ScolaSync

1.0

Généré par Doxygen 1.8.1.1

Dimanche Juin 24 2012 02 :27 :18

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

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

Index des espaces de nommage

2.1 Paquetages

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

Chapitre 3

Index des classes

3.1 Hiérarchie des classes

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

Chapitre 4

Index des classes

4.1 Liste des classes

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

Chapitre 5

Index des fichiers

5.1 Liste des fichiers

Liste de tous les fichiers avec une brève description :

Chapitre 6

Documentation des espaces de nommage

6.1 Référence de l'espace de nommage scolasync

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

6.1.1 Description détaillée

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

6.2 Référence de l'espace de nommage src

Paquetages

- namespace checkBoxDialog
- namespace choixEleves
- namespace chooseInSticks
- namespace copyToDialog1
- namespace db
- namespace deviceListener
- namespace diskFull
- namespace gestClasse
- namespace gestclassetreeview
- namespace globaldef
- namespace help
- namespace mainWindow
- namespace marques
- namespace mytextbrowser
- namespace nameAdrive
- namespace notification
- namespace ownedUsbDisk
- namespace preferences
- namespace scolasync
- namespace sconet
- namespace usbDisk
- namespace usbThread
- namespace version

6.3 Référence de l'espace de nommage src.checkBoxDialog

Classes

- class CheckBoxDialog
  Un dialogue pour gérer les cases à cocher de l'application.

Variables

– string licenceEn

6.3.1 Documentation des variables

6.3.1.1 string src.checkBoxDialog.licenceEn

Valeur initiale :

```
""" 
    file checkBoxDialog.py
    this file is part of the project scolasync

    Copyright (C) 2010 Georges Khaznadar <georgesk@ofset.org>

    This program is free software: you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version3 of the License, or
     (at your option) any later version.

     This program is distributed in the hope that it will be useful,
     but WITHOUT ANY WARRANTY; without even the implied warranty of
     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
     GNU General Public License for more details.

     You should have received a copy of the GNU General Public License
     along with this program.  If not, see <http://www.gnu.org/licenses/>.
"""
```

Définition à la ligne 3 du fichier checkBoxDialog.py.

6.4 Référence de l'espace de nommage src.choixEleves

Classes

– class choixElevesDialog
  implémente un dialogue permettant de choisir des élèves les propriétés importantes sont self.ok, vrai si on doit prendre en compte la liste sélectionnée, et le contenu de la liste des sélectionnés, dont on peut récupérer les élèves un par un à l'aide de self.pop()

Variables

– dictionary licence = {}
– tuple app = QApplication(sys.argv)
– tuple d = choixElevesDialog("../exemples/SCONET_test.xml", gestionnaire=gestClasse.Sconet)
– tuple i = d.pop()

6.4.1 Documentation des variables

6.4.1.1 tuple src.choixEleves.app = QApplication(sys.argv)

Définition à la ligne 254 du fichier choixEleves.py.

6.4.1.2 tuple src.choixEleves.d = choixElevesDialog("../exemples/SCONET_test.xml", gestionnaire=gestClasse.Sconet)

Définition à la ligne 255 du fichier choixEleves.py.

6.4.1.3 tuple src.choixEleves.i = d.pop()

Définition à la ligne 258 du fichier choixEleves.py.

6.4.1.4 dictionary src.choixEleves.licence = {}

Définition à la ligne 4 du fichier choixEleves.py.

6.5 Référence de l'espace de nommage src.chooseInSticks

Classes

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

Variables

– string licenceEn

6.5.1 Documentation des variables

6.5.1.1 string src.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.

6.6 Référence de l'espace de nommage src.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()

6.6.1 Documentation des variables

6.6.1.1 tuple src.copyToDialog1.app = QApplication(sys.argv)

Définition à la ligne 209 du fichier copyToDialog1.py.

6.6.1.2 string src.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.

6.6.1.3 tuple src.copyToDialog1.windows = copyToDialog1()

Définition à la ligne 210 du fichier copyToDialog1.py.

6.7 Référence de l'espace de nommage src.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 hasStudent
  vérifie qu'un étudiant est déjà connu
- 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

6.7.1 Documentation des fonctions

6.7.1.1 def src.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.

Voici le graphe des appelants de cette fonction :

6.7.1.2 def src.db.hasStudent (*student*)

vérifie qu'un étudiant est déjà connu

Paramètres

student	propriétaire du baladeur

Renvoie

True si le propriétaire existe déjà

Définition à la ligne 79 du fichier db.py.

6.7.1.3 def src.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 92 du fichier db.py.

Voici le graphe des appelants de cette fonction :

6.7.1.4 def src.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.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

6.7.1.5 def src.db.readPrefs ( )

renvoie les préférences de ScolaSync

Renvoie

un dictionnaire de préférences

Définition à la ligne 125 du fichier db.py.

6.7.1.6 def src.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 111 du fichier db.py.

6.7.1.7 def src.db.setWd (*newDir*)

définit le nouveau nom du répertoire de travail préféré.

Définition à la ligne 160 du fichier db.py.

6.7.1.8 def src.db.tattooList ( )

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

Définition à la ligne 101 du fichier db.py.

6.7.1.9 def src.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 186 du fichier db.py.

Voici le graphe d'appel pour cette fonction :

6.7.1.10 def src.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 170 du fichier db.py.

Voici le graphe d'appel pour cette fonction :

6.7.2 Documentation des variables

6.7.2.1 src.db.cursor = None

Définition à la ligne 30 du fichier db.py.

6.7.2.2 src.db.database = None

Définition à la ligne 29 du fichier db.py.

6.7.2.3 dictionary src.db.licence = {}

Définition à la ligne 4 du fichier db.py.

6.8 Référence de l'espace de nommage src.deviceListener

Classes

– class DeviceListener

Variables

– dictionary licence = {}

6.8.1 Documentation des variables

6.8.1.1 dictionary src.deviceListener.licence = {}

Définition à la ligne 4 du fichier deviceListener.py.

6.9 Référence de l'espace de nommage src.diskFull

Classes

– class mainWindow

Fonctions

– def sceneWithUsage

Variables

– dictionary licence = {}

6.9.1 Documentation des fonctions

6.9.1.1 def src.diskFull.sceneWithUsage (*parent, rect, percent*)

Paramètres

parent	le widget père
rect	le QRect contenant la scène
percent	pourcentage utilisé

Renvoie

une QGraphicsScene avec un symbole d'occupation du disque

Définition à la ligne 60 du fichier diskFull.py.

6.9.2 Documentation des variables

6.9.2.1 dictionary src.diskFull.licence = {}

Définition à la ligne 5 du fichier diskFull.py.

6.10 Référence de l'espace de nommage src.gestClasse

Classes

- class AbstractGestClasse
- class Sconet
  Une classe pour travailler avec des données Sconet.

6.11 Référence de l'espace de nommage src.gestclassetreeview

Classes

- class gestClasseTreeView

6.12 Référence de l'espace de nommage src.globaldef

Fonctions

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

Variables

- string licenceEn
  globaldef.py is part of the package scolasync.
- string userShareDir = "~/.scolasync"
- string logFileName = "~/.scolasync/scolasync.log"
- string markFileName = "~/.scolasync/marques.py"

6.12.1 Documentation des fonctions

6.12.1.1 def src.globaldef.firstdir (*l*)

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

Paramètres

l	la liste de propositions

Définition à la ligne 50 du fichier globaldef.py.

6.12.2 Documentation des variables

6.12.2.1 string src.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.

6.12.2.2 string src.globaldef.logFileName = "~/.scolasync/scolasync.log"

Définition à la ligne 37 du fichier globaldef.py.

