant la fenêtre, il ne suit pas immédiatement la position de la fenêtre. Son ajustement est fait de manière à ce que sa position s'adapte lorsque le programme détecte un mouvement de la fenêtre principale (*root*) et la fonction est seulement appelée lorsque le déplacement du *root* est complété. On ne peut pas prendre de capture d'écran du prévisionnement puisque c'est un *Overlay*. C'est comme s'il était par-dessus la capture. Pour plus d'informations, consultez la page du module PiCamera sur l'aperçu et l'*Overlay*.



Démarrage rapide

Si vous avez téléchargé le dossier au complet, vous remarquerez deux dossiers dans le répertoire principal: **Captures** et **Vidéo**. Ce sont les dossiers dans lesquels seront enregistrées vos photos et vidéos prises avec l'application. Si vous souhaitez enregistrer vos captures dans un autre répertoire vous pouvez commencer par ouvrir le menu **Fichier** et choisir un nouveau répertoire. Pour plus d'options, consultez la section *Modification de la sauvegarde*.

Vous pouvez maintenant prendre des photos en appuyant sur la barre espace ou sur le bouton avec l'icône d'un appareil photo dans le menu *Commandes* sous l'aperçu. Remarquez la barre d'état en bas à droite: elle indique si vous pouvez prendre une photo et ensuite dans quel répertoire la capture est enregistrée.

Pour prendre des vidéos, appuyez sur le bouton avec l'icône d'un caméscope. Un rond rouge vous indiquera que vous êtes en train de filmer. Pour finir l'enregistrement, appuyez à nouveau sur le bouton. La barre d'état vous dira dans quel répertoire la vidéo est enregistrée.

Note: Le format des fichiers vidéo est .h264 par défaut et le programme filme à 24 fps. Si vous voulez lire les vidéos, allez dans le répertoire dans lequel se trouve le fichier et exécutez la commande suivante dans l'invite de commande:

```
omxplayer --fps 24 file_name.h264
```

Omxplayer est le lecteur vidéo du Raspberry Pi.

Zoomer

Vous pouvez zoomer dans l'image à l'aide de la glissoire **Zoom**. Après avoir choisi le zoom désiré, vous pouvez vous déplacer avec les deux autres glissoires **X** et **Y**. Celles-ci vous feront déplacer horizontalement et verticalement respectivement. Pour la glissoire **X**, une valeur 0 indique que vous vous trouvez complètement à gauche de l'image et une valeur de 100, complètement à droite. Pour la glissoire **Y**, 0 indique que vous vous trouvez complètement en haut de l'image et une valeur de 100, complètement en bas.

Note: Il y a une sécurité qui limite le zoom maximal dans le code. À un certain point, la caméra zoome sur seulement quelques pixels et cela fait planter l'application. Si vous devez absolument zoomer plus que ce que le programme propose, vous pouvez changer la valeur 1.05 à la ligne 780 du fichier **PiCameraGUI.py** dans la fonction set_previewScale par 1.00. Cela vous permettera de zoomer jusqu'aux limites de la PiCamera, mais c'est déconseillé. Vous êtes conseillez de remettre ce paramètre à 1.05 par la suite.

Pour réinitialiser le zoom de l'image comme elle était à l'ouverture du programme, appuyez sur le bouton **Réinitialiser** sous la glissoire **Y**. Ça fait la même chose que de remettre les trois glissoires à 0.

Paramètres de l'image

Vous pouvez ajuster les paramètres de l'image avec les options sous le menu Exposition. L'`ISO` change la luminosité de l'image ainsi que la quantité de *grain*/bruit. Un faible ISO donnera une image claire en terme de bruit, mais sombre en terme de lumière. Un ISO élevé fera augmenter la luminosité de l'image, mais augmentera aussi la quantité de grain. Le `shutter speed` change le temps d'exposition. Un shutter speed faible permet de capter plus de lumière et d'avoir une image plus éclairée. Un shutter speed plus élevé permet de prendre une photo plus figée dans le temps au détriment de capter moins de lumière. Lorsqu'il y a beaucoup de mouvement, on voudrait avoir un shutter speed élevé pour éviter un flou. Les `modes d'exposition` sont présentés dans la rubrique de la documentation de la picamera. Ce sont des modes préprogrammés. Veuillez consulter ce lien pour de plus amples explications.

