ScolaSync

1.0

Généré par Doxygen 1.7.1

Mon Jun 13 2011 12 :37 :17

Table des matières

Chapitre 1

ScolaSync

1.1 But de l'application

Scolasync est un programme pour gérer des transferts de fichiers entre un ordinateur et une collection de clés USB.

1.2 CAHIER DE CHARGES DE SCOLASYNC

1. l'application doit pouvoir être utilisable par n'importe quel enseignant, par exemple un prof de langues quelques minutes après la prise en main.
2. une personne-essource, ou le prof lui-même, doit pouvoir très simplement créer une association permanente entre les identifiants des clés USB et les noms d'élèves. Cette association doit pouvoir évoluer en fonction des classes à la demande de l'enseignant, d'une année sur l'autre, ou d'un cycle de travail à un autre.
3. un prof doit pouvoir envoyer un ensemble de fichiers vers les clés USB de ses élèves identiquement pour tous. L'individualisation peut se faire en branchant/débranchant les clés. Le prof doit avoir la possibilité de choisir, voire de créer le dossier de réception.
4. chaque élève doit pouvoir retrouver facilement ces fichiers et surtout la consigne expliquant ce qu'il doit faire, et comment il sera noté. Comme les lecteurs mp3 stockent souvent des fichiers dans des répertoires de noms variés, il faut pouvoir gérer ça.
5. le prof doit pouvoir récolter les clés USB des élèves et récupérer leur travail en quelques minutes seulement, par exemple en sélectionnant le dossier dans lequel se trouve le fichier à récupérer.
6. l'application doit renommer les fichiers en tenant compte du nom du baladeur, donc du nom de l'élève.
7. il faut pouvoir effacer des fichiers sur les clés, voire les remettre à zéro.

1.3 Licence

ScolaSync version 1.0 :

un programme pour gérer des transferts de fichiers entre un ordinateur et une collection de clés USB.

Ce projet est un logiciel libre : vous pouvez le redistribuer, le modifier selon les terme de la GPL (GNU Public License) dans les termes de la Free Software Foundation concernant la version 3 ou plus de la dite licence.

Ce programme est fait avec l'espoir qu'il sera utile mais **SANS AUCUNE GARANTIE**. Lisez la licence pour plus de détails.

1.4 Support

Si vous avez besoin d'un support pour ce programme, tel que : **garantie contractuelle**, **formation**, **adaptation plus précise** aux besoins de votre entreprise, etc. contactez l'association OFSET et/ou l'auteur du logiciel.

1.5 Architecture de ScolaSync

Scolasync est bâti sur des composants logiciels libres, les plus notables sont les suivants :

- la bibliothèque Qt4 pour l'interface graphique
- la bibliothèque python-dbus pour l'interaction avec le noyau Linux 2.6
- la bibliothèque udisks pour interroger facilement le noyau sur le statut des disques, et pour réaliser certaines actions sur les disques et clés USB
- l'utilisation de threads pour mener en parallèle les actions qui concernent simultanément plusieurs clés USB

Chapitre 2

Hiérarchie de répertoires

2.1 Répertoires

Cette hiérarchie de répertoire est triée approximativement, mais pas complètement, par ordre alphabétique :

Chapitre 3

Index des espaces de nommage

3.1 Liste des paquetages

Liste des paquetages avec une brève description (si disponible) :

Chapitre 4

Index des classes

4.1 Hiérarchie des classes

Cette liste d'héritage est classée approximativement par ordre alphabétique :

Chapitre 5

Index des classes

5.1 Liste des classes

Liste des classes, structures, unions et interfaces avec une brève description :

Chapitre 6

Index des fichiers

6.1 Liste des fichiers

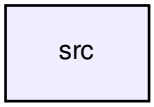
Liste de tous les fichiers avec une brève description :

Chapitre 7

Documentation des répertoires

7.1 Répertoire de référence de src/

Directory dependency graph for src/ :



Fichiers

- fichier chooseInSticks.py
- fichier copyToDialog1.py
- fichier db.py
- fichier diskFull.py
- fichier globaldef.py
- fichier help.py
- fichier mainWindow.py
- fichier marques.py
- fichier mytextbrowser.py
- fichier notification.py
- fichier ownedUsbDisk.py
- fichier preferences.py
- fichier scolasync.py
- fichier usbDisk.py
- fichier usbThread.py
- fichier version.py

Chapitre 8

Documentation des espaces de nommage

8.1 Paquetage chooseInSticks

Classes

– class chooseDialog
Un dialogue pour choisir un ensemble de fichiers à copier depuis une clé USB.

Variables

– string licenceEn

8.1.1 Documentation des variables

8.1.1.1 string chooseInSticks.licenceEn

Valeur initiale :

```
1 """
2     file chooseInSticks.py
3     this file is part of the project scolasync
4 
5     Copyright (C) 2010 Georges Khaznadar <georgesk@ofset.org>
6 
7     This program is free software: you can redistribute it and/or modify
8     it under the terms of the GNU General Public License as published by
9     the Free Software Foundation, either version3 of the License, or
10     (at your option) any later version.
11 
12     This program is distributed in the hope that it will be useful,
13     but WITHOUT ANY WARRANTY; without even the implied warranty of
14     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
15     GNU General Public License for more details.
16 
17     You should have received a copy of the GNU General Public License
18     along with this program.  If not, see <http://www.gnu.org/licenses/>.
19 """
```

Définition à la ligne 4 du fichier chooseInSticks.py.

8.2 Paquetage copyToDialog1

Classes

– class copyToDialog1
Un dialogue pour choisir un ensemble de fichiers à transférer vers une collection de clés USB.

Variables

– string licenceEn
– tuple app = QApplication(sys.argv)
– tuple windows = copyToDialog1()

8.2.1 Documentation des variables

8.2.1.1 tuple copyToDialog1.app = QApplication(sys.argv)

Définition à la ligne 209 du fichier copyToDialog1.py.

8.2.1.2 string copyToDialog1.licenceEn

Valeur initiale :

```
1 """
2     file copyToDialog1.py
3     this file is part of the project scolasync
4 
5     Copyright (C) 2010 Georges Khaznadar <georgesk@ofset.org>
6 
7     This program is free software: you can redistribute it and/or modify
8     it under the terms of the GNU General Public License as published by
9     the Free Software Foundation, either version3 of the License, or
10     (at your option) any later version.
11 
12     This program is distributed in the hope that it will be useful,
13     but WITHOUT ANY WARRANTY; without even the implied warranty of
14     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
15     GNU General Public License for more details.
16 
17     You should have received a copy of the GNU General Public License
18     along with this program.  If not, see <http://www.gnu.org/licenses/>.
19 """
```

Définition à la ligne 4 du fichier copyToDialog1.py.

8.2.1.3 tuple copyToDialog1.windows = copyToDialog1()

Définition à la ligne 210 du fichier copyToDialog1.py.

8.3 Paquetage db

Fonctions

– def openDb
Ouverture de la base de données de l'application, et création si nécessaire.

– def checkVersion
Vérifie si la base de données reste compatible.

– def knowsId
dit si une clé USB est déjà connue

– def tattooList
Renvoie la liste des tatouages connus de la base de données.

– def readStudent
renvoie l'étudiant qui possède une clé USB

– def readPrefs
renvoie les préférences de ScolaSync

– def setWd
définit le nouveau nom du répertoire de travail préféré.

– def writeStudent
inscrit un étudiant comme propriétaire d'une clé USB

– def writePrefs
inscrit les préférences

Variables

– dictionary licence = { }
– database = None
– cursor = None

8.3.1 Documentation des fonctions

8.3.1.1 def db.checkVersion (*major, minor*)

Vérifie si la base de données reste compatible.

Un changement de version majeur implique une mise à jour en cas de base de donnée ancienne. Un changmeent de version mineur n'implique pas de changement de structure de la base de données.

Définition à la ligne 57 du fichier db.py.

8.3.1.2 def db.knowsId (*stickid, uuid, tattoo*)

dit si une clé USB est déjà connue

Paramètres

stickid un identifiant de baladeur

uuid un identifiant de partition

tattoo un tatouage de partition

Renvoie

un booléen vrai si la clé USB est connue, faux sinon

Définition à la ligne 81 du fichier db.py.

8.3.1.3 def db.openDb ( )

Ouverture de la base de données de l'application, et création si nécessaire.

Renvoie

une instance de base de données sqlite3

Définition à la ligne 37 du fichier db.py.

8.3.1.4 def db.readPrefs ( )

renvoie les préférences de ScolaSync

Renvoie

un dictionnaire de préférences

Définition à la ligne 114 du fichier db.py.

8.3.1.5 def db.readStudent (*stickid, uuid, tattoo*)

renvoie l'étudiant qui possède une clé USB

Renvoie

un nom d'étudiant ou None si la clé est inconnue

Définition à la ligne 100 du fichier db.py.

8.3.1.6 def db.setWd (*newDir*)

définit le nouveau nom du répertoire de travail préféré.

Définition à la ligne 130 du fichier db.py.

8.3.1.7 def db.tattooList ( )

Renvoie la liste des tatouages connus de la base de données.

Définition à la ligne 90 du fichier db.py.

8.3.1.8 def db.writePrefs (*prefs*)

inscrit les préférences

Paramètres

prefs un dictionnaire {"checkable" : booléen vrai si on doit afficher des cases à cocher, "workdir" : le répertoire préféré pour les fichiers de travail}

Définition à la ligne 156 du fichier db.py.

8.3.1.9 def db.writeStudent (*stickid, uuid, tattoo, student*)

inscrit un étudiant comme propriétaire d'une clé USB

Paramètres

student un nom d'étudiant

Définition à la ligne 140 du fichier db.py.

8.3.2 Documentation des variables

8.3.2.1 db.cursor = None

Définition à la ligne 30 du fichier db.py.

8.3.2.2 db.database = None

Définition à la ligne 29 du fichier db.py.

8.3.2.3 dictionary db.licence = {}

Définition à la ligne 4 du fichier db.py.

8.4 Paquetage diskFull

Classes

– class mainWindow

Variables

– dictionary licence = { }

8.4.1 Documentation des variables

8.4.1.1 dictionary diskFull.licence = {}

Définition à la ligne 5 du fichier diskFull.py.

8.5 Paquetage globaldef

Variables

– string licenceEn
globaldef.py is part of the package scolasync.

– string userShareDir = "~/.scolasync"
– string logFileName = "~/.scolasync/scolasync.log"
– string markFileName = "~/.scolasync/marques.py"

8.5.1 Documentation des variables

8.5.1.1 string globaldef.licenceEn

Valeur initiale :

```
1 """
2     scolasync version %s:
3 
4     a program to manage file transfers between a computer and a collection
5     of USB sticks.
6 
7     Copyright (C) 2010 Georges Khaznadar <georgesk@ofset.org>
8 
9     This program is free software: you can redistribute it and/or modify
10     it under the terms of the GNU General Public License as published by
11     the Free Software Foundation, either version 3 of the License, or
12     (at your option) any later version.
13 
14     This program is distributed in the hope that it will be useful,
15     but WITHOUT ANY WARRANTY; without even the implied warranty of
16     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
17     GNU General Public License for more details.
18 
19     You should have received a copy of the GNU General Public License
20     along with this program.  If not, see <http://www.gnu.org/licenses/>.
21 """
```

globaldef.py is part of the package scolasync.

This module contains some definitions which can be reused globally in the application

Définition à la ligne 11 du fichier globaldef.py.

8.5.1.2 string globaldef : :logFileName = "~/.scolasync/scolasync.log"

Définition à la ligne 37 du fichier globaldef.py.

8.5.1.3 string globaldef : :markFileName = "~/.scolasync/marques.py"

Définition à la ligne 38 du fichier globaldef.py.

8.5.1.4 string globaldef : :userShareDir = "~/.scolasync"

Définition à la ligne 36 du fichier globaldef.py.

8.6 Paquetage help

Classes

– class helpWindow

Variables

– dictionary licence = { }

8.6.1 Documentation des variables

8.6.1.1 dictionary help.licence = {}

Définition à la ligne 5 du fichier help.py.

8.7 Paquetage mainWindow

Classes

– class mainWindow
– class usbTableModel
Un modèle de table pour des séries de clés USB.

– class CheckBoxDelegate
– class UsbDiskDelegate

Fonctions

– def firstdir
Renvoie le premier répertoire existant d'une liste de propositions.

– def CheckBoxRect

Variables

– dictionary licence = { }
– tuple globalDiskData = ownedUsbDisk.Available(True,access="firstFat")

8.7.1 Documentation des fonctions

8.7.1.1 def mainWindow.CheckBoxRect (*view_item_style_options*)

Définition à la ligne 484 du fichier mainWindow.py.

8.7.1.2 def mainWindow.firstdir (*l*)

Renvoie le premier répertoire existant d'une liste de propositions.

Paramètres

l la liste de propositions

Définition à la ligne 42 du fichier mainWindow.py.

8.7.2 Documentation des variables

8.7.2.1 tuple mainWindow.globalDiskData = ownedUsbDisk.Available(True,access="firstFat")

Définition à la ligne 35 du fichier mainWindow.py.

8.7.2.2 dictionary mainWindow.licence = {}

Définition à la ligne 5 du fichier mainWindow.py.

8.8 Paquetage marques

8.9 Paquetage mytextbrowser

Classes

– class myTextBrowser
Une classe qui ouvre Firefox quand on clique sur un lien externe.

Variables

– dictionary licence = { }

8.9.1 Documentation des variables

8.9.1.1 dictionary mytextbrowser.licence = {}

Définition à la ligne 5 du fichier mytextbrowser.py.

8.10 Paquetage notification

Classes

– class Notification
Une classe pour afficher des notifications à l'écran.

Variables

– dictionary licence = { }
– tuple notif

8.10.1 Documentation des variables

8.10.1.1 dictionary notification.licence = {}

Définition à la ligne 5 du fichier notification.py.