6.12.2.3 string src.globaldef.markFileName = "~/.scolasync/marques.py"

Définition à la ligne 38 du fichier globaldef.py.

6.12.2.4 string src.globaldef.userShareDir = "~/.scolasync"

Définition à la ligne 36 du fichier globaldef.py.

6.13 Référence de l'espace de nommage src.help

Classes

– class helpWindow

Variables

– dictionary licence = {}

6.13.1 Documentation des variables

6.13.1.1 dictionary src.help.licence = {}

Définition à la ligne 5 du fichier help.py.

6.14 Référence de l'espace de nommage src.mainWindow

Classes

- class mainWindow
- class usbTableModel
  Un modèle de table pour des séries de clés USB.
- class CheckBoxDelegate
- class UsbDiskDelegate
  Classe pour identifier le baladeur dans le tableau.
- class DiskSizeDelegate
  Classe pour figurer la taille de la mémoire du baladeur.

Fonctions

- def registerCmd
  enregistre la commande cmd pour la partition donnée
- def CheckBoxRect

Variables

- dictionary licence = {}
- dictionary activeThreads = {}
- dictionary pastCommands = {}
- lastCommand = None

6.14.1 Documentation des fonctions

6.14.1.1 def src.mainWindow.CheckBoxRect (*view_item_style_options*)

Paramètres

view_item_style-_options	des options permettant de décider de la taille d'un rectangle

Renvoie

un QRect dimensionné selon les bonnes options

Définition à la ligne 888 du fichier mainWindow.py.

Voici le graphe des appelants de cette fonction :

6.14.1.2 def src.mainWindow.registerCmd (*cmd, partition*)

enregistre la commande cmd pour la partition donnée

Paramètres

cmd	une commande pour créer un thread t
partition	une partition

Définition à la ligne 53 du fichier mainWindow.py.

Voici le graphe des appelants de cette fonction :

6.14.2 Documentation des variables

6.14.2.1 dictionary src.mainWindow.activeThreads = $\{\}$

Définition à la ligne 41 du fichier mainWindow.py.

6.14.2.2 src.mainWindow.lastCommand = None

Définition à la ligne 45 du fichier mainWindow.py.

6.14.2.3 dictionary src.mainWindow.licence = {}

Définition à la ligne 5 du fichier mainWindow.py.

6.14.2.4 dictionary src.mainWindow.pastCommands = {}

Définition à la ligne 44 du fichier mainWindow.py.

6.15 Référence de l'espace de nommage src.marques

6.16 Référence de l'espace de nommage src.mytextbrowser

Classes

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

Variables

– dictionary licence = {}

6.16.1 Documentation des variables

6.16.1.1 dictionary src.mytextbrowser.licence = {}

Définition à la ligne 5 du fichier mytextbrowser.py.

6.17 Référence de l'espace de nommage src.nameAdrive

Classes

– class nameAdriveDialog
un dialogue pour renommer un baladeur, compte tenu d'une liste de noms disponibles

Variables

– dictionary licence = {}

6.17.1 Documentation des variables

6.17.1.1 dictionary src.nameAdrive.licence = {}

Définition à la ligne 4 du fichier nameAdrive.py.

6.18 Référence de l'espace de nommage src.notification

Classes

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

Variables

– dictionary licence = {}
– tuple notif

6.18.1 Documentation des variables

6.18.1.1 dictionary src.notification.licence = {}

Définition à la ligne 5 du fichier notification.py.

6.18.1.2 tuple src.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.

6.19 Référence de l'espace de nommage src.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 tattooInDir
Renvoie le tatouage pour un point de montage donné, quitte à le créer si nécessaire.
– def editRecord
édition de la base de données.

Variables

– dictionary licence = {}

6.19.1 Documentation des fonctions

6.19.1.1 def src.ownedUsbDisk.editRecord (*owd*, *hint* = " ")

édition de la base de données.

Paramètres

owd	une instance de ownedUsbDisk
hint	chaîne vide par défaut. Peut être le nom de l'ancien propriétaire

Définition à la ligne 71 du fichier ownedUsbDisk.py.

6.19.1.2 def src.ownedUsbDisk.tattooInDir (*mountPoint*)

Renvoie le tatouage pour un point de montage donné, quitte à le créer si nécessaire.

Paramètres

mountPoint	un point de montage de partition

Renvoie

le tatouage

Définition à la ligne 44 du fichier ownedUsbDisk.py.

Voici le graphe des appelants de cette fonction :

6.19.2 Documentation des variables

6.19.2.1 dictionary src.ownedUsbDisk.licence = $\{\}$

Définition à la ligne 4 du fichier ownedUsbDisk.py.

6.20 Référence de l'espace de nommage src.preferences

Classes

– class preferenceWindow

Variables

– dictionary licence = {}

6.20.1 Documentation des variables

6.20.1.1 dictionary src.preferences.licence = $\{\}$

Définition à la ligne 5 du fichier preferences.py.

6.21 Référence de l'espace de nommage src.scolasync

Fonctions

– def usage
  affiche le mode d'emploi à la console
– def run
  Le lancement de l'application.

Variables

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

6.21.1 Documentation des fonctions

6.21.1.1 def src.scolasync.run ( )

Le lancement de l'application.

Définition à la ligne 150 du fichier scolasync.py.

Voici le graphe d'appel pour cette fonction :

6.21.1.2 def src.scolasync.usage ( )

affiche le mode d'emploi à la console

Définition à la ligne 138 du fichier scolasync.py.

Voici le graphe des appelants de cette fonction :

6.21.2 Documentation des variables

6.21.2.1 dictionary src.scolasync.licence = {}

Définition à la ligne 85 du fichier scolasync.py.

6.21.2.2 string src.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.

6.21.2.3 string src.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.

6.22 Référence de l'espace de nommage src.sconet

Classes

– class Sconet
Une classe pour travailler avec des données Sconet.

Variables

– tuple s = Sconet("SCONET_test.xml")

6.22.1 Documentation des variables

6.22.1.1 tuple src.sconet.s = Sconet("SCONET_test.xml")

Définition à la ligne 77 du fichier sconet.py.

6.23 Référence de l'espace de nommage src.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()

6.23.1 Documentation des variables

6.23.1.1 dictionary src.usbDisk.licence = {}

Définition à la ligne 4 du fichier usbDisk.py.

6.23.1.2 string src.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.

6.23.1.3 tuple src.usbDisk.machin = Available()

Définition à la ligne 564 du fichier usbDisk.py.

6.24 Référence de l'espace de nommage src.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 threadMoveFromUSB
  Classe pour les threads déplaçant des fichiers depuis les clés USB.
- class threadDeleteInUSB
  Classe pour les threads effaçant des sous-arbres dans les clés USB.

Variables

- string licenceEn
- int _threadNumber = 0

6.24.1 Documentation des variables

6.24.1.1 int src.usbThread._threadNumber = 0

Définition à la ligne 27 du fichier usbThread.py.

6.24.1.2 string src.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.

6.25 Référence de l'espace de nommage src.version

Fonctions

– def major
– def minor
– def version

Variables

– dictionary licence = {}

6.25.1 Documentation des fonctions

6.25.1.1 def src.version.major ( )

Renvoie

le numéro majeur de version

Définition à la ligne 30 du fichier version.py.

Voici le graphe des appelants de cette fonction :

6.25.1.2 def src.version.minor ( )

Renvoie

le numéro mineur de version

Définition à la ligne 37 du fichier version.py.

Voici le graphe des appelants de cette fonction :

6.25.1.3 def src.version.version ( )

Renvoie

l'identifiant de la version

Définition à la ligne 44 du fichier version.py.

Voici le graphe d'appel pour cette fonction :

6.25.2 Documentation des variables

6.25.2.1 dictionary src.version.licence = {}

Définition à la ligne 4 du fichier version.py.