Note: Activer un mode d'exposition empêche la configuration manuelle de l'`ISO` et du `shutter speed`.

Note: On ne peut changer l'ouverture mécanique de la PiCamera. Par contre, vous pouvez zoomer avec l'option décrite plus haut, mais sachez que l'ouverture reste fixe.

Note: Poser l'`ISO` ou le `shutter speed` à 0 activera l'ajustement automatique de ces paramètres.

Changer la résolution

L'application vous permet de changer la résolution de l'image. À l'ouverture du programme, la résolution sera de 1280x720 px (4:3). Il y a plusieurs résolutions préprogrammées dans la boîte `Résolutions par défaut`. Pour en sélectionner une, appuyer sur celle désirée et confirmez votre choix en appuyant sur le bouton confirmer directement sous la boîte. Vous pouvez voir la résolution actuelle un peu plus bas. Si vous souhaitez configurer votre propre résolution, vous pouvez entrer la largeur et la hauteur de l'image dans les deux boîtes sous `Résolution`

personnalisée. Pour confirmer votre choix, appuyez sur le bouton directement sous les deux boîtes. Le bouton au-dessus sert aux résolutions préprogrammées.

Note: Pour entrer votre résolution personnalisée, vous devez entrer deux nombres entiers. La résolution doit être supérieure à 64x64 px, la résolutoin minimale de la PiCamera, et inférieure à 2592x1944 px.

Note: Certains modes préprogrammés offrent un champ de vision partiel. Lorsque vous entrez une résolution personnalisée, le champ de vision sera toujours complet. Si vous voulez le changer, utilisez la fonctionnalité Zoom. Vous pouvez voir la configuration du champ de vision actuel sous la résolution actuelle.

Afficher un texte

Vous pouvez annoter du texte sur la photo. Pour ce faire, écrivez le texte à annoter à côté de la boîte Personnalisé dans la section **Texte**. Vous pouvez voir un aperçu ce ce qui sera affiché dans l'aperçu en temps réel à droite. Vous pouvez aussi annoter le temps présent en cochant la boîte **Annoter le temps à la photo**. Il sera annoté sous le format affiché juste au-dessous de la boîte.

Note: La fonction annotate_text ne peut contenir que les 128 caractères du code ASCII, ce qui exclue les lettres accentuées. Écrire un caractère interdit le changera automatiquement par % pour vous en indiquer.

Modification de la sauvegarde

Vous pouvez changer le format du fichier en appuyant sur les flèches de la spinbox dans la section Sauvegarde. Les formats disponibles sont

- 'jpeg' - Encodage JPEG
- 'png' - Encodage PNG sans pertes
- 'gif' - Encodage GIF
- 'bmp' - Encodage Windows bitmap
- 'rgb' - Données images brutes en format 24-bit RGB
- 'rgba' - Données images brutes en format 32-bit RGBA

Vous pouvez modifier le nom du fichier de la capture en écrivant dans la boîte Nom du fichier. Notez que le temps est ajouté suite au nom que vous aurez donné au fichier pour empêcher d'enregistrer une capture par-dessus une capture déjà existante. Appuyez sur le bouton Confirmer pour valider votre choix.

Si vous souhaitez enregistrer vos captures dans un autre répertoire vous pouvez commencer par ouvrir le menu **Fichier** et choisir un nouveau répertoire. Par défaut, le répertoire pour les photos est Capture/ et celui des vidéos est Vidéo/

Si vous n'êtes pas sûr de l'endroit de la sauvegarde ou du nom de fichier, vous pouvez voir un aperçu sous le bouton Confirmer

Autres fonctionnalités

- Changer le nombre de prise de photo en séquence

Pour changer le nombre de photos à prendre en séquence, cliquez sur les flèches de la `spinbox` dans la section `Séquence` du menu des commandes.