8.10.1.2 tuple notification.notif

Valeur initiale :

```
1 Notification(app_name="AppliTest",
2                          summary="Notification de test",
3                          body="Voici le corps de la notification",
4                          app_icon="/usr/share/pixmaps/vlc.png",
5                          expire_timeout=7000)
```

Définition à la ligne 75 du fichier notification.py.

8.11 Paquetage ownedUsbDisk

Classes

– class uDisk
une classe qui ajoute un nom de propriétaire aux disque USB, et qui en même temps ajoute des particularités selon le nom du vendeur et le modèle.

– class Available
Une classe qui fournit une collection de disques USB connectés, avec leurs propriétaires.

Fonctions

– def editRecord
édition de la base de données.

Variables

– dictionary licence = { }

8.11.1 Documentation des fonctions

8.11.1.1 def ownedUsbDisk.editRecord (*owd*, *student* = "")

édition de la base de données.

Paramètres

owd une instance de ownedUsbDisk

student nom de propriétaire pour la clé. Chaîne vide par défaut.

Définition à la ligne 43 du fichier ownedUsbDisk.py.

8.11.2 Documentation des variables

8.11.2.1 dictionary ownedUsbDisk.licence = {}

Définition à la ligne 4 du fichier ownedUsbDisk.py.

8.12 Paquetage preferences

Classes

– class preferenceWindow

Variables

– dictionary licence = { }

8.12.1 Documentation des variables

8.12.1.1 dictionary preferences.licence = {}

Définition à la ligne 5 du fichier preferences.py.

8.13 Paquetage scolasync

Scolasync est un programme pour gérer des transferts de fichiers entre un ordinateur et une collection de clés USB.

Fonctions

– def usage
affiche le mode d'emploi à la console

– def run
C'est la fonction principale.

Variables

– dictionary licence = { }
– string licenceEn
– string licenceFr

8.13.1 Description détaillée

Scolasync est un programme pour gérer des transferts de fichiers entre un ordinateur et une collection de clés USB.

8.13.2 Documentation des fonctions

8.13.2.1 def scolasync.run ( )

C'est la fonction principale.

Elle crée un fichier de journalisation vide, puis lance l'application interactive elle-même. Le fichier de journalisation est ${HOME}/.scolasync/scolasync.log sous Linux

Définition à la ligne 156 du fichier scolasync.py.

8.13.2.2 def scolasync.usage ( )

affiche le mode d'emploi à la console

Définition à la ligne 140 du fichier scolasync.py.

8.13.3 Documentation des variables

8.13.3.1 dictionary scolasync.licence = {}

Définition à la ligne 85 du fichier scolasync.py.

8.13.3.2 string scolasync.licenceEn

Valeur initiale :

```
1 """
2     scolasync version %s:
3 
4     a program to manage file transfers between a computer and a collection
5     of USB sticks.
6 
7     Copyright (C) 2010 Georges Khaznadar <georgesk@ofset.org>
8 
9     This program is free software: you can redistribute it and/or modify
10     it under the terms of the GNU General Public License as published by
11     the Free Software Foundation, either version 3 of the License, or
12     (at your option) any later version.
13 
14     This program is distributed in the hope that it will be useful,
15     but WITHOUT ANY WARRANTY; without even the implied warranty of
16     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
17     GNU General Public License for more details.
18 
19     You should have received a copy of the GNU General Public License
20     along with this program.  If not, see <http://www.gnu.org/licenses/>.
21 """
```

Définition à la ligne 86 du fichier scolasync.py.

8.13.3.3 string scolasync.licenceFr

Valeur initiale :

```
1 """
2     scolasync version %s :
```

```
3
4     un programme pour gérer des transferts de fichiers entre un
5     ordinateur et une collection de clés USB.
6
7     Copyright (C) 2010 Georges Khaznadar <georgesk@ofset.org>
8
9     Ce projet est un logiciel libre : vous pouvez le redistribuer, le
10     modifier selon les terme de la GPL (GNU Public License) dans les
11     termes de la Free Software Foundation concernant la version 3 ou
12     plus de la dite licence.
13
14     Ce programme est fait avec l'espoir qu'il sera utile mais SANS
15     AUCUNE GARANTIE. Lisez la licence pour plus de détails.
16
17     <http://www.gnu.org/licenses/>.
18 """
```

Définition à la ligne 109 du fichier scolasync.py.

8.14 Paquetage usbDisk

Classes

– class uDisk
une classe pour représenter un disque ou une partition.

– class Available
une classe pour représenter la collection des disques USB connectés

Variables

– dictionary licence = { }
– string licence_en
– tuple machin = Available()

8.14.1 Documentation des variables

8.14.1.1 dictionary usbDisk.licence = {}

Définition à la ligne 4 du fichier usbDisk.py.

8.14.1.2 string usbDisk.licence_en

Valeur initiale :

```
1 """
2     file usbDisk.py
3     this file is part of the project scolasync
4
5     Copyright (C) 2010 Georges Khaznadar <georgesk@ofset.org>
6
7     This program is free software: you can redistribute it and/or modify
8     it under the terms of the GNU General Public License as published by
9     the Free Software Foundation, either version3 of the License, or
10     (at your option) any later version.
11
```

```
12     This program is distributed in the hope that it will be useful,
13     but WITHOUT ANY WARRANTY; without even the implied warranty of
14     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
15     GNU General Public License for more details.
16
17     You should have received a copy of the GNU General Public License
18     along with this program.  If not, see <http://www.gnu.org/licenses/>.
19 """
```

Définition à la ligne 5 du fichier usbDisk.py.

8.14.1.3 tuple usbDisk.machin = Available()

Définition à la ligne 507 du fichier usbDisk.py.

8.15 Paquetage usbThread

Classes

– class ThreadRegister
Une classe pour tenir un registre des threads concernant les baladeurs.

– class abstractThreadUSB
Une classe abstraite Cette classe sert de creuset pour les classe servant aux copies et aux effacement.

– class threadCopyToUSB
Classe pour les threads copiant vers les clés USB.

– class threadCopyFromUSB
Classe pour les threads copiant depuis les clés USB.

– class threadDeleteInUSB
Classe pour les threads effaçant des sous-arbres dans les clés USB.

Variables

– string licenceEn
– int _threadNumber = 0

8.15.1 Documentation des variables

8.15.1.1 int usbThread._threadNumber = 0

Définition à la ligne 26 du fichier usbThread.py.

8.15.1.2 string usbThread.licenceEn

Valeur initiale :

```
1 """
2     file usbThread.py
3     this file is part of the project scolasync
4
```

```
5     Copyright (C) 2010 Georges Khaznadar <georgesk@ofset.org>
6
7     This program is free software: you can redistribute it and/or modify
8     it under the terms of the GNU General Public License as published by
9     the Free Software Foundation, either version3 of the License, or
10     (at your option) any later version.
11
12     This program is distributed in the hope that it will be useful,
13     but WITHOUT ANY WARRANTY; without even the implied warranty of
14     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
15     GNU General Public License for more details.
16
17     You should have received a copy of the GNU General Public License
18     along with this program.  If not, see <http://www.gnu.org/licenses/>.
19 """
```

Définition à la ligne 4 du fichier usbThread.py.

8.16 Paquetage version

Fonctions

– def major
– def minor
– def version

Variables

– dictionary licence = { }

8.16.1 Documentation des fonctions

8.16.1.1 def version.major ( )

Renvoie

le numéro majeur de version

Définition à la ligne 30 du fichier version.py.

8.16.1.2 def version.minor ( )

Renvoie

le numéro mineur de version

Définition à la ligne 37 du fichier version.py.

8.16.1.3 def version.version ( )

Renvoie

l'identifiant de la version

Définition à la ligne 44 du fichier version.py.

8.16.2 Documentation des variables

8.16.2.1 dictionary version.licence = {}

Définition à la ligne 4 du fichier version.py.

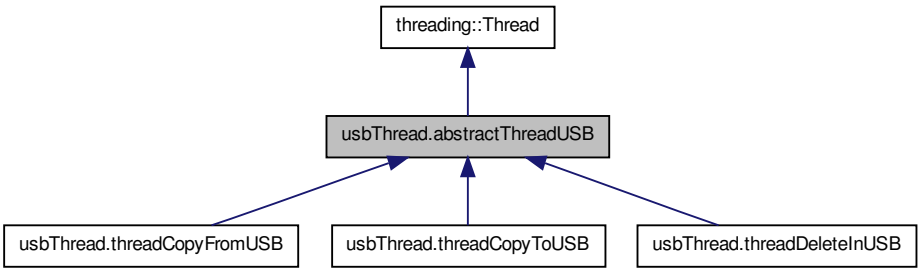
Chapitre 9

Documentation des classes

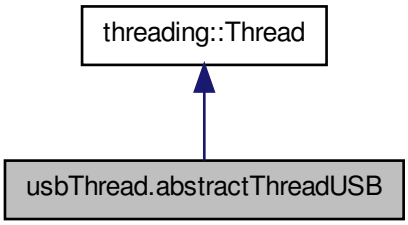
9.1 Référence de la classe usbThread.abstractThreadUSB

Une classe abstraite Cette classe sert de creuset pour les classe servant aux copies et aux effacement.

Graphe d'héritage de usbThread.abstractThreadUSB :



Graphe de collaboration de usbThread.abstractThreadUSB :



Fonctions membres publiques

– def __init__
Constructeur Crée un thread pour copier une liste de fichiers vers une clé USB.

– def __str__
Renvoie une chaîne informative sur le thread.

– def threadType
– def toDo
La fonction abstraite pour les choses à faire.

Attributs publics

– cmd
– ud
– fileList
– subdir
– dest
– logfile

9.1.1 Description détaillée

Une classe abstraite Cette classe sert de creuset pour les classe servant aux copies et aux effacement.

Définition à la ligne 133 du fichier usbThread.py.

9.1.2 Documentation des fonctions membres

9.1.2.1 def usbThread.abstractThreadUSB.__init__ (*self, ud, fileList, subdir, dest* = `None`, *logfile* = `"/dev/null"`)

Constructeur Crée un thread pour copier une liste de fichiers vers une clé USB.

Paramètres

ud l'instance uDisk correspondant à une partition de clé USB

fileList la liste des fichiers à traiter

subdir un sous-répertoire de la clé USB

dest un répertoire de destination si nécessaire, None par défaut

logfile un fichier de journalisation, /dev/null par défaut

Définition à la ligne 144 du fichier usbThread.py.

9.1.2.2 def usbThread.abstractThreadUSB.__str__ (*self*)

Renvoie une chaîne informative sur le thread.

Renvoie

une chaine donnant des informations sur ce qui va se passer dans le thread qui a été créé.

Définition à la ligne 162 du fichier usbThread.py.

9.1.2.3 def usbThread.abstractThreadUSB.threadType (*self*)

Renvoie

une chaîne courte qui informe sur le type de thread

Réimplémentée dans usbThread.threadCopyToUSB.

Définition à la ligne 177 du fichier usbThread.py.

9.1.2.4 def usbThread.abstractThreadUSB.toDo (*self, ud, fileList, subdir, dest, logfile*)

La fonction abstraite pour les choses à faire.

Paramètres

ud l'instance uDisk correspondant à une partition de clé USB

fileList la liste des fichiers à traiter

subdir un sous-répertoire de la clé USB

dest un répertoire de destination

logfile un fichier de journalisation

Réimplémentée dans usbThread.threadCopyToUSB, usbThread.threadCopyFromUSB, et usbThread.threadDeleteInUSB.

Définition à la ligne 189 du fichier usbThread.py.

9.1.3 Documentation des données membres

9.1.3.1 usbThread.abstractThreadUSB.cmd

Réimplémentée dans usbThread.threadCopyToUSB, usbThread.threadCopyFromUSB, et usbThread.threadDeleteInUSB.

Définition à la ligne 148 du fichier usbThread.py.

9.1.3.2 usbThread.abstractThreadUSB.dest

Définition à la ligne 153 du fichier usbThread.py.

9.1.3.3 usbThread.abstractThreadUSB.fileList

Définition à la ligne 151 du fichier usbThread.py.

9.1.3.4 usbThread.abstractThreadUSB.logfile

Définition à la ligne 154 du fichier usbThread.py.

9.1.3.5 usbThread.abstractThreadUSB.subdir

Définition à la ligne 152 du fichier usbThread.py.

9.1.3.6 usbThread.abstractThreadUSB.ud

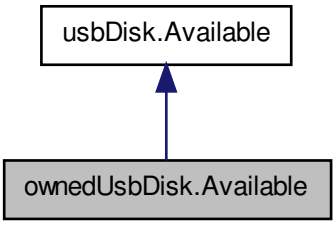
Définition à la ligne 149 du fichier usbThread.py.

La documentation de cette classe a été générée à partir du fichier suivant :
– src/usbThread.py

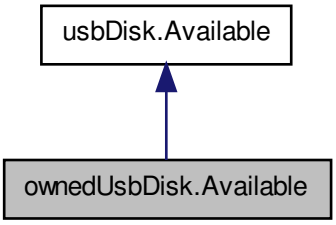
9.2 Référence de la classe ownedUsbDisk.Available

Une classe qui fournit une collection de disques USB connectés, avec leurs propriétaires.

Graphe d'héritage de ownedUsbDisk.Available :



Graphe de collaboration de ownedUsbDisk.Available :



Fonctions membres publiques

– def __init__

Attributs publics

– checkable
– access
– bus
– disks
– enumDev
– firstFats

9.2.1 Description détaillée

Une classe qui fournit une collection de disques USB connectés, avec leurs propriétaires. On a répliqué le code de la classe parente, ne sachant pas si le contructeur prendrait les uDisks dans le module courant ou dans le module parent. Pour éviter les confusion il faudrait peut-être faire une classe abstraite Available

Définition à la ligne 226 du fichier ownedUsbDisk.py.