Chapitre 7

Documentation des classes

7.1 Référence de la classe src.gestClasse.AbstractGestClasse

Graphe d'héritage de src.gestClasse.AbstractGestClasse :

Fonctions membres publiques

- def __init__
  le constructeur
- def collectClasses
- def elevesDeClasse
- def unique_name
- def showable_name

7.1.1 Description détaillée

Définition à la ligne 13 du fichier gestClasse.py.

7.1.2 Documentation des constructeurs et destructeur

7.1.2.1 def src.gestClasse.AbstractGestClasse.__init__ (*self, f*)

le constructeur

Paramètres

f	le nom d'un fichier, ou un fichier ouvert en lecture qui contient les données permettant la gestion des classes d'un établissement scolaire

Réimplémentée dans src.gestClasse.Sconet.

Définition à la ligne 21 du fichier gestClasse.py.

7.1.3 Documentation des fonctions membres

7.1.3.1 def src.gestClasse.AbstractGestClasse.collectClasses (*self*)

Renvoie

une liste de noms de classes d'un établissement scolaire

Réimplémentée dans src.gestClasse.Sconet.

Définition à la ligne 28 du fichier gestClasse.py.

7.1.3.2 def src.gestClasse.AbstractGestClasse.elevesDeClasse (*self, cl*)

Paramètres

cl	une classe dans un établissement scolaire

Renvoie

une liste d'élèves (sous forme d'objets)

Réimplémentée dans src.gestClasse.Sconet.

Définition à la ligne 36 du fichier gestClasse.py.

7.1.3.3 def src.gestClasse.AbstractGestClasse.showable_name (*self, el*)

Paramètres

el	un objet élève

Renvoie

une chaîne unicode, pour nommer l'élève

Définition à la ligne 52 du fichier gestClasse.py.

7.1.3.4 def src.gestClasse.AbstractGestClasse.unique_name (*self, el*)

Paramètres

el	un objet élève

Renvoie

une chaîne unicode, unique dans l'établissement

Définition à la ligne 44 du fichier gestClasse.py.

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

7.2 Référence de la classe src.usbThread.abstractThreadUSB

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

Graphe d'héritage de src.usbThread.abstractThreadUSB :

Graphe de collaboration de src.usbThread.abstractThreadUSB :

Fonctions membres publiques

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

- def writeToLog
  Écrit un message dans le fichier de journalisation.
- def copytree
  Une version modifiée de shutil.copytree qui accepte que les repertoires destination soient déjà existants.
- 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
- parent

7.2.1 Description détaillée

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

Définition à la ligne 130 du fichier usbThread.py.

7.2.2 Documentation des constructeurs et destructeur

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

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
parent	un widget qui recevra de signaux en début et en fin d'exécution

Définition à la ligne 144 du fichier usbThread.py.

Voici le graphe d'appel pour cette fonction :

7.2.3 Documentation des fonctions membres

7.2.3.1 def src.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 226 du fichier usbThread.py.

Voici le graphe d'appel pour cette fonction :

7.2.3.2 def src.usbThread.abstractThreadUSB.copytree (*self, src, dst, symlinks* = `False`, *ignore* = `None`, *erase* = `False`, *errors* = `[]`)

Une version modifiée de shutil.copytree qui accepte que les repertoires destination soient déjà existants.

Cette source dérive de la documentation fournie avec Python 2.7

Paramètres

src	un nom de fichier ou de répertoire
dst	un nom de de répertoire (déjà existant ou à créer)
symlinks	vrai si on veut recopier les liens tels quels
ignore	une fonction qui construit une liste de fichiers à ignorer (profil : répertoire, liste de noms de fichiers -> liste de noms de fichiers à ignorer)
erase	s'il est vrai la source est effacée après copie réussie
errors	la liste d'erreurs déjà relevées jusque là

Renvoie

une liste d'erreurs éventuellement relevées, sinon une liste vide

Définition à la ligne 179 du fichier usbThread.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.2.3.3 def src.usbThread.abstractThreadUSB.threadType (*self*)

Renvoie

une chaîne courte qui informe sur le type de thread

Réimplémentée dans src.usbThread.threadCopyToUSB.

Définition à la ligne 241 du fichier usbThread.py.

Voici le graphe des appelants de cette fonction :

7.2.3.4 def src.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 src.usbThread.threadDeleteInUSB, src.usbThread.threadMoveFromUSB, src.usbThread.threadCopyFromUSB, et src.usbThread.threadCopyToUSB.

Définition à la ligne 253 du fichier usbThread.py.

Voici le graphe des appelants de cette fonction :

7.2.3.5 def src.usbThread.abstractThreadUSB.writeToLog (*self, msg*)

Écrit un message dans le fichier de journalisation.

Paramètres

msg	le message

Définition à la ligne 162 du fichier usbThread.py.

Voici le graphe des appelants de cette fonction :

7.2.4 Documentation des données membres

7.2.4.1 src.usbThread.abstractThreadUSB.cmd

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

Définition à la ligne 148 du fichier usbThread.py.

7.2.4.2 src.usbThread.abstractThreadUSB.dest

Définition à la ligne 153 du fichier usbThread.py.

7.2.4.3 src.usbThread.abstractThreadUSB.fileList

Définition à la ligne 151 du fichier usbThread.py.

7.2.4.4 src.usbThread.abstractThreadUSB.logfile

Définition à la ligne 154 du fichier usbThread.py.

7.2.4.5 src.usbThread.abstractThreadUSB.parent

Définition à la ligne 155 du fichier usbThread.py.

7.2.4.6 src.usbThread.abstractThreadUSB.subdir

Définition à la ligne 152 du fichier usbThread.py.

7.2.4.7 src.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

7.3 Référence de la classe src.ownedUsbDisk.Available

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

Graphe d'héritage de src.ownedUsbDisk.Available :

Graphe de collaboration de src.ownedUsbDisk.Available :

Fonctions membres publiques

– def __init__
  Le constructeur est un proxy pour usbDisk.Available.__init__ qui force la classe de disques à utiliser : en effet ici uDisk désigne ownedUsbDisk.uDisk.
– def finishInit
  Fin de l'initialisation : trouve les propriétaires des disques puis identifie les partitions FAT et les monte.

Attributs publics

– noLoop

7.3.1 Description détaillée

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

Les propriétaires sont recensés juste avant le montage des partions FAT.

Définition à la ligne 238 du fichier ownedUsbDisk.py.

7.3.2 Documentation des constructeurs et destructeur

7.3.2.1 def src.ownedUsbDisk.Available.__init__ (*self*, *checkable* = `False`, *access* = `"disk"`, *diskClass* = **uDisk**, *diskDict* = `None`, *noLoop* = `True`)

Le constructeur est un proxy pour usbDisk.Available.__init__ qui force la classe de disques à utiliser : en effet ici uDisk désigne ownedUsbDisk.uDisk.

Paramètres

checkable	True si on veut pouvoir sélectionner des disques en cochant
access	le mode d'accès : 'disk' ou 'firstFat'
diskClass	la classe d'objets à créer pour chaque disque
diskDict	un dictionnaire des disque maintenu par deviceListener
noLoop	doit être True pour éviter de lancer un dialogue

Définition à la ligne 251 du fichier ownedUsbDisk.py.

7.3.3 Documentation des fonctions membres

7.3.3.1 def src.ownedUsbDisk.Available.finishInit (*self*)

Fin de l'initialisation : trouve les propriétaires des disques puis identifie les partitions FAT et les monte.

Réimplémentée à partir de src.usbDisk.Available.

Définition à la ligne 260 du fichier ownedUsbDisk.py.