- Revirements et rotation

Vous pouvez effectuer un revirement horizontal, vertical ou une rotation de l'image en appuyant sur les boutons associés à ces fonctionnalités du menu des commandes.

- Tout réinitialiser

Vous pouvez réinitialiser l'application comme elle était à l'ouverture du programme dans le menu `Fichier` et en appuyant sur la commande **Tout réinitialiser**.

Notes sur le code

Voici un résumé de ce que fait le code. La plupart des commentaires sont déjà présents dans les fichiers, n'hésitez pas à les consulter au besoin.

Main.py

En premier lieu, le programme essaie de détecter s'il y a des problèmes d'initialisation. D'abord, il détecte si Python peut bel et bien initialiser la fenêtre et, ensuite, si il peut initialiser l'objet PiCamera. S'il ne peut pas, il lèves deux types d'erreurs et propose des solutions exposées *Modules d'exception*. ' L'instance *app* de la classe *PiCameraGUI.py* est crée avec `win` comme `root` et `camera` comme objet `PiCamera`. Le programme exécute ensuite la ligne:

```
win.mainloop()
```

Essentiellement, il s'agit d'une boucle infinie dans laquelle la fenêtre réagit aux divers événements (cliques, touches, déplacements de la souris, etc.). Pour mieux comprendre la fonction mainloop() de Tkinter, consulter le sujet suivant sur **Stack Overflow**. Techniquement, le programme n'est pas supposé quitter la boucle `mainloop`, mais si c'est le cas, il se doit d'effacer l'instance *camera* pour éviter les problèmes de mémoire.

PiCameraGUI.py

Ce fichier contient la classe `PiCameraGUI` qui s'occupe de la création de l'interface ainsi que des fonctionnalités du programme.

class `PiCameraGUI.`**`PiCameraGUI`**(*root*, *camera*, *title*)

Classe pour l'interface graphique de la `PiCamera`

Fonctionnalités

•Aperçu en temps réel

•Aperçu de la photo prise

•Prise de photo, vidéo et de photos en séquence

•Zoom et déplacement à l'intérieur de l'image en temps réel

•Ajout de texte et du temps présent sur la photo

•Différents formats de photo supportés

•Rotation et revirement horizontal et vertical de l'image

Parameters

- **root** (*Tk() ou Toplevel()*) – Fenêtre principal issue de la librairie `tkinter`
- **camera** (*PiCamera()*) – Objet caméra issu du module `picamera`
- **title** (*String*) – Titre de la fenêtre

Voir le fichier `main.py` pour un exemple d'utilisation

capture()
Prends une photo

captureSeq()
Prends une séquence de photos

captureSpace(*event*)
Prends une photo avec la barre espace

centrerAide()
Place la fenêtre d'aide au centre de l'écran

changerRepertoirePhoto()
Change le répertoire d'enregistrement d'un fichier photo

•Ouvre une nouvelle fenêtre

•self.photo_dir: str

changerRepertoireVideo()
Change le répertoire d'enregistrement d'un fichier vidéo

•Ouvre une nouvelle fenêtre

•self.video_dir: str

createAide()
Crée une fenêtre indiquant les informations relatives au logiciel

createBindings()
Crée les liaisons entre les widgets et les événements

createFrames()
Crée les sous-fenêtres

createImage()
Crée la sous-fenêtre `Image` et initialise les widgets de celle-ci

createLabelFrames()
Crée les sous-fenêtres des options

createMenu()
Crée la barre de menu

createRes()
Crée la sous-fenêtre `Résolution` et initialise les widgets de celle-ci

createSave()
Crée la sous-fenêtre `Sauvegarder` et initialise les widgets de celle-ci

createText()
Crée la sous-fenêtre `Texte` et initialise les widgets de celle-ci

createWidgets()
Crée les boutons de l'interface des commandes et la barre de menu