9.2.2 Documentation des fonctions membres

9.2.2.1 def ownedUsbDisk.Available.__init__ (*self*, *checkable* = `False`, *access* = `"disk"`)

Paramètres

checkable : vrai si on veut pouvoir cocher les disques de la collection. Faux par défaut.

access définit le type d'accès souhaité. Par défaut, c'est "disk" c'est à dire qu'on veut la liste des disques USB. Autres valeurs possibles : "firstFat" pour les premières partitions vfat.

Réimplémentée à partir de usbDisk.Available.

Définition à la ligne 235 du fichier ownedUsbDisk.py.

9.2.3 Documentation des données membres

9.2.3.1 ownedUsbDisk.Available.access

Réimplémentée à partir de usbDisk.Available.

Définition à la ligne 237 du fichier ownedUsbDisk.py.

9.2.3.2 ownedUsbDisk.Available.bus

Réimplémentée à partir de usbDisk.Available.

Définition à la ligne 238 du fichier ownedUsbDisk.py.

9.2.3.3 ownedUsbDisk.Available.checkable

Réimplémentée à partir de usbDisk.Available.

Définition à la ligne 236 du fichier ownedUsbDisk.py.

9.2.3.4 ownedUsbDisk.Available.disks

Réimplémentée à partir de usbDisk.Available.

Définition à la ligne 242 du fichier ownedUsbDisk.py.

9.2.3.5 ownedUsbDisk.Available.enumDev

Réimplémentée à partir de usbDisk.Available.

Définition à la ligne 243 du fichier ownedUsbDisk.py.

9.2.3.6 ownedUsbDisk.Available.firstFats

Réimplémentée à partir de usbDisk.Available.

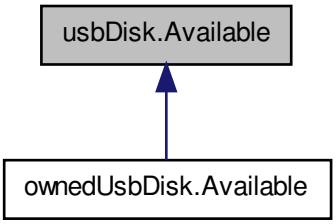
Définition à la ligne 261 du fichier ownedUsbDisk.py.

La documentation de cette classe a été générée à partir du fichier suivant :
– src/ownedUsbDisk.py

9.3 Référence de la classe usbDisk.Available

une classe pour représenter la collection des disques USB connectés

Graphe d'héritage de usbDisk.Available :



Fonctions membres publiques

– def __init__
Le constructeur.

– def compare
Sert à comparer deux collections de disques, par exemple une collection passée et une collection présente.

– def contains
Permet de déterminer si un disque est dans la collection.

– def summary
Fournit une représentation imprimable d'un résumé.

– def __str__
Fournit une représentation imprimable.

– def __getitem__
Renvoye le nième disque.

– def __len__
Renseigne sur la longueur de la collection.

– def getFirstFats
Facilite l'accès aux partitions de type DOS-FAT, et a un effet de bord : marque le disque avec l'uuid de la première partition FAT.

Attributs publics

– checkable
– access
– bus
– disks
– enumDev
– firstFats

9.3.1 Description détaillée

une classe pour représenter la collection des disques USB connectés les attributs publics sont :
– **checkable** booléen vrai si on veut gérer des sélections de disques
– **access** le type d'accès qu'on veut pour les items
– **bus** une instance de dbus.SystemBus
– **disks** la collection de disques USB, organisée en un dictionnaire de disques : les clés sont les disques, qui renvoient à un ensemble de partitions du disque
– **enumdev** une liste de chemins dbus vers les disques trouvés
– **firstFats** une liste composée de la première partion DOS-FAT de chaque disque USB.
Définition à la ligne 360 du fichier usbDisk.py.

9.3.2 Documentation des fonctions membres

9.3.2.1 def usbDisk.Available.__getitem__ (*self, n*)

Renvoye le nième disque.

Le fonctionnement dépend du paramètre self.access
Paramètres

n un numéro

Renvoie

le nième disque USB connecté

Définition à la ligne 465 du fichier usbDisk.py.

9.3.2.2 def usbDisk.Available.__init__ (*self, checkable* = `False`, *access* = `"disk"`)

Le constructeur.

Paramètres

checkable : vrai si on veut pouvoir cocher les disques de la collection. Faux par défaut.

access définit le type d'accès souhaité. Par défaut, c'est "disk" c'est à dire qu'on veut la liste des disques USB. Autres valeurs possibles : "firstFat" pour les premières partitions vfat.

Réimplémentée dans ownedUsbDisk.Available.

Définition à la ligne 371 du fichier usbDisk.py.

9.3.2.3 def usbDisk.Available.__len__ (*self*)

Renseigne sur la longueur de la collection.

Le fonctionnement dépend du paramètre self.access

Renvoie

la longueur de la collection de disques renvoyée

Définition à la ligne 477 du fichier usbDisk.py.

9.3.2.4 def usbDisk.Available.__str__ (*self*)

Fournit une représentation imprimable.

Renvoie

une représentation imprimable de la collection

Définition à la ligne 446 du fichier usbDisk.py.

9.3.2.5 def usbDisk.Available.compare (*self, other*)

Sert à comparer deux collections de disques, par exemple une collection passée et une collection présente.

Paramètres

other une instance de Available

Renvoie

vrai si other semble être la même collection de disques USB

Définition à la ligne 410 du fichier usbDisk.py.

9.3.2.6 def usbDisk.Available.contains (*self, ud*)

Permet de déterminer si un disque est dans la collection.

Paramètres

ud une instance de uDisk

Renvoie

vrai si le uDisk ud est dans la collection

Définition à la ligne 420 du fichier usbDisk.py.

9.3.2.7 def usbDisk.Available.getFirstFats (*self*)

Facilite l'accès aux partitions de type DOS-FAT, et a un effet de bord : marque le disque avec l'uuid de la première partition FAT.

Renvoie

une liste de partitions, constituée de la première partition de type FAT de chaque disque USB connecté

Définition à la ligne 490 du fichier usbDisk.py.

9.3.2.8 def usbDisk.Available.summary (*self*)

Fournit une représentation imprimable d'un résumé.

Renvoie

une représentation imprimable d'un résumé de la collection

Définition à la ligne 430 du fichier usbDisk.py.

9.3.3 Documentation des données membres

9.3.3.1 usbDisk.Available.access

Réimplémentée dans ownedUsbDisk.Available.

Définition à la ligne 373 du fichier usbDisk.py.

9.3.3.2 usbDisk.Available.bus

Réimplémentée dans ownedUsbDisk.Available.

Définition à la ligne 374 du fichier usbDisk.py.

9.3.3.3 usbDisk.Available.checkable

Réimplémentée dans ownedUsbDisk.Available.

Définition à la ligne 372 du fichier usbDisk.py.

9.3.3.4 usbDisk.Available.disks

Réimplémentée dans ownedUsbDisk.Available.

Définition à la ligne 378 du fichier usbDisk.py.

9.3.3.5 usbDisk.Available.enumDev

Réimplémentée dans ownedUsbDisk.Available.

Définition à la ligne 379 du fichier usbDisk.py.

9.3.3.6 usbDisk.Available.firstFats

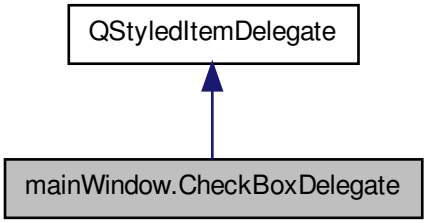
Réimplémentée dans ownedUsbDisk.Available.

Définition à la ligne 397 du fichier usbDisk.py.

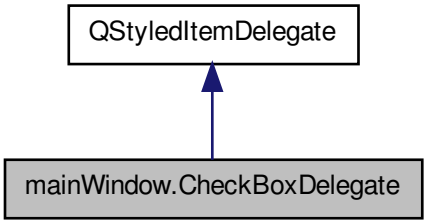
La documentation de cette classe a été générée à partir du fichier suivant :
– src/usbDisk.py

9.4 Référence de la classe mainWindow.CheckBoxDelegate

Graphe d'héritage de mainWindow.CheckBoxDelegate :



Graphe de collaboration de mainWindow.CheckBoxDelegate :



Fonctions membres publiques

– def __init__
– def paint
– def editorEvent

9.4.1 Description détaillée

Définition à la ligne 490 du fichier mainWindow.py.

9.4.2 Documentation des fonctions membres

9.4.2.1 def mainWindow.CheckBoxDelegate.__init__ (*self, parent*)

Définition à la ligne 491 du fichier mainWindow.py.

9.4.2.2 def mainWindow.CheckBoxDelegate.editorEvent (*self, event, model, option, index*)

Définition à la ligne 505 du fichier mainWindow.py.

9.4.2.3 def mainWindow.CheckBoxDelegate.paint (*self, painter, option, index*)

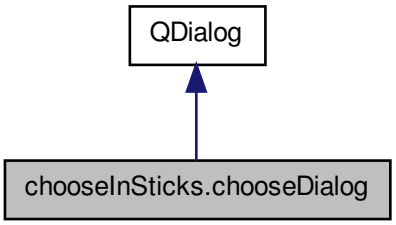
Définition à la ligne 494 du fichier mainWindow.py.

La documentation de cette classe a été générée à partir du fichier suivant :
– src/mainWindow.py

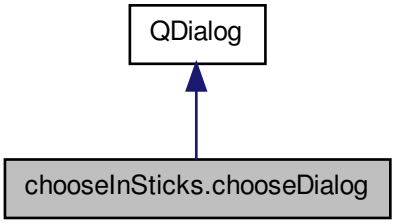
9.5 Référence de la classe chooseInSticks.chooseDialog

Un dialogue pour choisir un ensemble de fichiers à copier depuis une clé USB.

Graphe d'héritage de chooseInSticks.chooseDialog :



Graphe de collaboration de chooseInSticks.chooseDialog :



Fonctions membres publiques

– def __init__
Le constructeur.

– def listStorages
Met en place la liste des noms de baladeurs connectés en tenant compte du nom de répertoire de travail et d'un baladeur éventuellement sélectionné dans la fenêtre principale.

– def checkWorkDirs
met à jour la possibilité de sélectionner les baladeurs dans la liste selon qu'ils ont ou pas un répertoire de travail, puis sélectionne si possible un baladeur, si aucun ne l'était avant.

– def baseDir
– def selectedDiskMountPoint
– def selectedDiskOwner
– def changeWd
changement du répertoire de travail

– def choose
Facilite le choix de motifs de fichiers en recherchant dans les clés USB, modifie l'éditeur de ligne de texte et place le fichier choisi dans la liste.

– def choose_dir
Facilite le choix de motifs de répertoires en recherchant dans les clés USB, modifie l'éditeur de ligne de texte et place le répertoire choisi dans la liste.

– def activate
Fonction de rappel quand un item de la liste est activé.

– def plus
Permet de choisir et d'ajouter un nouveau fichier ou répertoire à supprimer.

– def minus
Permet de retirer de la liste des fichiers à supprimer ceux qu'on a sélectionnés.

– def append
Ajoute un chemin avec ou sans jokers à la liste des chemins à supprimer.

– def pathList
renvoie la liste des chemins sélectionés

Attributs publics

– mainWindow
– ownedUsbDictionary
peuplement de la zone des noms de baladeurs

9.5.1 Description détaillée

Un dialogue pour choisir un ensemble de fichiers à copier depuis une clé USB.

Définition à la ligne 34 du fichier chooseInSticks.py.

9.5.2 Documentation des fonctions membres

9.5.2.1 def chooseInSticks.chooseDialog.__init__ (*self, parent* = `None`, *title1* = `""`, *title2* = `""`, *ok* = `"OK"`)

Le constructeur.

Paramètres

parent un mainWindow, qui est censé contenir des données telles que parent.workdir, ...

title1 le titre du dialogue

title2 le titre pour la série de fichiers/modèles

ok le texte du bouton OK

Définition à la ligne 44 du fichier chooseInSticks.py.

9.5.2.2 def chooseInSticks.chooseDialog.activate (*self, item*)

Fonction de rappel quand un item de la liste est activé.

Paramètres

item désignation de l'item activé

Définition à la ligne 238 du fichier chooseInSticks.py.

9.5.2.3 def chooseInSticks.chooseDialog.append (*self, path*)

Ajoute un chemin avec ou sans jokers à la liste des chemins à supprimer.

Paramètres

path le chemin

Définition à la ligne 272 du fichier chooseInSticks.py.

9.5.2.4 def chooseInSticks.chooseDialog.baseDir (*self*)

Renvoie

le répertoire à partir duquel on peut commencer à faire un choix de fichier ou de sous-répertoire. Il dépend du baladeur sélectionné s'il y en a un et du nom du répertoire de travail. Si on n'arrive pas à déterminer ce répertoire, renvoie None

Définition à la ligne 148 du fichier chooseInSticks.py.

9.5.2.5 def chooseInSticks.chooseDialog.changeWd (*self*)

changement du répertoire de travail

Définition à la ligne 182 du fichier chooseInSticks.py.

9.5.2.6 def chooseInSticks.chooseDialog.checkWorkDirs (*self*)

met à jour la possibilité de sélectionner les baladeurs dans la liste selon qu'ils ont ou pas un répertoire de travail, puis sélectionne si possible un baladeur, si aucun ne l'était avant.

Définition à la ligne 111 du fichier chooseInSticks.py.

9.5.2.7 def chooseInSticks.chooseDialog.choose (*self, kind* = `"file"`)

Facilite le choix de motifs de fichiers en recherchant dans les clés USB, modifie l'éditeur de ligne de texte et place le fichier choisi dans la liste.

Paramètres

kind type d'élément à choisir : "file" pour un fichier, "dir" pour un répertoire

Définition à la ligne 195 du fichier chooseInSticks.py.

9.5.2.8 def chooseInSticks.chooseDialog.choose_dir (*self*)

Facilite le choix de motifs de répertoires en recherchant dans les clés USB, modifie l'éditeur de ligne de texte et place le répertoire choisi dans la liste.

Définition à la ligne 230 du fichier chooseInSticks.py.