Voici le graphe d'appel pour cette fonction :

7.3.4 Documentation des données membres

7.3.4.1 src.ownedUsbDisk.Available.noLoop

Définition à la ligne 252 du fichier ownedUsbDisk.py.

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

7.4 Référence de la classe src.usbDisk.Available

une classe pour représenter la collection des disques USB connectés

Graphe d'héritage de src.usbDisk.Available :

Fonctions membres publiques

- def __init__
  Le constructeur.
- def finishInit
  Fin de l'initialisation.
- def mountFirstFats
  fabrique la liste des partitions FAT, monte les partitions FAT si elles ne le sont pas
- def __trunc__
- 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 des effets de bord :
- def hasDev

Attributs publics

- checkable
  print "GRRRR should use diskDict=", diskDict
- access
- bus
- disks
- enumDev
- firstFats
- fatPaths

7.4.1 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 371 du fichier usbDisk.py.

7.4.2 Documentation des constructeurs et destructeur

7.4.2.1 def src.usbDisk.Available.__init__ (*self*, *checkable* = `False`, *access* = `"disk"`, *diskClass* = **uDisk**, *diskDict* = `None`)

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.
diskClass	la classe de disques à créer
diskDict	un dictionnaire des disque maintenu par deviceListener

Définition à la ligne 384 du fichier usbDisk.py.

7.4.3 Documentation des fonctions membres

7.4.3.1 def src.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 499 du fichier usbDisk.py.

7.4.3.2 def src.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 511 du fichier usbDisk.py.

7.4.3.3 def src.usbDisk.Available.__str__ (*self*)

Fournit une représentation imprimable.

Renvoie

une représentation imprimable de la collection

Définition à la ligne 480 du fichier usbDisk.py.

7.4.3.4 def src.usbDisk.Available.__trunc__ (*self*)

Renvoie

le nombre de medias connectés

Définition à la ligne 434 du fichier usbDisk.py.

7.4.3.5 def src.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 444 du fichier usbDisk.py.

Voici le graphe d'appel pour cette fonction :

7.4.3.6 def src.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 454 du fichier usbDisk.py.

7.4.3.7 def src.usbDisk.Available.finishInit (*self*)

Fin de l'initialisation.

Réimplémentée dans src.ownedUsbDisk.Available.

Définition à la ligne 416 du fichier usbDisk.py.

Voici le graphe d'appel pour cette fonction :

7.4.3.8 def src.usbDisk.Available.getFirstFats (*self, setOwners* = `False`)

Facilite l'accès aux partitions de type DOS-FAT, et a des effets de bord :

- marque le disque avec l'uuid de la première partition FAT.
- construit une liste des chemins uDisk des FATs

Paramètres

setOwners	si égale à True, signale que la liste devra comporter des attributs de propriétaire de medias.

Renvoie

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

Définition à la ligne 529 du fichier usbDisk.py.

Voici le graphe des appelants de cette fonction :

7.4.3.9 def src.usbDisk.Available.hasDev (*self, dev*)

Paramètres

dev	un chemin comme /org/freedesktop/UDisks/devices/sdb3

Renvoie

True si la partition est dans la liste des partions disponibles

Définition à la ligne 554 du fichier usbDisk.py.

7.4.3.10 def src.usbDisk.Available.mountFirstFats (*self*)

fabrique la liste des partitions FAT, monte les partitions FAT si elles ne le sont pas

Définition à la ligne 424 du fichier usbDisk.py.

Voici le graphe des appelants de cette fonction :

7.4.3.11 def src.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 464 du fichier usbDisk.py.

Voici le graphe des appelants de cette fonction :

7.4.4 Documentation des données membres

7.4.4.1 src.usbDisk.Available.access

Définition à la ligne 387 du fichier usbDisk.py.

7.4.4.2 src.usbDisk.Available.bus

Définition à la ligne 388 du fichier usbDisk.py.

7.4.4.3 src.usbDisk.Available.checkable

print "GRRRR should use diskDict=", diskDict

Définition à la ligne 386 du fichier usbDisk.py.

7.4.4.4 src.usbDisk.Available.disks

Définition à la ligne 392 du fichier usbDisk.py.

7.4.4.5 src.usbDisk.Available.enumDev

Définition à la ligne 393 du fichier usbDisk.py.

7.4.4.6 src.usbDisk.Available.fatPaths

Définition à la ligne 531 du fichier usbDisk.py.

7.4.4.7 src.usbDisk.Available.firstFats

Définition à la ligne 425 du fichier usbDisk.py.

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

7.5 Référence de la classe src.mainWindow.CheckBoxDelegate

Graphe d'héritage de src.mainWindow.CheckBoxDelegate :

Graphe de collaboration de src.mainWindow.CheckBoxDelegate :

Fonctions membres publiques

– def __init__
– def paint
– def editorEvent

7.5.1 Description détaillée

Définition à la ligne 894 du fichier mainWindow.py.

7.5.2 Documentation des constructeurs et destructeur

7.5.2.1 def src.mainWindow.CheckBoxDelegate.__init__ (*self, parent*)

Définition à la ligne 895 du fichier mainWindow.py.

7.5.3 Documentation des fonctions membres

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

Définition à la ligne 909 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

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

Définition à la ligne 898 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

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

7.6 Référence de la classe src.checkBoxDialog.CheckBoxDialog

Un dialogue pour gérer les cases à cocher de l'application.

Graphe d'héritage de src.checkBoxDialog.CheckBoxDialog :

Graphe de collaboration de src.checkBoxDialog.CheckBoxDialog :

Fonctions membres publiques

- def __init__
  Le constructeur.
- def all
  Fait cocher tous les baladeurs.
- def toggle
  Fait inverser tous les boutons.
- def none
  Fait décocher tous les boutons.
- def esc
  termine le dialogue sans rien faire

Attributs publics

- mainWindow
- ui

7.6.1 Description détaillée

Un dialogue pour gérer les cases à cocher de l'application.

Définition à la ligne 32 du fichier checkBoxDialog.py.

7.6.2 Documentation des constructeurs et destructeur

7.6.2.1 def src.checkBoxDialog.CheckBoxDialog.__init__ (*self, parent* = `None`)

Le constructeur.

Paramètres

parent	un mainWindow, qui est censé contenir des données

Définition à la ligne 38 du fichier checkBoxDialog.py.

7.6.3 Documentation des fonctions membres

7.6.3.1 def src.checkBoxDialog.CheckBoxDialog.all (*self*)

Fait cocher tous les baladeurs.

Définition à la ligne 52 du fichier checkBoxDialog.py.

7.6.3.2 def src.checkBoxDialog.CheckBoxDialog.esc (*self*)

termine le dialogue sans rien faire

Définition à la ligne 76 du fichier checkBoxDialog.py.

7.6.3.3 def src.checkBoxDialog.CheckBoxDialog.none (*self*)

Fait décocher tous les boutons.

Définition à la ligne 68 du fichier checkBoxDialog.py.

7.6.3.4 def src.checkBoxDialog.CheckBoxDialog.toggle (*self*)

Fait inverser tous les boutons.

Définition à la ligne 60 du fichier checkBoxDialog.py.

7.6.4 Documentation des données membres

7.6.4.1 src.checkBoxDialog.CheckBoxDialog.mainWindow

Définition à la ligne 40 du fichier checkBoxDialog.py.

7.6.4.2 src.checkBoxDialog.CheckBoxDialog.ui

Définition à la ligne 41 du fichier checkBoxDialog.py.