get_cmdsize()
Retourne la taille des sections `Séquence` et `État`

hflip()
Effectue le revirement horizontal de l'image

posPreview()
Trouve la position de l'aperçu en temps réel dans l'interface

quit()
Quitte le programme

recVideo()
Prends un enregistrement vidéo Appelé chaque fois que le bouton vidéo est appuyé

reset_all()
Réinitialise tous les paramètres

reset_size()
Effectue la réinitialisation de la taille de l'aperçu

rotate()
Effectue la rotation de l'image

set_expmode()
Fixe le `mode d'exposition` de l'image

set_iso(*event*)
Fixe l'`ISO` de l'image

set_overlayText()
Fixe le texte à afficher sur l'image

set_previewPos(*event*)
Ajuste la position de l'aperçu à la fenêtre

set_previewScale(*event*)
Fixe l'échelle de l'aperçu

set_res()
Vérifie et fixe la résolution personnalisée

set_resdef()
Fixe la résolution parmi les modes par défaut

set_sequence()
Fixe le nombre de photos à prendre en séquence

set_shutter(*event*)
Fixe le `shutter speed` de l'image

update_capture()
Actualise l'image de la dernière capture

update_etatVid()
Actualise l'état de l'enregistrement vidéo

update_nomFichier()
Actualise le nom du fichier

update_resactuelle()
Actualise l'affichage de la résolution et du champ de vision actuel

update_temps(*file=True*)
Actualise l'affichage du temps

verify_resH()
Vérifie si la résolution en hauteur entrée par l'utilisateur est d'un format adéquat `int`

verify_resW()
Vérifie si la résolution en largeur entrée par l'utilisateur est d'un format adéquat `int`

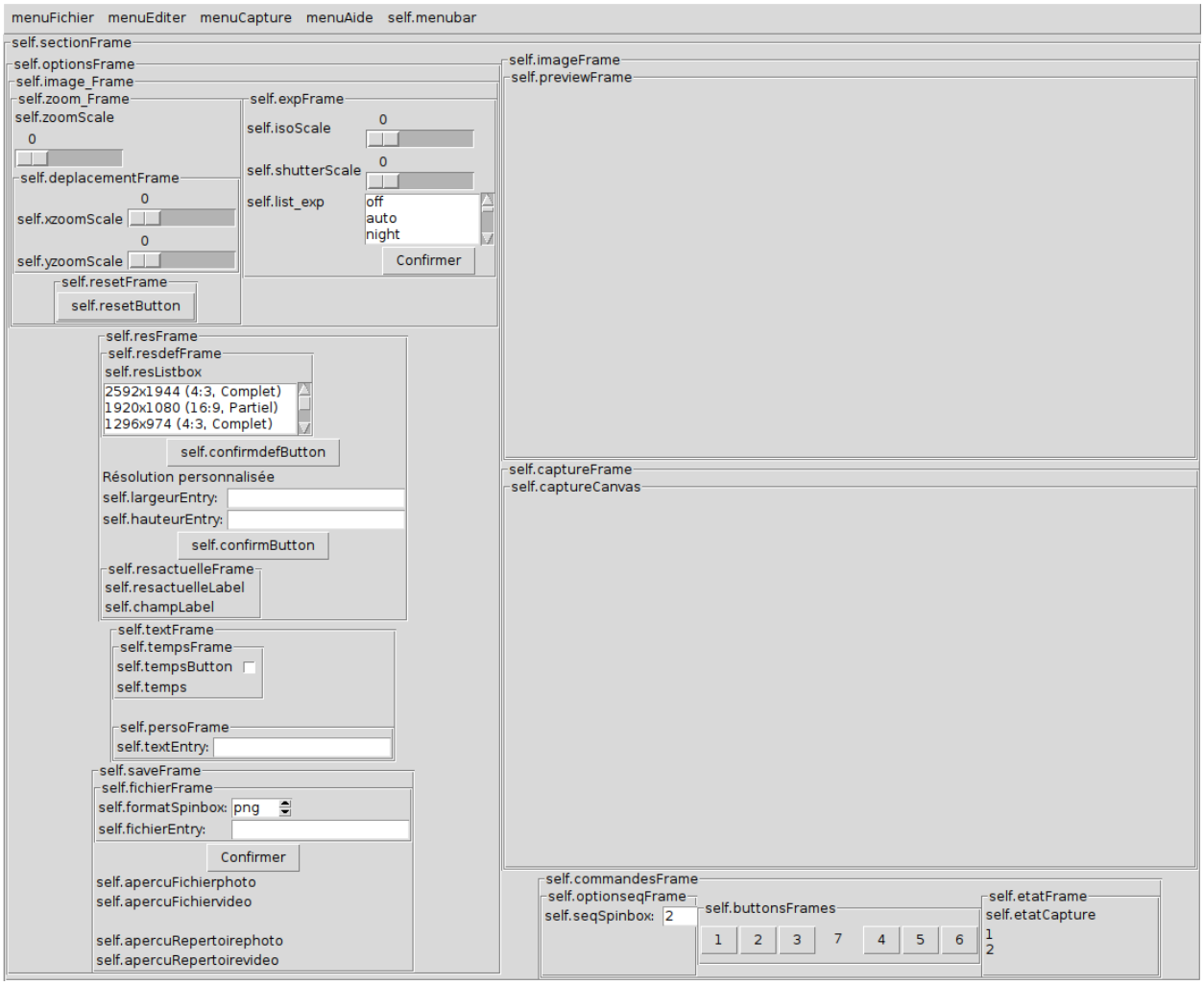
verify_text()
Vérifie et formatte les caractères invalides du texte personnalisé