9.5.2.9 def chooseInSticks.chooseDialog.listStorages (*self*)

Met en place la liste des noms de baladeurs connectés en tenant compte du nom de répertoire de travail et d'un baladeur éventuellement sélectionné dans la fenêtre principale.

Définition à la ligne 89 du fichier chooseInSticks.py.

9.5.2.10 def chooseInSticks.chooseDialog.minus (*self*)

Permet de retirer de la liste des fichiers à supprimer ceux qu'on a sélectionnés.

Définition à la ligne 256 du fichier chooseInSticks.py.

9.5.2.11 def chooseInSticks.chooseDialog.pathList (*self*)

renvoie la liste des chemins sélectionés

Renvoie

une liste de chemins, sous forme de QStrings

Définition à la ligne 286 du fichier chooseInSticks.py.

9.5.2.12 def chooseInSticks.chooseDialog.plus (*self*)

Permet de choisir et d'ajouter un nouveau fichier ou répertoire à supprimer.

Définition à la ligne 246 du fichier chooseInSticks.py.

9.5.2.13 def chooseInSticks.chooseDialog.selectedDiskMountPoint (*self*)

Renvoie

le point de montage du support sélectionné s'il y en a un

Définition à la ligne 159 du fichier chooseInSticks.py.

9.5.2.14 def chooseInSticks.chooseDialog.selectedDiskOwner (*self*)

Renvoie

le nom du propriétaire du disque sélectionné s'il y en a un, sinon None.

Définition à la ligne 171 du fichier chooseInSticks.py.

9.5.3 Documentation des données membres

9.5.3.1 chooseInSticks.chooseDialog.mainWindow

Définition à la ligne 46 du fichier chooseInSticks.py.

9.5.3.2 chooseInSticks.chooseDialog.ownedUsbDictionary

peuplement de la zone des noms de baladeurs

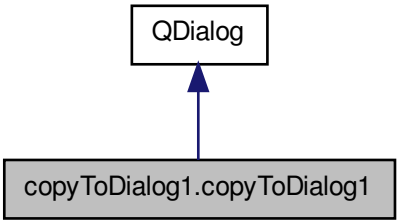
Définition à la ligne 70 du fichier chooseInSticks.py.

La documentation de cette classe a été générée à partir du fichier suivant :
– src/chooseInSticks.py

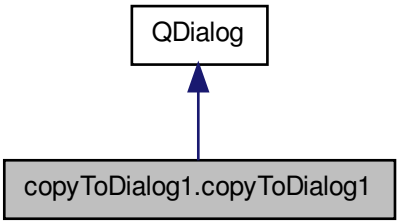
9.6 Référence de la classe copyToDialog1.copyToDialog1

Un dialogue pour choisir un ensemble de fichiers à transférer vers une collection de clés USB.

Graphe d'héritage de copyToDialog1.copyToDialog1 :



Graphe de collaboration de copyToDialog1.copyToDialog1 :



Fonctions membres publiques

– def __init__
Le constructeur.

– def changeWd
changement du répertoire de travail

– def cancel
L'action provoquée par le bouton d'échappement : fermeture du dialogue.

– def cont
L'action provoquée par le bouton de continuation : fermeture du dialogue et self.ok devient vrai.

– def setupFromListe
Met en place un visionneur de fichiers dans la liste source.

– def setFromListeDir
Choisit un répertoire pour la liste source.

– def cd

Change le répertoire courant si possible.

– def setupToListe
Met en place un visionneur de fichierspour les fichiers reçus.

– def select
Ajoute le répertoire ou le fichier sélectionné dans le navigateur de fichiers à la liste de sélections.

– def displaySize
Affiche la taille de la sélection courante.

– def remove
Supprime le répertoire ou le fichier sélectionné dans la liste de sélections.

– def selectedList
Renvoie une liste de répertoires et de fichiers qui ont été sélectionnés pour la copie sur clé USB.

Attributs publics

– mainWindow
– ok

9.6.1 Description détaillée

Un dialogue pour choisir un ensemble de fichiers à transférer vers une collection de clés USB.

Paramètres

parent un widget

workdir un répertoire cible sur les baladeurs

Définition à la ligne 37 du fichier copyToDialog1.py.

9.6.2 Documentation des fonctions membres

9.6.2.1 def copyToDialog1.copyToDialog1.__init__ (*self, parent* = `None`, *workdir* = `""`)

Le constructeur.

Paramètres

parent un QWidget

Définition à la ligne 43 du fichier copyToDialog1.py.

9.6.2.2 def copyToDialog1.copyToDialog1.cancel (*self*)

L'action provoquée par le bouton d'échappement : fermeture du dialogue.

Définition à la ligne 74 du fichier copyToDialog1.py.

9.6.2.3 def copyToDialog1.copyToDialog1.cd (*self, index*)

Change le répertoire courant si possible.

Paramètres

ev un évènement

Définition à la ligne 112 du fichier copyToDialog1.py.

9.6.2.4 def copyToDialog1.copyToDialog1.changeWd (*self*)

changement du répertoire de travail

Définition à la ligne 66 du fichier copyToDialog1.py.

9.6.2.5 def copyToDialog1.copyToDialog1.cont (*self*)

L'action provoquée par le bouton de continuation : fermeture du dialogue et self.ok devient vrai.

Définition à la ligne 82 du fichier copyToDialog1.py.

9.6.2.6 def copyToDialog1.copyToDialog1.displaySize (*self*)

Affiche la taille de la sélection courante.

Définition à la ligne 163 du fichier copyToDialog1.py.

9.6.2.7 def copyToDialog1.copyToDialog1.remove (*self*)

Supprime le répertoire ou le fichier sélectionné dans la liste de sélections.

Définition à la ligne 187 du fichier copyToDialog1.py.

9.6.2.8 def copyToDialog1.copyToDialog1.select (*self*)

Ajoute le répertoire ou le fichier sélectionné dans le navigateur de fichiers à la liste de sélections.

Définition à la ligne 143 du fichier copyToDialog1.py.

9.6.2.9 def copyToDialog1.copyToDialog1.selectedList (*self*)

Renvoie une liste de répertoires et de fichiers qui ont été sélectionnés pour la copie sur clé USB.

Renvoie

une liste de QStrings

Définition à la ligne 203 du fichier copyToDialog1.py.

9.6.2.10 def copyToDialog1.copyToDialog1.setFromListeDir (*self, directory*)

Choisit un répertoire pour la liste source.

Paramètres

directory une instance de QDir

Définition à la ligne 101 du fichier copyToDialog1.py.

9.6.2.11 def copyToDialog1.copyToDialog1.setupFromListe (*self*)

Met en place un visionneur de fichiers dans la liste source.

Définition à la ligne 90 du fichier copyToDialog1.py.

9.6.2.12 def copyToDialog1.copyToDialog1.setupToListe (*self*)

Met en place un visionneur de fichierspour les fichiers reçus.

Définition à la ligne 124 du fichier copyToDialog1.py.

9.6.3 Documentation des données membres

9.6.3.1 copyToDialog1.copyToDialog1.mainWindow

Définition à la ligne 45 du fichier copyToDialog1.py.

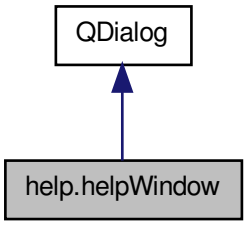
9.6.3.2 copyToDialog1.copyToDialog1.ok

Définition à la ligne 55 du fichier copyToDialog1.py.

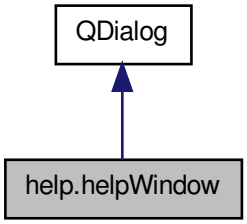
La documentation de cette classe a été générée à partir du fichier suivant :
– src/copyToDialog1.py

9.7 Référence de la classe help.helpWindow

Graphe d'héritage de help.helpWindow :



Graphe de collaboration de help.helpWindow :



Fonctions membres publiques

– def __init__
Le constructeur.

– def loadBrowsers
met en place les textes dans les afficheurs, en fonction de la locale.

Attributs publics

– ui

9.7.1 Description détaillée

Définition à la ligne 30 du fichier help.py.

9.7.2 Documentation des fonctions membres

9.7.2.1 def help.helpWindow.__init__ (*self, parent* = `None`)

Le constructeur.

Définition à la ligne 35 du fichier help.py.

9.7.2.2 def help.helpWindow.loadBrowsers (*self, dir, locale*)

met en place les textes dans les afficheurs, en fonction de la locale.

le répertoire où sont les textes au format HTML est **dir**.

Paramètres

dir le répertoire où sont les fichiers HTML

locale la langue choisie

Définition à la ligne 50 du fichier help.py.

9.7.3 Documentation des données membres

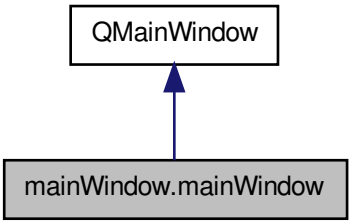
9.7.3.1 help.helpWindow.ui

Définition à la ligne 38 du fichier help.py.

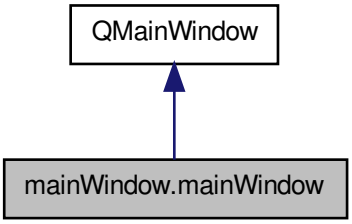
La documentation de cette classe a été générée à partir du fichier suivant :
– src/help.py

9.8 Référence de la classe mainWindow.mainWindow

Graphe d'héritage de mainWindow.mainWindow :



Graphe de collaboration de mainWindow.mainWindow :



Fonctions membres publiques

– def __init__
Le constructeur.

– def applyPreferences

Applique les préférences et les options de ligne de commande.

– def changeWd
change le répertoire par défaut contenant les fichiers de travail

– def tableClicked
fonction de rappel pour un double clic sur un élément de la table

– def diskFromTableRow
trouve le disque qui correspond à une ligne du tableau

– def editOwner
Édition du propriétaire d'une clé.

– def updateButtons
Désactive ou active les flèches selon que l'option correspondante est possible ou non.

– def preference
lance le dialogue des préférences

– def delFiles
Lance l'action de supprimer des fichiers ou des répertoires dans les clés USB.

– def copyTo
Lance l'action de copier vers les clés USB.

– def copyFrom
Lance l'action de copier depuis les clés USB.

– def help
Affiche le widget d'aide.

– def umount
Démonte et détache les clés USB affichées.

– def connectTableModel
Connecte le modèle de table à la table.

– def checkDisks
fonction relancée périodiquement pour vérifier s'il y a un changement dans le baladeurs, et signaler dans le tableau les threads en cours.

Attributs publics

– locale
– ui
– t
– opts
– timer
– threads
– workdir
– checkable
– header
– visibleheader
– tm

9.8.1 Description détaillée

Définition à la ligne 62 du fichier mainWindow.py.

9.8.2 Documentation des fonctions membres

9.8.2.1 def mainWindow.mainWindow.__init__ (*self, parent, opts, locale = "fr_FR"*)

Le constructeur.

Paramètres

parent un QWidget

opts une liste d'options extraite à l'aide de getopts

locale la langue de l'application

Définition à la ligne 70 du fichier mainWindow.py.

9.8.2.2 def mainWindow.mainWindow.applyPreferences (*self*)

Applique les préférences et les options de ligne de commande.

Définition à la ligne 97 du fichier mainWindow.py.

9.8.2.3 def mainWindow.mainWindow.changeWd (*self, newDir*)

change le répertoire par défaut contenant les fichiers de travail

Paramètres

newDir le nouveau nom de répertoire

Définition à la ligne 116 du fichier mainWindow.py.

9.8.2.4 def mainWindow.mainWindow.checkDisks (*self*)

fonction relancée périodiquement pour vérifier s'il y a un changement dans le baladeurs, et signaler dans le tableau les threads en cours.

Le tableau est complètement régénéré à chaque fois, ce qui n'est pas toujours souhaitable.

Définition à la ligne 385 du fichier mainWindow.py.

9.8.2.5 def mainWindow.mainWindow.connectTableModel (*self, data*)

Connecte le modèle de table à la table.

Paramètres

data les données de la table

Définition à la ligne 363 du fichier mainWindow.py.

9.8.2.6 def mainWindow.mainWindow.copyFrom (*self*)

Lance l'action de copier depuis les clés USB.

Définition à la ligne 270 du fichier mainWindow.py.

9.8.2.7 def mainWindow.mainWindow.copyTo (*self*)

Lance l'action de copier vers les clés USB.

Définition à la ligne 250 du fichier mainWindow.py.

9.8.2.8 def mainWindow.mainWindow.delFiles (*self*)

Lance l'action de supprimer des fichiers ou des répertoires dans les clés USB.

Définition à la ligne 224 du fichier mainWindow.py.

9.8.2.9 def mainWindow.mainWindow.diskFromTableRow (*self, ligne*)

trouve le disque qui correspond à une ligne du tableau

Paramètres

ligne la ligne du tableau qui concerne une clé

Renvoie

le disque correspondant à la ligne de tableau (type uDisk)

Définition à la ligne 161 du fichier mainWindow.py.

9.8.2.10 def mainWindow.mainWindow.editOwner (*self, ligne*)

Édition du propriétaire d'une clé.

Paramètres

ligne la ligne du tableau qui concerne une clé

Définition à la ligne 179 du fichier mainWindow.py.

9.8.2.11 def mainWindow.mainWindow.help (*self*)

Affiche le widget d'aide.

Définition à la ligne 320 du fichier mainWindow.py.

9.8.2.12 def mainWindow.mainWindow.preference (*self*)

lance le dialogue des préférences

Définition à la ligne 210 du fichier mainWindow.py.

9.8.2.13 def mainWindow.mainWindow.tableClicked (*self, idx*)

fonction de rappel pour un double clic sur un élément de la table

Paramètres

idx un QModelIndex

Définition à la ligne 125 du fichier mainWindow.py.

9.8.2.14 def mainWindow.mainWindow.umount (*self*)

Démonte et détache les clés USB affichées.