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

7.7 Référence de la classe src.choixEleves.choixElevesDialog

implémente un dialogue permettant de choisir des élèves les propriétés importantes sont self.ok, vrai si on doit prendre en compte la liste sélectionnée, et le contenu de la liste des sélectionnés, dont on peut récupérer les élèves un par un à l'aide de self.pop()

Graphe d'héritage de src.choixEleves.choixElevesDialog :

Graphe de collaboration de src.choixEleves.choixElevesDialog :

Fonctions membres publiques

– def __init__
  le constructeur récupérer des données SCONET
– def fichierEleves
  choisit et ouvre un nouveau fichiers d'élèves
– def connecteGestionnaire
  met en place l'arbre des noms d'élèves
– def checkNum
  fonction de rappel utilisée quand on coche/décoche la case pour prendre en compte le numéro
– def replie
  replie toutes les classes du dialogue
– def coche
  coche toutes les cases d'élèves visibles
– def decoche
  décoche toutes les cases d'élèves, visibles ou cachées
– def updateParentIcon
  Met à jour l'icône du bouton d'activation dans l'application parente pour refléter la présence d'éléments dans la liste.
– def addToList
  ajoute les élèves cochés dans la liste (s'ils n'y sont pas déjà)
– def delInList
  retire les élèves de la liste quand ils y sont sélectionnés
– def pop
  renvoie et supprime le premier élément de la liste de noms ; si cette liste est vide, renvoie None
– def itemStrings
– def takeItem
  retire un item de la liste et le renvoie (pourvu qu'il y existe)
– def valid
  Prend acte de la validation.
– def escape
  Prend acte de l'abandon ; supprime les éléments de la liste et ferme le dialogue.
– def listeChoix
– def listeUnique_Names

Attributs publics

– ok
– ui
– prefs
– gestionnaire

7.7.1 Description détaillée

implémente un dialogue permettant de choisir des élèves les propriétés importantes sont self.ok, vrai si on doit prendre en compte la liste sélectionnée, et le contenu de la liste des sélectionnés, dont on peut récupérer les élèves un par un à l'aide de self.pop()

Définition à la ligne 40 du fichier choixEleves.py.

7.7.2 Documentation des constructeurs et destructeur

7.7.2.1 def src.choixEleves.choixElevesDialog.__init__ (*self*, *parent* = None, *gestionnaire* = gestClasse.Sconet)

le constructeur récupérer des données SCONET

Paramètres

parent	le widget parent
gestionnaire	le système censé gérer les données du fichier f

Définition à la ligne 49 du fichier choixEleves.py.

7.7.3 Documentation des fonctions membres

7.7.3.1 def src.choixEleves.choixElevesDialog.addToList (*self*)

ajoute les élèves cochés dans la liste (s'ils n'y sont pas déjà)

Définition à la ligne 154 du fichier choixEleves.py.

Voici le graphe d'appel pour cette fonction :

7.7.3.2 def src.choixEleves.choixElevesDialog.checkNum (*self*, *state*)

fonction de rappel utilisée quand on coche/décoche la case pour prendre en compte le numéro

Paramètres

state	: l'état coché ou décoché

Définition à la ligne 108 du fichier choixEleves.py.

7.7.3.3 def src.choixEleves.choixElevesDialog.coche (*self*)

coche toutes les cases d'élèves visibles

Définition à la ligne 127 du fichier choixEleves.py.

7.7.3.4 def src.choixEleves.choixElevesDialog.connecteGestionnaire (*self*, *renew* = False)

met en place l'arbre des noms d'élèves

Paramètres

renew	vrai si on veut vider tout l'arbre et recommencer

Définition à la ligne 91 du fichier choixEleves.py.

Voici le graphe des appelants de cette fonction :

7.7.3.5 def src.choixEleves.choixElevesDialog.decoche (*self*)

décoche toutes les cases d'élèves, visibles ou cachées

Définition à la ligne 136 du fichier choixEleves.py.

7.7.3.6 def src.choixEleves.choixElevesDialog.delInList (*self*)

retire les élèves de la liste quand ils y sont sélectionnés

Définition à la ligne 165 du fichier choixEleves.py.

Voici le graphe d'appel pour cette fonction :

7.7.3.7 def src.choixEleves.choixElevesDialog.escape (*self*)

Prend acte de l'abandon ; supprime les éléments de la liste et ferme le dialogue.

Définition à la ligne 227 du fichier choixEleves.py.

Voici le graphe d'appel pour cette fonction :

7.7.3.8 def src.choixEleves.choixElevesDialog.fichierEleves (*self*)

choisit et ouvre un nouveau fichiers d'élèves

Définition à la ligne 76 du fichier choixEleves.py.

Voici le graphe d'appel pour cette fonction :

7.7.3.9 def src.choixEleves.choixElevesDialog.itemStrings (*self*)

Renvoie

une liste des chaînes contenues dans les items

Définition à la ligne 192 du fichier choixEleves.py.

7.7.3.10 def src.choixEleves.choixElevesDialog.listeChoix (*self*)

Renvoie

la liste de QStandardItems sélectionnés

Définition à la ligne 239 du fichier choixEleves.py.

Voici le graphe des appelants de cette fonction :

7.7.3.11 def src.choixEleves.choixElevesDialog.listeUnique_Names (*self*)

Définition à la ligne 242 du fichier choixEleves.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.7.3.12 def src.choixEleves.choixElevesDialog.pop (*self*)

renvoie et supprime le premier élément de la liste de noms ; si cette liste est vide, renvoie None

Renvoie

un nom (QString) pour un baladeur, sinon None

Définition à la ligne 181 du fichier choixEleves.py.

Voici le graphe d'appel pour cette fonction :

7.7.3.13 def src.choixEleves.choixElevesDialog.replie (*self*)

replie toutes les classes du dialogue

Définition à la ligne 119 du fichier choixEleves.py.

7.7.3.14 def src.choixEleves.choixElevesDialog.takeItem (*self, item*)

retire un item de la liste et le renvoie (pourvu qu'il y existe)

Paramètres

une	chaîne donnant le texte d'un item à trouver

Renvoie

un nom (QString) pour un baladeur, sinon None

Définition à la ligne 204 du fichier choixEleves.py.

Voici le graphe d'appel pour cette fonction :

7.7.3.15 def src.choixEleves.choixElevesDialog.updateParentIcon (*self*)

Met à jour l'icône du bouton d'activation dans l'application parente pour refléter la présence d'éléments dans la liste.

Définition à la ligne 146 du fichier choixEleves.py.

Voici le graphe des appelants de cette fonction :

7.7.3.16 def src.choixEleves.choixElevesDialog.valid (*self*)

Prend acte de la validation.

Définition à la ligne 217 du fichier choixEleves.py.

7.7.4 Documentation des données membres

7.7.4.1 src.choixEleves.choixElevesDialog.gestionnaire

Définition à la ligne 55 du fichier choixEleves.py.

7.7.4.2 src.choixEleves.choixElevesDialog.ok

Définition à la ligne 51 du fichier choixEleves.py.

7.7.4.3 src.choixEleves.choixElevesDialog.prefs

Définition à la ligne 54 du fichier choixEleves.py.

7.7.4.4 src.choixEleves.choixElevesDialog.ui

Définition à la ligne 52 du fichier choixEleves.py.

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

7.8 Référence de la classe src.chooseInSticks.chooseDialog

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

Graphe d'héritage de src.chooseInSticks.chooseDialog :

Graphe de collaboration de src.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 sélectionnés ; dans le cas de chemins sans jokers (caractères ∗ ou ?), les chemins sont protégés par des guillemets, afin qu'ils soient adaptés à un shell POSIX.

Attributs publics

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

7.8.1 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.

7.8.2 Documentation des constructeurs et destructeur

7.8.2.1 def src.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.

7.8.3 Documentation des fonctions membres

7.8.3.1 def src.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 244 du fichier chooseInSticks.py.