- La fonction PiCamera.annotate_text() ne prend que les 128 premiers caractères du code ASCII
- Remplace les caractères invalides par `%`

vflip()
Effectue le revirement vertical de l'image

Disposition de l'interface

Si vous devez ajouter, modifier ou supprimer des fonctionnalités, vous devrez connaitre la disposition des objets dans la fenêtre. Voici l'architecture du logiciel. Les *Frames* agissent comme des fenêtres qui contiennent les boutons et autres widgets. Les canvas servent à contenir les images. Elles servent donc à l'organisation et à la disposition. Les `buttons`, `scales`, `entry`, `listbox`, `spinbox` servent à utiliser ou changer une fonctionnalité.



- 1: self.seqButton
- 2: self.photoButton
- 3: self.videoButton
- 4: self.hflipButton
- 5: self.rotateButton
- 6: self.vflipButton
- 7: self.etatCanvas

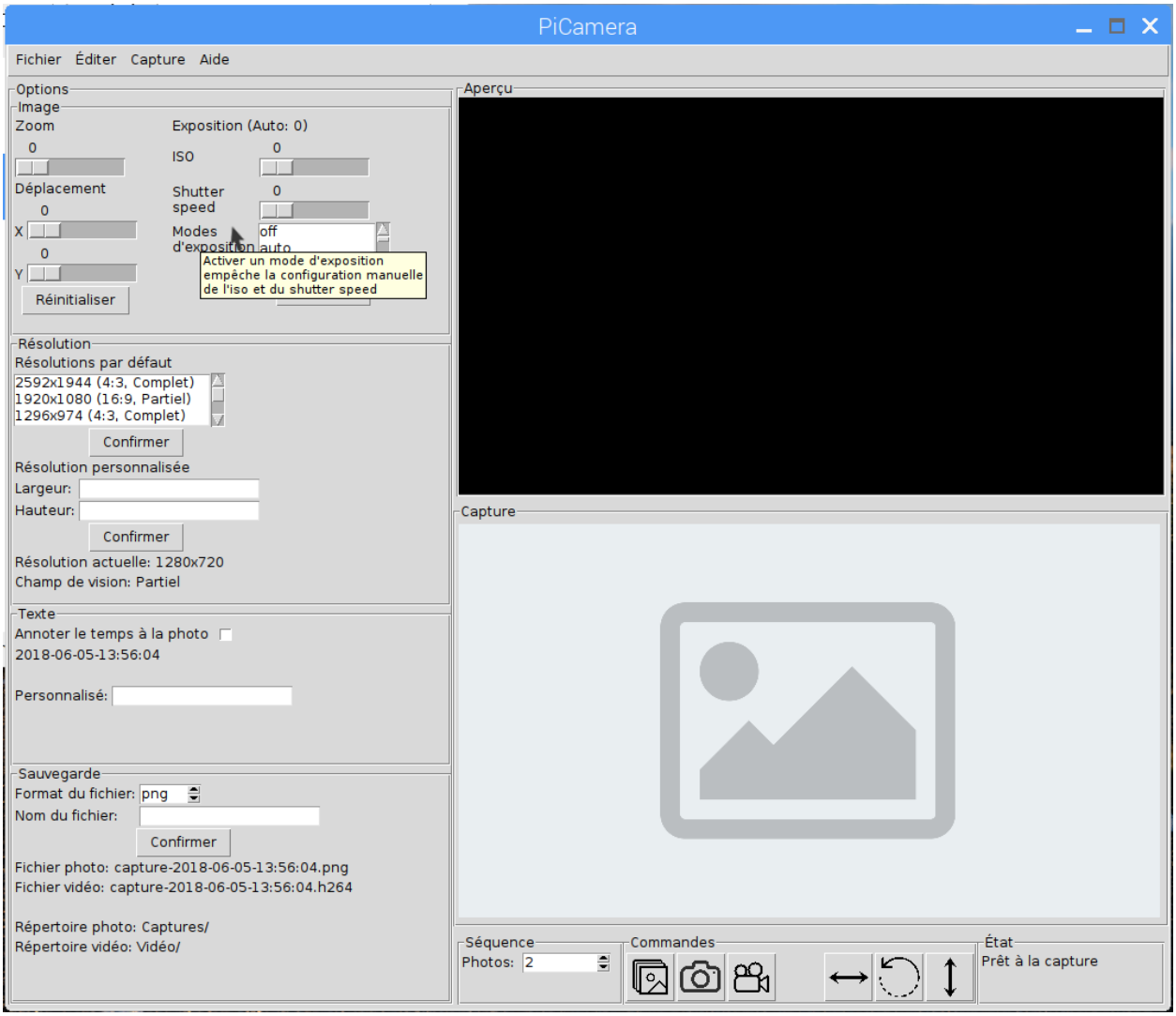
Call Tip Window

Ce package sert à afficher un label en survolant un objet Tkinter.

Par exemple, pour attacher un ToolTip au Label “self.modeLabel”, il suffit d’appeler la fonction “createToolTip” à partir du fichier CallTipWindow.py et de mettre le widget parent en premier argument et le texte à afficher en deuxième argument.:

```