Définition à la ligne 330 du fichier mainWindow.py.

9.8.2.15 def mainWindow.mainWindow.updateButtons (*self*)

Désactive ou active les flèches selon que l'option correspondante est possible ou non.

Pour les flèches : ça aurait du sens de préparer une opération de copie avant même de brancher des clés, donc on les active. Par contre démonter les clés quand elles sont absentes ça n'a pas d'utilité.

Définition à la ligne 195 du fichier mainWindow.py.

9.8.3 Documentation des données membres

9.8.3.1 mainWindow.mainWindow.checkable

Définition à la ligne 103 du fichier mainWindow.py.

9.8.3.2 mainWindow.mainWindow.header

Définition à la ligne 106 du fichier mainWindow.py.

9.8.3.3 mainWindow.mainWindow.locale

Définition à la ligne 73 du fichier mainWindow.py.

9.8.3.4 mainWindow.mainWindow.opts

Définition à la ligne 78 du fichier mainWindow.py.

9.8.3.5 mainWindow.mainWindow.t

Définition à la ligne 77 du fichier mainWindow.py.

9.8.3.6 mainWindow.mainWindow.threads

Définition à la ligne 83 du fichier mainWindow.py.

9.8.3.7 mainWindow.mainWindow.timer

Définition à la ligne 80 du fichier mainWindow.py.

9.8.3.8 mainWindow.mainWindow.tm

Définition à la ligne 370 du fichier mainWindow.py.

9.8.3.9 mainWindow.mainWindow.ui

Définition à la ligne 75 du fichier mainWindow.py.

9.8.3.10 mainWindow.mainWindow.visibleheader

Définition à la ligne 364 du fichier mainWindow.py.

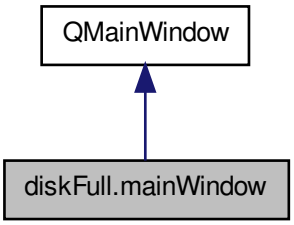
9.8.3.11 mainWindow.mainWindow.workdir

Définition à la ligne 100 du fichier mainWindow.py.

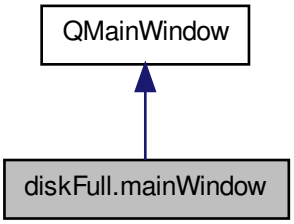
La documentation de cette classe a été générée à partir du fichier suivant :
– src/mainWindow.py

9.9 Référence de la classe diskFull.mainWindow

Graphe d'héritage de diskFull.mainWindow :



Graphe de collaboration de diskFull.mainWindow :



Fonctions membres publiques

– def __init__
Le constructeur.

Attributs publics

– ui
– v
– total
– used

9.9.1 Description détaillée

Définition à la ligne 29 du fichier diskFull.py.

9.9.2 Documentation des fonctions membres

9.9.2.1 def diskFull.mainWindow.__init__ (*self, parent, percent, total = 0, used = 0, title = "Disk"*)

Le constructeur.

Paramètres

parent un QWidget

percent un pourcentage de remplissage de disque

total place totale en kilo-octets

used place utilisée en kilo-octets

title le titre pour la fenêtre

Définition à la ligne 39 du fichier diskFull.py.

9.9.3 Documentation des données membres

9.9.3.1 diskFull.mainWindow.total

Définition à la ligne 47 du fichier diskFull.py.

9.9.3.2 diskFull.mainWindow.ui

Définition à la ligne 43 du fichier diskFull.py.

9.9.3.3 diskFull.mainWindow.used

Définition à la ligne 48 du fichier diskFull.py.

9.9.3.4 diskFull.mainWindow.v

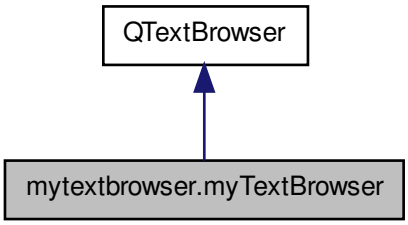
Définition à la ligne 46 du fichier diskFull.py.

La documentation de cette classe a été générée à partir du fichier suivant :
– src/diskFull.py

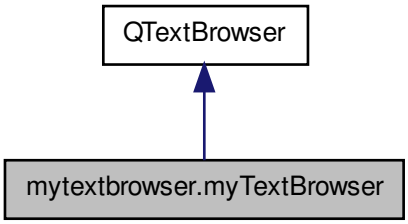
9.10 Référence de la classe mytextbrowser.myTextBrowser

Une classe qui ouvre Firefox quand on clique sur un lien externe.

Graphe d'héritage de mytextbrowser.myTextBrowser :



Graphe de collaboration de mytextbrowser.myTextBrowser :



Fonctions membres publiques

– def setSource
lance Firefox en tâche de fond.

– def setHtml
lien vers la méthode setSource originale

9.10.1 Description détaillée

Une classe qui ouvre Firefox quand on clique sur un lien externe.

Définition à la ligne 34 du fichier mytextbrowser.py.

9.10.2 Documentation des fonctions membres

9.10.2.1 def mytextbrowser.myTextBrowser.setHtml (*self, url*)

lien vers la méthode setSource originale

Paramètres

url l'adresse à ouvrir.

Définition à la ligne 48 du fichier mytextbrowser.py.

9.10.2.2 def mytextbrowser.myTextBrowser.setSource (*self, url*)

lance Firefox en tâche de fond.

Paramètres

url l'adresse à ouvrir.

Définition à la ligne 40 du fichier mytextbrowser.py.

La documentation de cette classe a été générée à partir du fichier suivant :

– src/mytextbrowser.py

9.11 Référence de la classe notification.Notification

Une classe pour afficher des notifications à l'écran.

Fonctions membres publiques

– def __init__
Le constructeur.

– def notify

Attributs publics

– app_name
– replaces_id
– app_icon
– summary
– body
– actions
– hints
– expire_timeout
– interface

9.11.1 Description détaillée

Une classe pour afficher des notifications à l'écran. Doit fonctionner avec tous les gestionnaires de bureau qui adhèrent aux standards de freedesktop.org. Cette classe est basée sur la documentation disponible à http://www.galago-project.org/specs/notification/0.9/x408.html

Définition à la ligne 37 du fichier notification.py.

9.11.2 Documentation des fonctions membres

9.11.2.1 def notification.Notification.__init__ (*self, app_name = "", replaces_id = 0, app_icon = "", summary = "", body = "", actions = [], hints = {}, expire_timeout = 1000*)

Le constructeur.

Paramètres

app_name nom d'une application, valeur par défaut =""
replaces_id identifiant d'une notification à remplacer valeur par défaut=0
app_icon nom d'un fichier servant pour l'icône valeur par défaut=""
summary description brève de la notification valeur par défaut =""
body le texte de la notification, valeur pa défaut=""
actions une liste de paires représeantant des actions, valeur par défaut=[]
hints un dictionnaire de suggestions, valeur par défaut={},
expire_timeout durée maximale d'affichage en millisecondes, valeur par défaut=1000

Définition à la ligne 51 du fichier notification.py.

9.11.2.2 def notification.Notification.notify (*self*)

Définition à la ligne 70 du fichier notification.py.

9.11.3 Documentation des données membres

9.11.3.1 notification.Notification.actions

Définition à la ligne 57 du fichier notification.py.

9.11.3.2 notification.Notification.app_icon

Définition à la ligne 54 du fichier notification.py.

9.11.3.3 notification.Notification.app_name

Définition à la ligne 52 du fichier notification.py.

9.11.3.4 notification.Notification.body

Définition à la ligne 56 du fichier notification.py.

9.11.3.5 notification.Notification.expire_timeout

Définition à la ligne 59 du fichier notification.py.

9.11.3.6 notification.Notification.hints

Définition à la ligne 58 du fichier notification.py.

9.11.3.7 notification.Notification.interface

Définition à la ligne 64 du fichier notification.py.

9.11.3.8 notification.Notification.replaces_id

Définition à la ligne 53 du fichier notification.py.

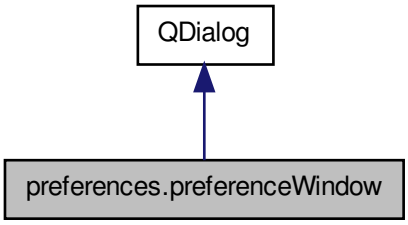
9.11.3.9 notification.Notification.summary

Définition à la ligne 55 du fichier notification.py.

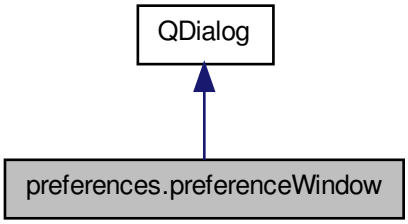
La documentation de cette classe a été générée à partir du fichier suivant :
– src/notification.py

9.12 Référence de la classe preferences.preferenceWindow

Graphe d'héritage de preferences.preferenceWindow :



Graphe de collaboration de preferences.preferenceWindow :



Fonctions membres publiques

– def __init__
Le constructeur.

– def values
– def setValues
Met en place les préférences dans le dialogue.

Attributs publics

– ui

9.12.1 Description détaillée

Définition à la ligne 29 du fichier preferences.py.

9.12.2 Documentation des fonctions membres

9.12.2.1 def preferences.preferenceWindow.__init__ (*self*, *parent* = `None`)

Le constructeur.

Définition à la ligne 34 du fichier preferences.py.

9.12.2.2 def preferences.preferenceWindow.setValues (*self*, *prefs*)

Met en place les préférences dans le dialogue.

Paramètres

prefs un dictionnaire de préférences

Définition à la ligne 55 du fichier preferences.py.

9.12.2.3 def preferences.preferenceWindow.values (*self*)

Renvoie

un dictionnaire de préférences

Définition à la ligne 44 du fichier preferences.py.

9.12.3 Documentation des données membres

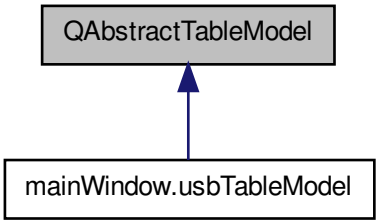
9.12.3.1 preferences.preferenceWindow.ui

Définition à la ligne 37 du fichier preferences.py.

La documentation de cette classe a été générée à partir du fichier suivant :
– src/preferences.py

9.13 Référence de la classe QAbstractTableModel

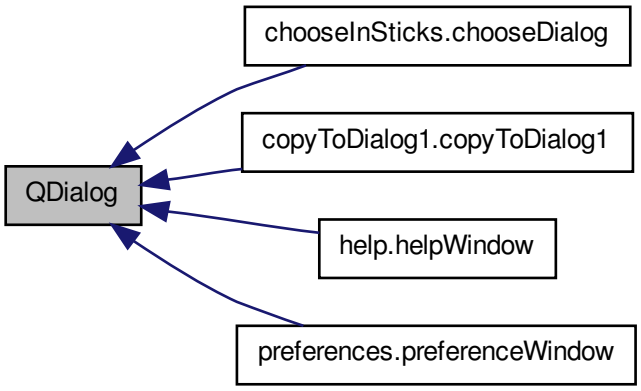
Graphe d'héritage de QAbstractTableModel :



La documentation de cette classe a été générée à partir du fichier suivant :
– src/mainWindow.py

9.14 Référence de la classe QDialog

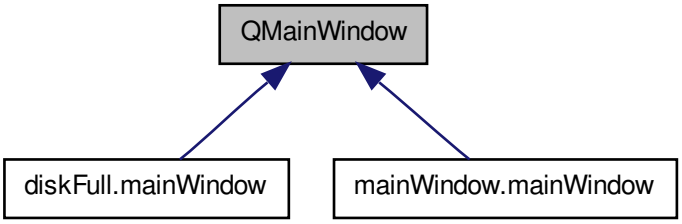
Graphe d'héritage de QDialog :



La documentation de cette classe a été générée à partir du fichier suivant :
– src/chooseInSticks.py

9.15 Référence de la classe QMainWindow

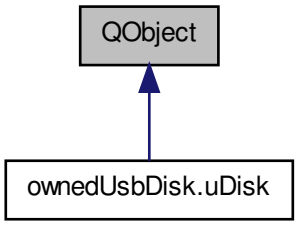
Graphe d'héritage de QMainWindow :



La documentation de cette classe a été générée à partir du fichier suivant :
– src/mainWindow.py

9.16 Référence de la classe QObject

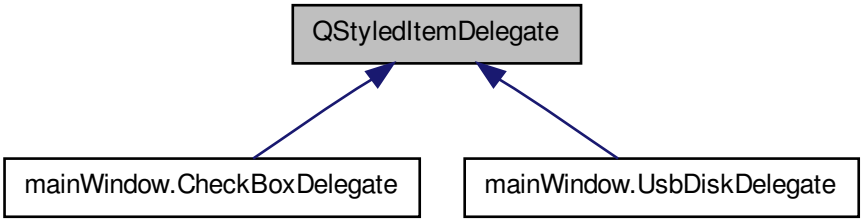
Graphe d'héritage de QObject :



La documentation de cette classe a été générée à partir du fichier suivant :
– src/ownedUsbDisk.py

9.17 Référence de la classe QStyledItemDelegate

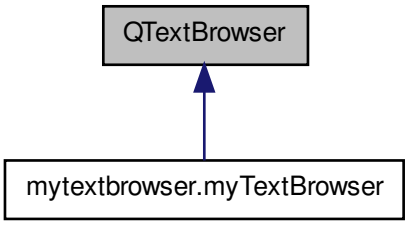
Graphe d'héritage de QStyledItemDelegate :



La documentation de cette classe a été générée à partir du fichier suivant :
– src/mainWindow.py