7.8.3.2 def src.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 278 du fichier chooseInSticks.py.

Voici le graphe des appelants de cette fonction :

7.8.3.3 def src.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 151 du fichier chooseInSticks.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.8.3.4 def src.chooseInSticks.chooseDialog.changeWd (*self*)

changement du répertoire de travail

Définition à la ligne 185 du fichier chooseInSticks.py.

Voici le graphe d'appel pour cette fonction :

7.8.3.5 def src.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 110 du fichier chooseInSticks.py.

Voici le graphe des appelants de cette fonction :

7.8.3.6 def src.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 198 du fichier chooseInSticks.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.8.3.7 def src.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 236 du fichier chooseInSticks.py.

Voici le graphe d'appel pour cette fonction :

7.8.3.8 def src.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.

Voici le graphe d'appel pour cette fonction :

7.8.3.9 def src.chooseInSticks.chooseDialog.minus (*self*)

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

Définition à la ligne 262 du fichier chooseInSticks.py.

7.8.3.10 def src.chooseInSticks.chooseDialog.pathList (*self*)

renvoie la liste des chemins sélectionnés ; dans le cas de chemins sans jokers (caractères $*$ ou ?), les chemins sont protégés par des guillemets, afin qu'ils soient adaptés à un shell POSIX.

Renvoie

une liste de chemins, sous forme de QStrings

Définition à la ligne 294 du fichier chooseInSticks.py.

7.8.3.11 def src.chooseInSticks.chooseDialog.plus (*self*)

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

Définition à la ligne 252 du fichier chooseInSticks.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.8.3.12 def src.chooseInSticks.chooseDialog.selectedDiskMountPoint (*self*)

Renvoie

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

Définition à la ligne 162 du fichier chooseInSticks.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.8.3.13 def src.chooseInSticks.chooseDialog.selectedDiskOwner (*self*)

Renvoie

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

Définition à la ligne 174 du fichier chooseInSticks.py.

Voici le graphe des appelants de cette fonction :

7.8.4 Documentation des données membres

7.8.4.1 src.chooseInSticks.chooseDialog.mainWindow

Définition à la ligne 46 du fichier chooseInSticks.py.

7.8.4.2 src.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

7.9 Référence de la classe src.copyToDialog1.copyToDialog1

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

Graphe d'héritage de src.copyToDialog1.copyToDialog1 :

Graphe de collaboration de src.copyToDialog1.copyToDialog1 :

Fonctions membres publiques

- 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

7.9.1 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.

7.9.2 Documentation des fonctions membres

7.9.2.1 def src.copyToDialog1.copyToDialog1.cancel (*self*)

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

Définition à la ligne 74 du fichier copyToDialog1.py.

7.9.2.2 def src.copyToDialog1.copyToDialog1.cd (*self, index*)

Change le répertoire courant si possible.

Paramètres

ev	un évènement

Définition à la ligne 112 du fichier copyToDialog1.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.9.2.3 def src.copyToDialog1.copyToDialog1.changeWd (*self*)

changement du répertoire de travail

Définition à la ligne 66 du fichier copyToDialog1.py.

7.9.2.4 def src.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.

7.9.2.5 def src.copyToDialog1.copyToDialog1.displaySize (*self*)

Affiche la taille de la sélection courante.

Définition à la ligne 163 du fichier copyToDialog1.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.9.2.6 def src.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.

Voici le graphe d'appel pour cette fonction :

7.9.2.7 def src.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.

Voici le graphe d'appel pour cette fonction :

7.9.2.8 def src.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.

Voici le graphe des appelants de cette fonction :

7.9.2.9 def src.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.

Voici le graphe des appelants de cette fonction :

7.9.2.10 def src.copyToDialog1.copyToDialog1.setupFromListe (*self*)

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

Définition à la ligne 90 du fichier copyToDialog1.py.

Voici le graphe d'appel pour cette fonction :

7.9.2.11 def src.copyToDialog1.copyToDialog1.setupToListe (*self*)

Met en place un visionneur de fichierspour les fichiers reçus.

Définition à la ligne 124 du fichier copyToDialog1.py.

7.9.3 Documentation des données membres

7.9.3.1 src.copyToDialog1.copyToDialog1.mainWindow

Définition à la ligne 45 du fichier copyToDialog1.py.

7.9.3.2 src.copyToDialog1.copyToDialog1.ok

Définition à la ligne 83 du fichier copyToDialog1.py.

La documentation de cette classe a été générée à partir des fichiers suivants :
– src/copyToDialog1.py

7.10 Référence de la classe src.deviceListener.DeviceListener

Fonctions membres publiques

– def __init__
le constructeur
– def pollDevices
peuple le dictionnaire self.connectedVolumes avec les volumes actuellement gérés par dbus.
– def cbAdd
fonction de rappel pour une clé qu'on vient de brancher
– def cbChange
fonction de rappel pour une clé qui vient de changer de type
– def cbDel
fonction de rappel pour une clé retirée
– def vfatUsbPath
Détermine le chemin UDisks pour une partition vfat connectée par USB.
– def identify
Renvoie une identification de baladeur pour scolasync.
– def isVfatUsb
Décide si une partition est de type vfat, et connectée par USB.

Attributs publics

– bus
– manager
– interface
– connectedVolumes
– widget

7.10.1 Description détaillée

Définition à la ligne 29 du fichier deviceListener.py.

7.10.2 Documentation des constructeurs et destructeur

7.10.2.1 def src.deviceListener.DeviceListener.__init__ (*self, widget* = None)

le constructeur

Paramètres

widget	un QWidget qui s'intéresse aux disques amovibles

Définition à la ligne 35 du fichier deviceListener.py.

7.10.3 Documentation des fonctions membres

7.10.3.1 def src.deviceListener.DeviceListener.cbAdd (*self, path*)

fonction de rappel pour une clé qu'on vient de brancher

Paramètres

path	un chemin de type UDisks vers une partition

Définition à la ligne 69 du fichier deviceListener.py.

Voici le graphe d'appel pour cette fonction :

7.10.3.2 def src.deviceListener.DeviceListener.cbChange (*self, path*)

fonction de rappel pour une clé qui vient de changer de type

Paramètres

path	un chemin de type UDisks vers une partition

Définition à la ligne 83 du fichier deviceListener.py.

Voici le graphe d'appel pour cette fonction :

7.10.3.3 def src.deviceListener.DeviceListener.cbDel (*self, path*)

fonction de rappel pour une clé retirée

Paramètres

path	un chemin de type UDisks vers une partition

Définition à la ligne 98 du fichier deviceListener.py.

7.10.3.4 def src.deviceListener.DeviceListener.identify (*self, dev*)

Renvoie une identification de baladeur pour scolasync.

Paramètres

dev	est un objet dbus renvoyé par EnumerateDevices(), ou une simple chaine de caractères, clé dans le tableau connectedVolumes

Renvoie

un triplet (stickId, tatouage, uuid)

Définition à la ligne 129 du fichier deviceListener.py.

7.10.3.5 def src.deviceListener.DeviceListener.isVfatUsb (*self, o*)

Décide si une partition est de type vfat, et connectée par USB.

Paramètres

o	un objet dbus correspondant à une partition

Définition à la ligne 158 du fichier deviceListener.py.

Voici le graphe des appelants de cette fonction :

7.10.3.6 def src.deviceListener.DeviceListener.pollDevices (*self*)

peuple le dictionnaire self.connectedVolumes avec les volumes actuellement gérés par dbus.

Définition à la ligne 55 du fichier deviceListener.py.

Voici le graphe d'appel pour cette fonction :

7.10.3.7 def src.deviceListener.DeviceListener.vfatUsbPath (*self, dev*)

Détermine le chemin UDisks pour une partition vfat connectée par USB.