from CallTipWindow import createToolTip

note = "Activer un mode d'exposition\nempêche la configuration manuelle
        \nde l'iso et du shutter speed"
createToolTip(self.modeLabel, note)
```



Note: L'auteur du programme PiCamera GUI ne détient pas les droits sur ce package et la license ne s'applique pas sur celui-ci. Les droits de ce package reviennent à Michael Foord.

Modules d'exception

Deux classes d'exceptions qui héritent de la classe Exception de la librairie standard de Python. Cela permet de créer deux nouveaux types d'erreurs: les PiCameraError TkinterError. Ces deux classes sont utilisées dans le fichier "Main.py". Si le logiciel ne détecte pas la PiCamera, il lèvera une PiCameraError et propose deux solutions:

- Débrancher et rebrancher la caméra
- Redémarrer le Rasperry Pi avec la caméra branchée

Si le logiciel ne peut importer les packages de la librairie Tkinter, il lèvera une TkinterError. Dans ce cas, il faut s'assurer d'avoir Python 3 sur le Raspberry Pi. Peut-être installer messagebox La création des deux classes sont basées sur le modèle suivant.

class Exceptions_ModuleCamera.**TkinterError**(*mismatch*)

class Exceptions_ModuleCamera.**PiCameraError**(*mismatch*)