9.18 Référence de la classe QTextBrowser

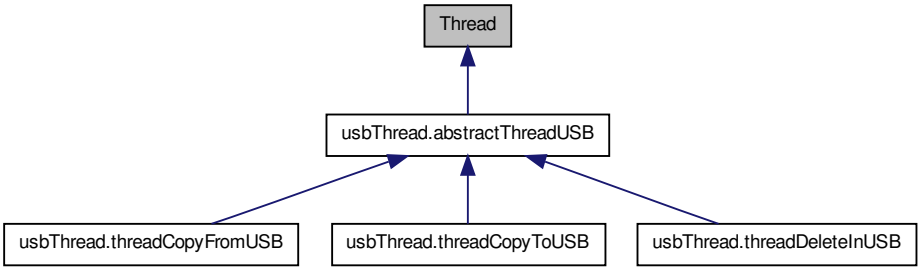
Graphe d'héritage de QTextBrowser :



La documentation de cette classe a été générée à partir du fichier suivant :
– src/mytextbrowser.py

9.19 Référence de la classe Thread

Graphe d'héritage de Thread :

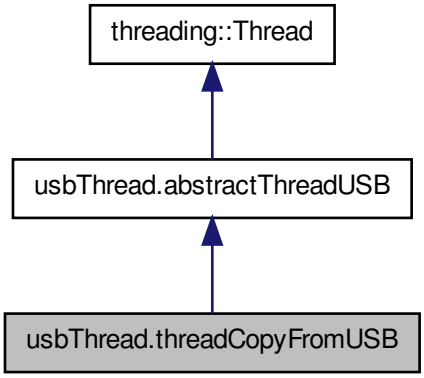


La documentation de cette classe a été générée à partir du fichier suivant :
– src/usbThread.py

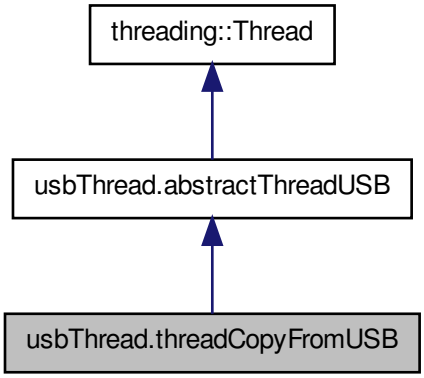
9.20 Référence de la classe usbThread.threadCopyFromUSB

Classe pour les threads copiant depuis les clés USB.

Graphe d'héritage de usbThread.threadCopyFromUSB :



Graphe de collaboration de usbThread.threadCopyFromUSB :



Fonctions membres publiques

– def __init__
Constructeur Crée un thread pour copier une liste de fichiers depuis une clé USB vers un répertoire de disque.

– def toDo
Copie une liste de fichiers d'une clé USB sous un répertoire donné.

Attributs publics

– rootPath
– cmd

9.20.1 Description détaillée

Classe pour les threads copiant depuis les clés USB.

Définition à la ligne 248 du fichier usbThread.py.

9.20.2 Documentation des fonctions membres

9.20.2.1 def usbThread.threadCopyFromUSB.__init__ (*self, ud, fileList, subdir = ".", dest = "/tmp", rootPath = "/", logfile = "/dev/null"*)

Constructeur Crée un thread pour copier une liste de fichiers depuis une clé USB vers un répertoire de disque.

Paramètres

ud l'instance uDisk correspondant à une partition de clé USB

fileList la liste des fichiers à copier

subdir le sous-répertoire de la clé USB d'où faire la copie

dest un répertoire de destination

logfile un fichier de journalisation, /dev/null par défaut

Définition à la ligne 260 du fichier usbThread.py.

9.20.2.2 def usbThread.threadCopyFromUSB.toDo (*self, ud, fileList, subdir, dest, logfile*)

Copie une liste de fichiers d'une clé USB sous un répertoire donné.

À chaque fichier ou répertoire copié, une ligne est journalisée dans le fichier de journal de l'application.

Paramètres

ud l'instance uDisk correspondant à une partition de clé USB

fileList la liste des fichiers à copier

dest un répertoire de destination

logfile un fichier de journalisation

subdir le sous-répertoire de la clé USB où faire la copie

Réimplémentée à partir de usbThread.abstractThreadUSB.

Définition à la ligne 278 du fichier usbThread.py.

9.20.3 Documentation des données membres

9.20.3.1 usbThread.threadCopyFromUSB.cmd

Réimplémentée à partir de usbThread.abstractThreadUSB.

Définition à la ligne 264 du fichier usbThread.py.

9.20.3.2 usbThread.threadCopyFromUSB.rootPath

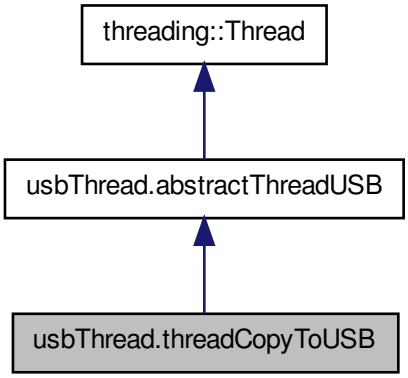
Définition à la ligne 263 du fichier usbThread.py.

La documentation de cette classe a été générée à partir du fichier suivant :
– src/usbThread.py

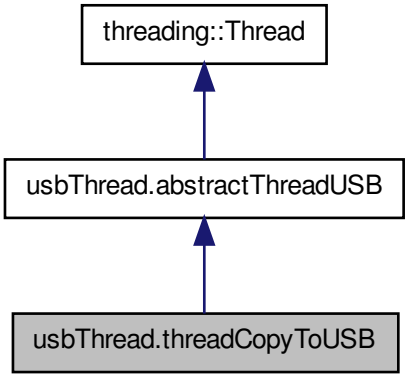
9.21 Référence de la classe usbThread.threadCopyToUSB

Classe pour les threads copiant vers les clés USB.

Graphe d'héritage de usbThread.threadCopyToUSB :



Graphe de collaboration de usbThread.threadCopyToUSB :



Fonctions membres publiques

– def __init__
Constructeur Crée un thread pour copier une liste de fichiers vers une clé USB.

– def threadType
– def toDo
Copie une liste de fichiers vers une clé USB sous un répertoire donné.

Attributs publics

– cmd

9.21.1 Description détaillée

Classe pour les threads copiant vers les clés USB.

Définition à la ligne 197 du fichier usbThread.py.

9.21.2 Documentation des fonctions membres

9.21.2.1 def usbThread.threadCopyToUSB.__init__ (*self, ud, fileList, subdir, logfile* = *"/dev/null"*)

Constructeur Crée un thread pour copier une liste de fichiers vers une clé USB.

Paramètres

ud l'instance uDisk correspondant à une partition de clé USB

fileList la liste des fichiers à copier

subdir le sous-répertoire de la clé USB où faire la copie

logfile un fichier de journalisation, /dev/null par défaut

Définition à la ligne 207 du fichier usbThread.py.

9.21.2.2 def usbThread.threadCopyToUSB.threadType (*self*)

Renvoie

une chaîne courte qui informe sur le type de thread

Réimplémentée à partir de usbThread.abstractThreadUSB.

Définition à la ligne 215 du fichier usbThread.py.

9.21.2.3 def usbThread.threadCopyToUSB.toDo (*self, ud, fileList, subdir, dest, logfile*)

Copie une liste de fichiers vers une clé USB sous un répertoire donné.

Ce répertoire est composé de ud.visibleDir() joint au sous-répertoire subdir. À chaque fichier ou répertoire copié, une ligne est journalisée dans le fichier de journal de l'application.

Paramètres

ud l'instance uDisk correspondant à une partition de clé USB

fileList la liste des fichiers à copier

logfile un fichier de journalisation

subdir le sous-répertoire de la clé USB où faire la copie

Réimplémentée à partir de usbThread.abstractThreadUSB.

Définition à la ligne 230 du fichier usbThread.py.

9.21.3 Documentation des données membres

9.21.3.1 usbThread.threadCopyToUSB.cmd

Réimplémentée à partir de usbThread.abstractThreadUSB.

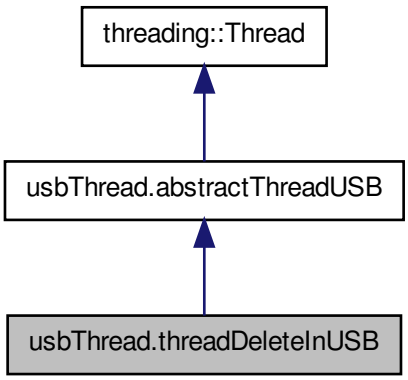
Définition à la ligne 209 du fichier usbThread.py.

La documentation de cette classe a été générée à partir du fichier suivant :
– src/usbThread.py

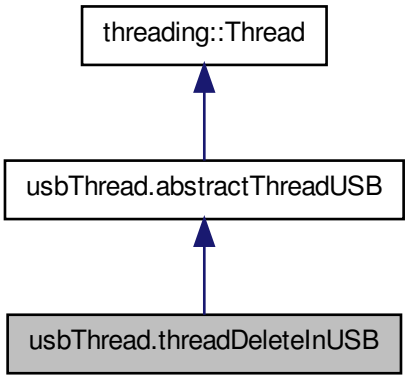
9.22 Référence de la classe usbThread.threadDeleteInUSB

Classe pour les threads effaçant des sous-arbres dans les clés USB.

Graphe d'héritage de usbThread.threadDeleteInUSB :



Graphe de collaboration de usbThread.threadDeleteInUSB :



Fonctions membres publiques

– def __init__
Constructeur Crée un thread pour supprimer une liste de fichiers dans une clé USB.

– def toDo
Supprime une liste de fichiers dans une clé USB.

Attributs publics

– cmd

9.22.1 Description détaillée

Classe pour les threads effaçant des sous-arbres dans les clés USB.

Définition à la ligne 298 du fichier usbThread.py.

9.22.2 Documentation des fonctions membres

9.22.2.1 def usbThread.threadDeleteInUSB.__init__ (*self, ud, fileList, subdir, logfile* = `"/dev/null"`)

Constructeur Crée un thread pour supprimer une liste de fichiers dans une clé USB.

Paramètres

ud l'instance uDisk correspondant à une partition de clé USB

fileList la liste des fichiers à supprimer

subdir le sous-répertoire de la clé USB où faire les suppressions

logfile un fichier de journalisation, /dev/null par défaut

Définition à la ligne 308 du fichier usbThread.py.

9.22.2.2 def usbThread.threadDeleteInUSB.toDo (*self, ud, fileList, subdir, dest, logfile*)

Supprime une liste de fichiers dans une clé USB.

La liste est prise sous un répertoire donné. Le répertoire visible qui dépend du constructuer d ela clé est pris en compte. À chaque fichier ou répertoire supprimé, une ligne est journalisée dans le fichier de journal de l'application.

Paramètres

l'instance uDisk correspondant à une partition de clé USB

fileList la liste des fichiers à copier

dest un répertoire de destination

logfile un fichier de journalisation

subdir le sous-répertoire de la clé USB où faire la copie

Réimplémentée à partir de usbThread.abstractThreadUSB.

Définition à la ligne 325 du fichier usbThread.py.

9.22.3 Documentation des données membres

9.22.3.1 usbThread.threadDeleteInUSB.cmd

Réimplémentée à partir de usbThread.abstractThreadUSB.

Définition à la ligne 310 du fichier usbThread.py.

La documentation de cette classe a été générée à partir du fichier suivant :
– src/usbThread.py

9.23 Référence de la classe usbThread.ThreadRegister

Une classe pour tenir un registre des threads concernant les baladeurs.

Fonctions membres publiques

– def __init__
Le constructure met en place un dictionnaire.

– def __str__
– def push
– def pop
– def busy
Indique si le disque est occupé par des threads.

Attributs publics

– dico

9.23.1 Description détaillée

Une classe pour tenir un registre des threads concernant les baladeurs.

Définition à la ligne 32 du fichier usbThread.py.

9.23.2 Documentation des fonctions membres

9.23.2.1 def usbThread.ThreadRegister.__init__ (*self*)

Le constructure met en place un dictionnaire.

Définition à la ligne 38 du fichier usbThread.py.

9.23.2.2 def usbThread.ThreadRegister.__str__ (*self*)

Définition à la ligne 41 du fichier usbThread.py.

9.23.2.3 def usbThread.ThreadRegister.busy (*self, owner*)

Indique si le disque est occupé par des threads.

Paramètres

owner le propriétaire du disque

Renvoie

les données associées par le dictionnaire

Définition à la ligne 71 du fichier usbThread.py.

9.23.2.4 def usbThread.ThreadRegister.pop (*self, ud, thread*)

Paramètres

ud un disque

thread un thread Dépile un thread pour le baladeur ud

Définition à la ligne 62 du fichier usbThread.py.

9.23.2.5 def usbThread.ThreadRegister.push (*self, ud, thread*)

Paramètres

ud un disque

thread un thread Empile un thread pour le baladeur ud

Définition à la ligne 50 du fichier usbThread.py.

9.23.3 Documentation des données membres

9.23.3.1 usbThread.ThreadRegister.dico

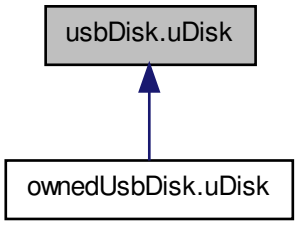
Définition à la ligne 39 du fichier usbThread.py.

La documentation de cette classe a été générée à partir du fichier suivant :
– src/usbThread.py

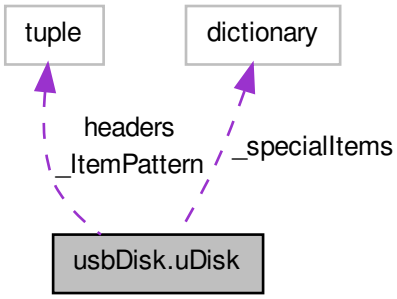
9.24 Référence de la classe usbDisk.uDisk

une classe pour représenter un disque ou une partition.