Paramètres

dev	est un objet dbus renvoyé par EnumerateDevices(), ou une simple chaine de caractères, clé dans le tableau connectedVolumes

Renvoie

un chemin vers le disque (selon UDisks), sinon ""

Définition à la ligne 114 du fichier deviceListener.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.10.4 Documentation des données membres

7.10.4.1 src.deviceListener.DeviceListener.bus

Définition à la ligne 36 du fichier deviceListener.py.

7.10.4.2 src.deviceListener.DeviceListener.connectedVolumes

Définition à la ligne 46 du fichier deviceListener.py.

7.10.4.3 src.deviceListener.DeviceListener.interface

Définition à la ligne 40 du fichier deviceListener.py.

7.10.4.4 src.deviceListener.DeviceListener.manager

Définition à la ligne 37 du fichier deviceListener.py.

7.10.4.5 src.deviceListener.DeviceListener.widget

Définition à la ligne 47 du fichier deviceListener.py.

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

7.11 Référence de la classe src.mainWindow.DiskSizeDelegate

Classe pour figurer la taille de la mémoire du baladeur.

Graphe d'héritage de src.mainWindow.DiskSizeDelegate :

Graphe de collaboration de src.mainWindow.DiskSizeDelegate :

Fonctions membres publiques

– def __init__
– def paint
– def val2txt

7.11.1 Description détaillée

Classe pour figurer la taille de la mémoire du baladeur.

Trace un petit secteur représentant la place occupée, puis affiche la place avec l'unité le plus parropriée.

Définition à la ligne 961 du fichier mainWindow.py.

7.11.2 Documentation des constructeurs et destructeur

7.11.2.1 def src.mainWindow.DiskSizeDelegate.__init__ (*self, parent*)

Définition à la ligne 962 du fichier mainWindow.py.

7.11.3 Documentation des fonctions membres

7.11.3.1 def src.mainWindow.DiskSizeDelegate.paint (*self, painter, option, index*)

Définition à la ligne 966 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

7.11.3.2 def src.mainWindow.DiskSizeDelegate.val2txt (*self, val*)

Renvoie

a string with a value with unit K, M, or G

Définition à la ligne 987 du fichier mainWindow.py.

Voici le graphe des appelants de cette fonction :

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

7.12 Référence de la classe src.gestclassetreeview.gestClasseTreeView

Graphe d'héritage de src.gestclassetreeview.gestClasseTreeView :

Graphe de collaboration de src.gestclassetreeview.gestClasseTreeView :

Fonctions membres publiques

– def __init__
Le constructeur.
– def connecteGestionnaire
– def expandedItems
– def allItems
– def checkedItems

Attributs publics

– gest
– root

7.12.1 Description détaillée

Définition à la ligne 9 du fichier gestclassetreeview.py.

7.12.2 Documentation des constructeurs et destructeur

7.12.2.1 def src.gestclassetreeview.gestClasseTreeView.__init__ (*self*, *parent* = `None`)

Le constructeur.

Paramètres

parent	un parent pour le widget

Définition à la ligne 15 du fichier gestclassetreeview.py.

7.12.3 Documentation des fonctions membres

7.12.3.1 def src.gestclassetreeview.gestClasseTreeView.allItems (*self*)

Renvoie

la liste de tous les élèves

Définition à la ligne 68 du fichier gestclassetreeview.py.

7.12.3.2 def src.gestclassetreeview.gestClasseTreeView.checkedItems (*self*)

Renvoie

la liste de tous les élèves sélectionnés

Définition à la ligne 82 du fichier gestclassetreeview.py.

7.12.3.3 def src.gestclassetreeview.gestClasseTreeView.connecteGestionnaire (*self, fichier, gestionnaire =* **gestClasse.Sconet**, *renew =* `False`)

Paramètres

fichier	le nom d'un fichier, ou un fichier ouvert en lecture, pour récupérer des données SCONET
gestionnaire	un gestionnaire pour exploiter les données du fichier
renew	vrai si on doit tout effacer avant de recommencer

Définition à la ligne 30 du fichier gestclassetreeview.py.

7.12.3.4 def src.gestclassetreeview.gestClasseTreeView.expandedItems (*self*)

Renvoie

la liste des items non repliés (donc visibles)

Définition à la ligne 53 du fichier gestclassetreeview.py.

7.12.4 Documentation des données membres

7.12.4.1 src.gestclassetreeview.gestClasseTreeView.gest

Définition à la ligne 17 du fichier gestclassetreeview.py.

7.12.4.2 src.gestclassetreeview.gestClasseTreeView.root

Définition à la ligne 20 du fichier gestclassetreeview.py.

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

7.13 Référence de la classe src.help.helpWindow

Graphe d'héritage de src.help.helpWindow :

Graphe de collaboration de src.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

7.13.1 Description détaillée

Définition à la ligne 32 du fichier help.py.

7.13.2 Documentation des constructeurs et destructeur

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

Le constructeur.

Définition à la ligne 37 du fichier help.py.

7.13.3 Documentation des fonctions membres

7.13.3.1 def src.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 53 du fichier help.py.

7.13.4 Documentation des données membres

7.13.4.1 src.help.helpWindow.ui

Définition à la ligne 40 du fichier help.py.

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

7.14 Référence de la classe src.mainWindow.mainWindow

Graphe d'héritage de src.mainWindow.mainWindow :

Graphe de collaboration de src.mainWindow.mainWindow :

Fonctions membres publiques

– def __init__
  Le constructeur.
– def checkModify
– def checkAll
  Coche tous les baladeurs.
– def checkToggle
  Inverse la coche des baladeurs.
– def checkNone
  Décoche tous les baladeurs.
– def namingADrive
  Gère un dialogue pour renommer un baladeur désigné par self.recentConnect.
– def deviceAdded
  fonction de rappel pour un medium ajouté
– def deviceRemoved

fonction de rappel pour un medium retiré
- def initRedoStuff
  Initialise des données pour le bouton central (refaire/stopper)
- def showEvent
  modification du comportement du widget original, pour démarrer le timer et les vérifications de baladeurs après construction de la fenêtre seulement
- def setTimer
  sets the main timer
- 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 manageCheckBoxes
  ouvre un dialogue pour permettre de gérer les cases à cocher globalement
- def diskSizeData
- def diskFromOwner
  trouve le disque qui correspond à un propriétaire
- def editOwner
  Édition du propriétaire d'une clé.
- def setAvailableNames
  Met à jour l'icône qui reflète la disponibilité de noms pour renommer automatiquement des baladeurs.
- 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 redoCmd
  Relance la dernière commande, mais en l'appliquant seulement aux baladeurs nouvellement branchés.
- def namesCmd
  montre le dialogue de choix de nouveaux noms à partir d'un fichier administratif.
- 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.
- def sameDiskData
- def flashLCD
  change le style de l'afficheur LCD pendant une fraction de seconde
- def normalLCD
  remet le style par défaut pour l'afficheur LCD

Attributs publics

- locale
- ui
- namesFullIcon
- namesEmptyIcon
- namesFullTip
- namesEmptyTip
- namesDialog
- recentConnect
- t
- proxy
- opts
- timer
- listener
- availableNames
- operations
- oldThreads

– flashTimer
– checkDisksLock
– iconRedo
– iconStop
– redoToolTip
– redoStatusTip
– stopToolTip
– stopStatusTip
– schoolFile
– workdir
– refreshEnabled
– refreshDelay
– manFileLocation
– checkable
– mv
– header
– visibleheader
– tm

7.14.1 Description détaillée

Définition à la ligne 61 du fichier mainWindow.py.