Bibliographie

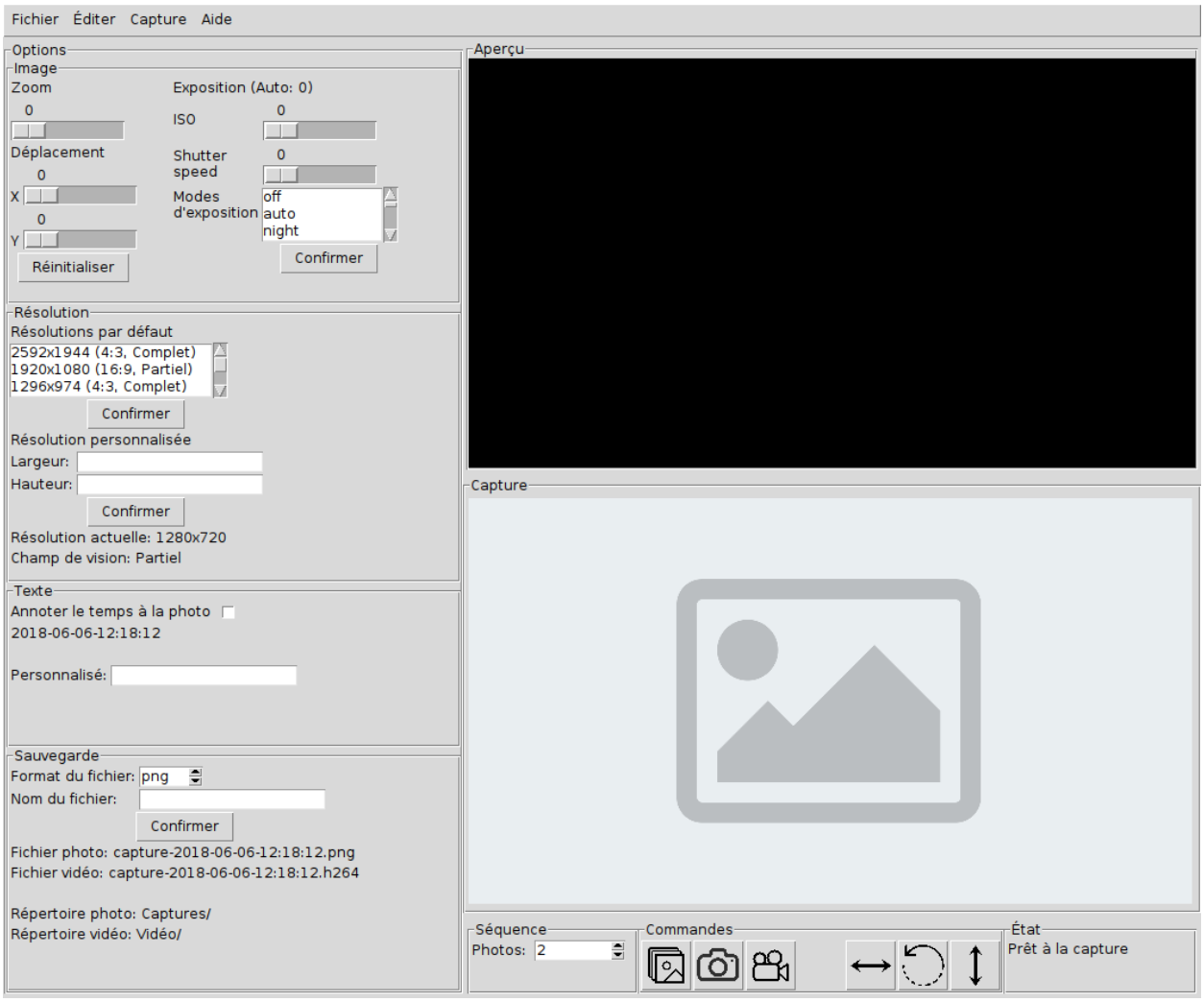
Vous trouverez ici toutes les documentations utilisées pour créer le logiciel.

Références

Droits d'auteurs

CHAPTER 4

Interface



CHAPTER 5

Indices and tables

- genindex
- modindex
- search

Index