Graphe d'héritage de usbDisk.uDisk :



Graphe de collaboration de usbDisk.uDisk :



Fonctions membres publiques

- def __init__
  Le constructeur.
- def getFatUuid
  renvoie l'uuid de la première partition FAT après que celle-ci aura été identifiée (utile pour les disques partitionnés)
- def uniqueId
  renvoie un identifiant unique.
- def headers
  Méthode statique, pour avoir des titres de colonne.
- def devicePropProxy
  renvoie un proxy vers un navigateur de propriétés
- def isTrue
  Renvoie la valeur de vérité d'une propriété.
- def isUsbDisk
  Facilite le réprage des disques USB USB.
- def __str__
  Fournit une représentation imprimable.
- def title
  Permet d'obtenir un identifiant unique de disque.
- def file
  Permet d'accèder à l'instance par un nom de fichier.
- def mountPoint
  Permet d'accèder à l'instance par un point de montage.
- def getProp
  Facilite l'accès aux propriétés à l'aide des mots clés du module udisks.
- def isDosFat
  Permet de reconnaitre les partitions DOS-FAT.
- def valuableProperties
  Facilite l'accès aux propriétés intéressantes d'une instance.
- def master
  renvoie le chemin du disque, dans le cas où self est une partition
- def unNumberProp
  retire le numéro des en-têtes pour en faire un nom de propriété valide pour interroger dbus
- def __getitem__
  Renvoie un élément de listage de données internes au disque.
- def showableProp
  Renvoie une propriété dans un type "montrable" par QT.
- def getFirstFat
  Renvoie la première partition VFAT.
- def ensureMounted
  Permet de s'assurer qu'une partition ou un disque sera bien monté.

Attributs publics

- path
- device
- device_prop
- selected
- checkable
- stickid
- uuid
- fatuuid
- firstFat

Attributs publics statiques

- tuple headers = staticmethod(headers)

9.24.1 Description détaillée

une classe pour représenter un disque ou une partition. les attributs publics sont :
- **path** le chemin dans le système dbus
- **device** l'objet dbus qui correspond à l'instance
- **device_prop** un proxy pour questionner cet objet dbus
- **selected** booléen vrai si on doit considérer cette instance comme sélectionnée. Vrai à l'initialisation
- **checkable** booléen vrai si on veut que la sélection puisse être modifiée par l'utilisateur dans l'interface graphique

Définition à la ligne 42 du fichier usbDisk.py.

9.24.2 Documentation des fonctions membres

9.24.2.1 def usbDisk.uDisk.__getitem__ (*self, n*)

Renvoie un élément de listage de données internes au disque.

Paramètres

n un nombre

checkable vrai si on doit renvoyer une propriété supplémentaire pour n==0

Renvoie

si checkable est vrai, un élément si n>0, et le drapeau self.selected si n==0 ; sinon un élément de façon ordinaire. Les noms des éléments sont dans la liste itemNames utilisée dans la fonction statique headers

Définition à la ligne 283 du fichier usbDisk.py.

9.24.2.2 def usbDisk.uDisk.__init__ (*self, path, bus, checkable* = `False`)

Le constructeur.

Paramètres

path un chemin dans le système dbus

bus un objet dbus.BusSystem

checkable vrai si on fera usage de self.selected

Définition à la ligne 51 du fichier usbDisk.py.

9.24.2.3 def usbDisk.uDisk.__str__ (*self*)

Fournit une représentation imprimable.

Renvoie

une représentation imprimable de l'instance

Définition à la ligne 148 du fichier usbDisk.py.

9.24.2.4 def usbDisk.uDisk.devicePropProxy (*self, bus*)

renvoie un proxy vers un navigateur de propriétés

Paramètres

bus une instace de dbus.SystemBus

Renvoie

l'objet proxy

Définition à la ligne 119 du fichier usbDisk.py.

9.24.2.5 def usbDisk.uDisk.ensureMounted (*self*)

Permet de s'assurer qu'une partition ou un disque sera bien monté.

Renvoie

le chemin du point de montage

Définition à la ligne 333 du fichier usbDisk.py.

9.24.2.6 def usbDisk.uDisk.file (*self*)

Permet d'accèder à l'instance par un nom de fichier.

Renvoie

un nom valide dans le système de fichiers, pour accéder à l'instance.

Définition à la ligne 165 du fichier usbDisk.py.

9.24.2.7 def usbDisk.uDisk.getFatUuid (*self*)

renvoie l'uuid de la première partition FAT après que celle-ci aura été identifiée (utile pour les disques partitionnés)

Renvoie

un uuid

Définition à la ligne 82 du fichier usbDisk.py.

9.24.2.8 def usbDisk.uDisk.getFirstFat (*self*)

Renvoie la première partition VFAT.

Renvoie

la première partition VFAT ou None s'il n'y en a pas

Définition à la ligne 320 du fichier usbDisk.py.

9.24.2.9 def usbDisk.uDisk.getProp (*self, name*)

Facilite l'accès aux propriétés à l'aide des mots clés du module udisks.

Paramètres

name le nom d'une propriété

Renvoie

une propriété dbus du disque ou de la partition, sinon None si le nom name est illégal

Définition à la ligne 188 du fichier usbDisk.py.

9.24.2.10 def usbDisk.uDisk.headers (*checkable* = `False`, *locale* = `"C"`)

Méthode statique, pour avoir des titres de colonne.

renvoie des titres pour les items obtenus par __getitem__. Le résultat dépend du paramètre checkable.

Paramètres

checkable vrai si le premier en-tête correspond à une colonne de cases à cocher

locale la locale, pour traduire les titres éventuellement. Valeur par défaut : "C"

Renvoie

une liste de titres de colonnes

Définition à la ligne 104 du fichier usbDisk.py.

9.24.2.11 def usbDisk.uDisk.isDosFat (*self*)

Permet de reconnaitre les partitions DOS-FAT.

Renvoie

vrai dans le cas d'une partition FAT16 ou FAT32

Définition à la ligne 199 du fichier usbDisk.py.

9.24.2.12 def usbDisk.uDisk.isTrue (*self, prop, value = None*)

Renvoie la valeur de vérité d'une propriété.

Paramètres

prop une propriété

value

Renvoie

vrai si la propriété est vraie (cas où value==None) ou vrai si la propriété a exactement la valeur value.

Définition à la ligne 129 du fichier usbDisk.py.

9.24.2.13 def usbDisk.uDisk.isUsbDisk (*self*)

Facilite le réprage des disques USB USB.

Renvoie

vrai dans le cas d'un disque USB

Définition à la ligne 140 du fichier usbDisk.py.

9.24.2.14 def usbDisk.uDisk.master (*self*)

renvoie le chemin du disque, dans le cas où self est une partition

Renvoie

le chemin dbus du disque maître, sinon "/"

Définition à la ligne 257 du fichier usbDisk.py.

9.24.2.15 def usbDisk.uDisk.mountPoint (*self*)

Permet d'accèder à l'instance par un point de montage.

Renvoie

un point de montage, s'il en existe, sinon None

Définition à la ligne 175 du fichier usbDisk.py.

9.24.2.16 def usbDisk.uDisk.showableProp (*self, name*)

Renvoie une propriété dans un type "montrable" par QT.

les propriétés que renvoie dbus ont des types inconnus de Qt4, cette fonction les transtype pour que QVariant arrive à les prendre en compte.

Paramètres

name le nom de la propriété

Renvoie

une nombre ou une chaîne selon le type de propriété

Définition à la ligne 303 du fichier usbDisk.py.

9.24.2.17 def usbDisk.uDisk.title (*self*)

Permet d'obtenir un identifiant unique de disque.

Renvoie

le chemin dbus de l'instance

Définition à la ligne 156 du fichier usbDisk.py.

9.24.2.18 def usbDisk.uDisk.uniqueId (*self*)

renvoie un identifiant unique.

Dans cette classe, cette fonction est synonyme de getFatUuid

Renvoie

un identifiant supposément unique

Définition à la ligne 91 du fichier usbDisk.py.

9.24.2.19 def usbDisk.uDisk.unNumberProp (*self, n*)

retire le numéro des en-têtes pour en faire un nom de propriété valide pour interroger dbus

Paramètres

n un numéro de propriété qui se réfère aux headers

Renvoie

une propriété renvoyée par dbus, dans un format imprimable

Définition à la ligne 267 du fichier usbDisk.py.

9.24.2.20 def usbDisk.uDisk.valuableProperties (*self, indent = 4*)

Facilite l'accès aux propriétés intéressantes d'une instance.

Renvoie

une chaîne indentée avec les propriétés intéressantes, une par ligne

Définition à la ligne 212 du fichier usbDisk.py.

9.24.3 Documentation des données membres

9.24.3.1 usbDisk.uDisk.checkable

Définition à la ligne 56 du fichier usbDisk.py.

9.24.3.2 usbDisk.uDisk.device

Définition à la ligne 53 du fichier usbDisk.py.

9.24.3.3 usbDisk.uDisk.device_prop

Définition à la ligne 54 du fichier usbDisk.py.

9.24.3.4 usbDisk.uDisk.fatuuid

Définition à la ligne 59 du fichier usbDisk.py.

9.24.3.5 usbDisk.uDisk.firstFat

Définition à la ligne 60 du fichier usbDisk.py.

9.24.3.6 tuple usbDisk.uDisk.headers = staticmethod(headers) [static]

Définition à la ligne 111 du fichier usbDisk.py.

9.24.3.7 usbDisk.uDisk.path

Définition à la ligne 52 du fichier usbDisk.py.

9.24.3.8 usbDisk.uDisk.selected

Définition à la ligne 55 du fichier usbDisk.py.

9.24.3.9 usbDisk.uDisk.stickid

Définition à la ligne 57 du fichier usbDisk.py.

9.24.3.10 usbDisk.uDisk.uuid

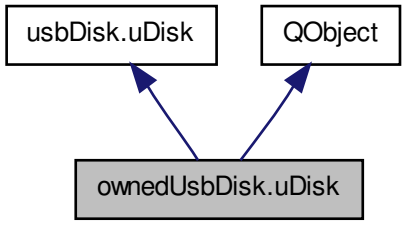
Définition à la ligne 58 du fichier usbDisk.py.

La documentation de cette classe a été générée à partir du fichier suivant :
– src/usbDisk.py

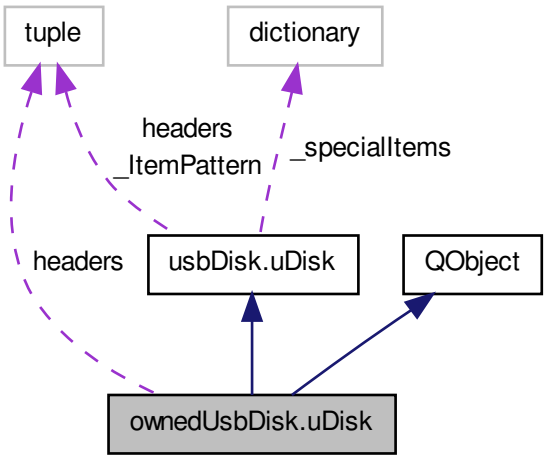
9.25 Référence de la classe ownedUsbDisk.uDisk

une classe qui ajoute un nom de propriétaire aux disque USB, et qui en même temps ajoute des particularités selon le nom du vendeur et le modèle.

Graphe d'héritage de ownedUsbDisk.uDisk :



Graphe de collaboration de ownedUsbDisk.uDisk :



Fonctions membres publiques

– def __init__
– def uniqueId
renvoie un identifiant unique.

– def tattoo
Renvoie un tatouage présent sur la clé, quitte à le créer.

– def readQuirks
Lit un dictionnaire indexé par le noms de vendeurs et les noms de modèle pour associer à ces modèles particuliers un répertoire visible.

- def visibleDir
  Renvoie le répertoire particulier de la partition qui sera visible quand le baladeur est utilisé par son interface utilisateur.

- def headers
  Méthode statique renvoie des titres pour les items obtenus par __getitem__ la deuxième colonne sera toujours le propriétaire.

- def ownerByDb
  renvoie un nom de propriétaire dans tous les cas.

- def __getitem__
  renvoie un élément de listage de données internes au disque Fait en sorte que la deuxième colonne soit toujours le propriétaire

- def ensureOwner
  Demande un nom de propriétaire si celui-ci n'est pas encore défini pour cette clé USB.

Attributs publics

- owner
- vendor
- model
- visibleDirs

Attributs publics statiques

- tuple headers = staticmethod(headers)

9.25.1 Description détaillée

une classe qui ajoute un nom de propriétaire aux disque USB, et qui en même temps ajoute des particularités selon le nom du vendeur et le modèle.

Définition à la ligne 59 du fichier ownedUsbDisk.py.

9.25.2 Documentation des fonctions membres

9.25.2.1 def ownedUsbDisk.uDisk.__getitem__ (*self, n*)

renvoie un élément de listage de données internes au disque Fait en sorte que la deuxième colonne soit toujours le propriétaire

Paramètres

n un nombre

checkable vrai si on doit renvoyer une propriété supplémentaire pour n==0

Renvoie

si checkable est vrai, un élément si n>0, et le drapeau self.selected si n==0 ; sinon un élément de façon ordinaire. Les noms des éléments sont dans la liste self.itemNames

Définition à la ligne 179 du fichier ownedUsbDisk.py.

9.25.2.2 def ownedUsbDisk.uDisk.__init__ (*self, path, bus, checkable* = `False`)

Paramètres

path un chemin dans le système dbus

bus un objet dbus.BusSystem

checkable vrai si on fera usage de self.selected

Définition à la ligne 66 du fichier ownedUsbDisk.py.

9.25.2.3 def ownedUsbDisk.uDisk.ensureOwner (*self*)

Demande un nom de propriétaire si celui-ci n'est pas encore défini pour cette clé USB.

Renvoie

un nom de propriétaire si c'est un disque, sinon None

Définition à la ligne 207 du fichier ownedUsbDisk.py.

9.25.2.4 def ownedUsbDisk.uDisk.headers (*checkable* = `False`, *locale* = `"C"`)

Méthode statique renvoie des titres pour les items obtenus par __getitem__ la deuxième colonne sera toujours le propriétaire.

Paramètres

checkable vrai si le premier en-tête correspond à une colonne de cases à cocher

locale la locale, pour traduire les titres

Renvoie

une liste de titres de colonnes

Définition à la ligne 150 du fichier ownedUsbDisk.py.

9.25.2.5 def ownedUsbDisk.uDisk.ownerByDb (*self*)

renvoie un nom de propriétaire dans tous les cas.

Définition à la ligne 160 du fichier ownedUsbDisk.py.

9.25.2.6 def ownedUsbDisk.uDisk.readQuirks (*self*)

Lit un dictionnaire indexé par le noms de vendeurs et les noms de modèle pour associer à ces modèles particuliers un répertoire visible.

voir la fonction visibleDir. Ce dictionnaire est dans le fichier /usr/share/scolasync/marques.py ou dans ${HOME}/.scolasync/marques.py, (sous Linux) cette dernière place étant prépondérante.

Définition à la ligne 118 du fichier ownedUsbDisk.py.

9.25.2.7 def ownedUsbDisk.uDisk.tattoo (*self*)

Renvoie un tatouage présent sur la clé, quitte à le créer.

Renvoie

un tatouage, supposément unique.

Définition à la ligne 90 du fichier ownedUsbDisk.py.

9.25.2.8 def ownedUsbDisk.uDisk.uniqueId (*self*)

renvoie un identifiant unique.

reprend l'Uuid hérité de la classe parente, mais y ajoute un nombre supplémentaire, éventuellement placé par "tatouage" sur la clé.

Renvoie

un identifiant supposément unique

Définition à la ligne 82 du fichier ownedUsbDisk.py.

9.25.2.9 def ownedUsbDisk.uDisk.visibleDir (*self*)

Renvoie le répertoire particulier de la partition qui sera visible quand le baladeur est utilisé par son interface utilisateur.

Ce répertoire peut varier selon les vendeurs et les modèles.

Définition à la ligne 134 du fichier ownedUsbDisk.py.

9.25.3 Documentation des données membres

9.25.3.1 tuple ownedUsbDisk.uDisk.headers = staticmethod(headers) `[static]`

Définition à la ligne 199 du fichier ownedUsbDisk.py.

9.25.3.2 ownedUsbDisk.uDisk.model

Définition à la ligne 71 du fichier ownedUsbDisk.py.

9.25.3.3 ownedUsbDisk.uDisk.owner

Définition à la ligne 69 du fichier ownedUsbDisk.py.

9.25.3.4 ownedUsbDisk.uDisk.vendor

Définition à la ligne 70 du fichier ownedUsbDisk.py.

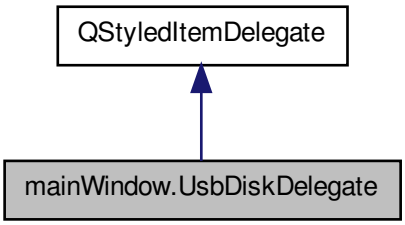
9.25.3.5 ownedUsbDisk.uDisk.visibleDirs

Définition à la ligne 72 du fichier ownedUsbDisk.py.

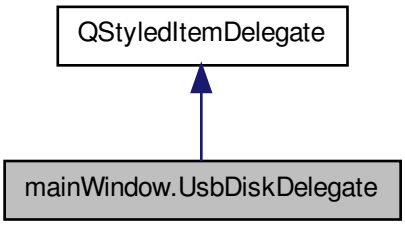
La documentation de cette classe a été générée à partir du fichier suivant :
– src/ownedUsbDisk.py

9.26 Référence de la classe mainWindow.UsbDiskDelegate

Graphe d'héritage de mainWindow.UsbDiskDelegate :



Graphe de collaboration de mainWindow.UsbDiskDelegate :



Fonctions membres publiques

– def __init__
– def paint

Attributs publics

– okPixmap

– busyPixmap

9.26.1 Description détaillée

Définition à la ligne 521 du fichier mainWindow.py.

9.26.2 Documentation des fonctions membres

9.26.2.1 def mainWindow.UsbDiskDelegate.__init__ (*self, parent*)

Définition à la ligne 522 du fichier mainWindow.py.

9.26.2.2 def mainWindow.UsbDiskDelegate.paint (*self, painter, option, index*)

Définition à la ligne 527 du fichier mainWindow.py.

9.26.3 Documentation des données membres

9.26.3.1 mainWindow.UsbDiskDelegate.busyPixmap

Définition à la ligne 525 du fichier mainWindow.py.

9.26.3.2 mainWindow.UsbDiskDelegate.okPixmap

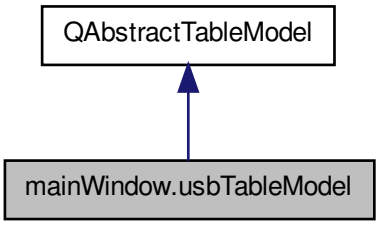
Définition à la ligne 524 du fichier mainWindow.py.

La documentation de cette classe a été générée à partir du fichier suivant :
– src/mainWindow.py

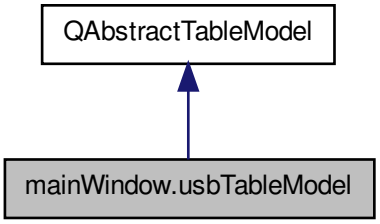
9.27 Référence de la classe mainWindow.usbTableModel

Un modèle de table pour des séries de clés USB.

Graphe d'héritage de mainWindow.usbTableModel :



Graphe de collaboration de mainWindow.usbTableModel :



Fonctions membres publiques

- def __init__
- def rowCount
  un QModelIndex
- def columnCount
  un QModelIndex
- def setData
- def data
- def headerData
- def sort
  Sort table by given column number.

Attributs publics

- header
- donnees
- checkable
- pere

9.27.1 Description détaillée

Un modèle de table pour des séries de clés USB.

Définition à la ligne 399 du fichier mainWindow.py.

9.27.2 Documentation des fonctions membres

9.27.2.1 def mainWindow.usbTableModel.__init__ (*self*, *parent* = `None`, *header* = `[]`, *donnees* = `None`, *checkable* = `False`)

Paramètres

parent un QObject

header les en-têtes de colonnes
donnees les données
checkable vrai si la première colonne est composée de boîtes à cocher. Faux par défaut

Définition à la ligne 408 du fichier mainWindow.py.

9.27.2.2 def mainWindow.usbTableModel.columnCount (*self, parent*)

un QModelIndex

Définition à la ligne 426 du fichier mainWindow.py.

9.27.2.3 def mainWindow.usbTableModel.data (*self, index, role*)

Définition à la ligne 436 du fichier mainWindow.py.

9.27.2.4 def mainWindow.usbTableModel.headerData (*self, section, orientation, role*)

Définition à la ligne 465 du fichier mainWindow.py.

9.27.2.5 def mainWindow.usbTableModel.rowCount (*self, parent*)

un QModelIndex

Définition à la ligne 419 du fichier mainWindow.py.

9.27.2.6 def mainWindow.usbTableModel.setData (*self, index, value, role*)

Définition à la ligne 429 du fichier mainWindow.py.

9.27.2.7 def mainWindow.usbTableModel.sort (*self, Ncol, order* = `Qt.DescendingOrder`)

Sort table by given column number.

Paramètres

Ncol numéro de la colonne de tri
order l'odre de tri, Qt.DescendingOrder par défaut

Définition à la ligne 477 du fichier mainWindow.py.

9.27.3 Documentation des données membres

9.27.3.1 mainWindow.usbTableModel.checkable

Définition à la ligne 412 du fichier mainWindow.py.

9.27.3.2 mainWindow.usbTableModel.donnees

Définition à la ligne 411 du fichier mainWindow.py.

9.27.3.3 mainWindow.usbTableModel.header

Définition à la ligne 410 du fichier mainWindow.py.

9.27.3.4 mainWindow.usbTableModel.pere

Définition à la ligne 413 du fichier mainWindow.py.

La documentation de cette classe a été générée à partir du fichier suivant :
– src/mainWindow.py

Chapitre 10

Documentation des fichiers

10.1 Référence du fichier src/chooseInSticks.py

Classes

– class chooseInSticks.chooseDialog
Un dialogue pour choisir un ensemble de fichiers à copier depuis une clé USB.

Paquetages

– package chooseInSticks

Variables

– string chooseInSticks.licenceEn

10.2 Référence du fichier src/copyToDialog1.py

Classes

– class copyToDialog1.copyToDialog1
Un dialogue pour choisir un ensemble de fichiers à transférer vers une collection de clés USB.

Paquetages

– package copyToDialog1

Variables

– string copyToDialog1.licenceEn
– tuple copyToDialog1.app = QApplication(sys.argv)
– tuple copyToDialog1.windows = copyToDialog1()

10.3 Référence du fichier src/db.py

Paquetages

– package db

Fonctions

– def db.openDb
Ouverture de la base de données de l'application, et création si nécessaire.

– def db.checkVersion
Vérifie si la base de données reste compatible.

– def db.knowsId
dit si une clé USB est déjà connue

– def db.tattooList
Renvoie la liste des tatouages connus de la base de données.

– def db.readStudent
renvoie l'étudiant qui possède une clé USB

– def db.readPrefs
renvoie les préférences de ScolaSync

– def db.setWd
définit le nouveau nom du répertoire de travail préféré.

– def db.writeStudent
inscrit un étudiant comme propriétaire d'une clé USB

– def db.writePrefs
inscrit les préférences

Variables

– dictionary db.licence = {}
– db.database = None
– db.cursor = None

10.4 Référence du fichier src/diskFull.py

Classes

– class diskFull.mainWindow

Paquetages

– package diskFull

Variables

– dictionary diskFull.licence = {}

10.5 Référence du fichier src/globaldef.py

Paquetages

– package globaldef

Variables

– string globaldef.licenceEn
globaldef.py is part of the package scolasync.

– string globaldef.userShareDir = "~/.scolasync"
– string globaldef.logFileName = "~/.scolasync/scolasync.log"
– string globaldef.markFileName = "~/.scolasync/marques.py"

10.6 Référence du fichier src/help.py

Classes

– class help.helpWindow

Paquetages

– package help

Variables

– dictionary help.licence = { }

10.7 Référence du fichier src/mainWindow.py

Classes

– class mainWindow.mainWindow
– class mainWindow.usbTableModel
Un modèle de table pour des séries de clés USB.

– class mainWindow.CheckBoxDelegate
– class mainWindow.UsbDiskDelegate

Paquetages

– package mainWindow

Fonctions

– def mainWindow.firstdir
Renvoie le premier répertoire existant d'une liste de propositions.

– def mainWindow.CheckBoxRect

Variables

– dictionary mainWindow.licence = {}
– tuple mainWindow.globalDiskData = ownedUsbDisk.Available(True,access="firstFat")

10.8 Référence du fichier src/marques.py

Paquetages

– package marques

10.9 Référence du fichier src/mytextbrowser.py

Classes

– class mytextbrowser.myTextBrowser
Une classe qui ouvre Firefox quand on clique sur un lien externe.

Paquetages

– package mytextbrowser

Variables

– dictionary mytextbrowser.licence = {}

10.10 Référence du fichier src/notification.py

Classes

– class notification.Notification
Une classe pour afficher des notifications à l'écran.

Paquetages

– package notification

Variables

– dictionary notification.licence = {}
– tuple notification.notif

10.11 Référence du fichier src/ownedUsbDisk.py

Classes

– class ownedUsbDisk.uDisk
une classe qui ajoute un nom de propriétaire aux disque USB, et qui en même temps ajoute des particularités selon le nom du vendeur et le modèle.

– class ownedUsbDisk.Available
Une classe qui fournit une collection de disques USB connectés, avec leurs propriétaires.

Paquetages

– package ownedUsbDisk

Fonctions

– def ownedUsbDisk.editRecord
édition de la base de données.

Variables

– dictionary ownedUsbDisk.licence = { }

10.12 Référence du fichier src/preferences.py

Classes

– class preferences.preferenceWindow

Paquetages

– package preferences

Variables

– dictionary preferences.licence = { }

10.13 Référence du fichier src/scolasync.py

Paquetages

– package scolasync
Scolasync est un programme pour gérer des transferts de fichiers entre un ordinateur et une collection de clés USB.

Fonctions

– def scolasync.usage
  affiche le mode d'emploi à la console

– def scolasync.run
  C'est la fonction principale.

Variables

– dictionary scolasync.licence = {}
– string scolasync.licenceEn
– string scolasync.licenceFr

10.14 Référence du fichier src/usbDisk.py

Classes

– class usbDisk.uDisk
  une classe pour représenter un disque ou une partition.

– class usbDisk.Available
  une classe pour représenter la collection des disques USB connectés

Paquetages

– package usbDisk

Variables

– dictionary usbDisk.licence = {}
– string usbDisk.licence_en
– tuple usbDisk.machin = Available()

10.15 Référence du fichier src/usbThread.py

Classes

– class usbThread.ThreadRegister
  Une classe pour tenir un registre des threads concernant les baladeurs.

– class usbThread.abstractThreadUSB
  Une classe abstraite Cette classe sert de creuset pour les classe servant aux copies et aux effacement.

– class usbThread.threadCopyToUSB
  Classe pour les threads copiant vers les clés USB.

– class usbThread.threadCopyFromUSB
  Classe pour les threads copiant depuis les clés USB.

– class usbThread.threadDeleteInUSB
  Classe pour les threads effaçant des sous-arbres dans les clés USB.

Paquetages

– package usbThread

Variables

– string usbThread.licenceEn
– int usbThread._threadNumber = 0

10.16 Référence du fichier src/version.py

Paquetages

– package version

Fonctions

– def version.major
– def version.minor
– def version.version

Variables

– dictionary version.licence = {}