7.14.2 Documentation des constructeurs et destructeur

7.14.2.1 def src.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 69 du fichier mainWindow.py.

7.14.3 Documentation des fonctions membres

7.14.3.1 def src.mainWindow.mainWindow.applyPreferences (*self*)

Applique les préférences et les options de ligne de commande.

Définition à la ligne 242 du fichier mainWindow.py.

Voici le graphe des appelants de cette fonction :

7.14.3.2 def src.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 264 du fichier mainWindow.py.

7.14.3.3 def src.mainWindow.mainWindow.checkAll (*self*)

Coche tous les baladeurs.

Définition à la ligne 139 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

7.14.3.4 def src.mainWindow.mainWindow.checkDisks (*self, force* = `False`, *noLoop* = `True`)

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. À la fin de chaque vérification, un court flash est déclenché sur l'afficheur de nombre de baladeurs connectés et sa valeur est mise à jour.

Paramètres

force	pour forcer une mise à jour du tableau
noLoop	si False, on ne rentrera pas dans une boucle de Qt

Définition à la ligne 675 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.14.3.5 def src.mainWindow.mainWindow.checkModify (*self, boolFunc*)

Paramètres

boolfunc	une fonction pour décider du futur état de la coche étant donné l'état antérieur Modifie les coches des baladeurs

Définition à la ligne 126 du fichier mainWindow.py.

Voici le graphe des appelants de cette fonction :

7.14.3.6 def src.mainWindow.mainWindow.checkNone (*self*)

Décoche tous les baladeurs.

Définition à la ligne 153 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

7.14.3.7 def src.mainWindow.mainWindow.checkToggle (*self*)

Inverse la coche des baladeurs.

Définition à la ligne 146 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

7.14.3.8 def src.mainWindow.mainWindow.connectTableModel (*self, data*)

Connecte le modèle de table à la table.

Paramètres

data	les données de la table

Définition à la ligne 644 du fichier mainWindow.py.

Voici le graphe des appelants de cette fonction :

7.14.3.9 def src.mainWindow.mainWindow.copyFrom (*self*)

Lance l'action de copier depuis les clés USB.

Définition à la ligne 495 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

7.14.3.10 def src.mainWindow.mainWindow.copyTo (*self*)

Lance l'action de copier vers les clés USB.

Définition à la ligne 471 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

7.14.3.11 def src.mainWindow.mainWindow.delFiles (*self*)

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

Définition à la ligne 436 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

7.14.3.12 def src.mainWindow.mainWindow.deviceAdded (*self, s*)

fonction de rappel pour un medium ajouté

Paramètres

s	chemin UDisks, exemple : /org/freedesktop/UDisks/devices/sdb3

Définition à la ligne 181 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

7.14.3.13 def src.mainWindow.mainWindow.deviceRemoved (*self, s*)

fonction de rappel pour un medium retiré

Paramètres

s	une chaine de caractères du type /dev/sdxy

Définition à la ligne 195 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

7.14.3.14 def src.mainWindow.mainWindow.diskFromOwner (*self, student*)

trouve le disque qui correspond à un propriétaire

Paramètres

student	le propriétaire du disque

Renvoie

le disque correspondant à l'étudiant

Définition à la ligne 331 du fichier mainWindow.py.

Voici le graphe des appelants de cette fonction :

7.14.3.15 def src.mainWindow.mainWindow.diskSizeData (*self, rowOrDev*)

Paramètres

rowOrDev	a row number in the tableView, or a device string

Renvoie

a tuple dev,total,used,remain,pcent,path for the disk in the given row of the tableView (the tuple comes from the command df)

Définition à la ligne 314 du fichier mainWindow.py.

Voici le graphe des appelants de cette fonction :

7.14.3.16 def src.mainWindow.mainWindow.editOwner (*self, idx*)

Édition du propriétaire d'une clé.

Paramètres

idx	un QModelIndex qui pointe sur le propriétaire d'une clé

Définition à la ligne 352 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.14.3.17 def src.mainWindow.mainWindow.flashLCD (*self*)

change le style de l'afficheur LCD pendant une fraction de seconde

Définition à la ligne 715 du fichier mainWindow.py.

Voici le graphe des appelants de cette fonction :

7.14.3.18 def src.mainWindow.mainWindow.help (*self*)

Affiche le widget d'aide.

Définition à la ligne 598 du fichier mainWindow.py.

7.14.3.19 def src.mainWindow.mainWindow.initRedoStuff (*self*)

Initialise des données pour le bouton central (refaire/stopper)

Définition à la ligne 204 du fichier mainWindow.py.

7.14.3.20 def src.mainWindow.mainWindow.manageCheckBoxes (*self*)

ouvre un dialogue pour permettre de gérer les cases à cocher globalement

Définition à la ligne 303 du fichier mainWindow.py.

Voici le graphe des appelants de cette fonction :

7.14.3.21 def src.mainWindow.mainWindow.namesCmd (*self*)

montre le dialogue de choix de nouveaux noms à partir d'un fichier administratif.

Définition à la ligne 591 du fichier mainWindow.py.

7.14.3.22 def src.mainWindow.mainWindow.namingADrive (*self*)

Gère un dialogue pour renommer un baladeur désigné par self.recentConnect.

Définition à la ligne 161 du fichier mainWindow.py.

Voici le graphe des appelants de cette fonction :

7.14.3.23 def src.mainWindow.mainWindow.normalLCD (*self*)

remet le style par défaut pour l'afficheur LCD

Définition à la ligne 723 du fichier mainWindow.py.

7.14.3.24 def src.mainWindow.mainWindow.preference (*self*)

lance le dialogue des préférences

Définition à la ligne 422 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

7.14.3.25 def src.mainWindow.mainWindow.redoCmd (*self*)

Relance la dernière commande, mais en l'appliquant seulement aux baladeurs nouvellement branchés.

Définition à la ligne 560 du fichier mainWindow.py.

7.14.3.26 def src.mainWindow.mainWindow.sameDiskData (*self, one, two*)

Renvoie

True si les ensembles de uniqueId de one et two sont identiques

Définition à la ligne 708 du fichier mainWindow.py.

Voici le graphe des appelants de cette fonction :

7.14.3.27 def src.mainWindow.mainWindow.setAvailableNames (*self, available*)

Met à jour l'icône qui reflète la disponibilité de noms pour renommer automatiquement des baladeurs.

Paramètres

available	vrai s'il y a des noms disponibles pour renommer des baladeurs.

Définition à la ligne 367 du fichier mainWindow.py.

7.14.3.28 def src.mainWindow.mainWindow.setTimer (*self, enabled* = `True`)

sets the main timer

Définition à la ligne 232 du fichier mainWindow.py.

Voici le graphe des appelants de cette fonction :

7.14.3.29 def src.mainWindow.mainWindow.showEvent (*self, ev*)

modification du comportement du widget original, pour démarrer le timer et les vérifications de baladeurs après construction de la fenêtre seulement

Définition à la ligne 222 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

7.14.3.30 def src.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 273 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

7.14.3.31 def src.mainWindow.mainWindow.umount (*self*)

Démonte et détache les clés USB affichées.

Définition à la ligne 607 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

7.14.3.32 def src.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. Change l'icône du dialogue des noms selon qu'il reste ou non des noms disponibles dans le dialogue des noms.

Définition à la ligne 389 du fichier mainWindow.py.

Voici le graphe des appelants de cette fonction :

7.14.4 Documentation des données membres

7.14.4.1 src.mainWindow.mainWindow.availableNames

Définition à la ligne 94 du fichier mainWindow.py.

7.14.4.2 src.mainWindow.mainWindow.checkable

Définition à la ligne 252 du fichier mainWindow.py.

7.14.4.3 src.mainWindow.mainWindow.checkDisksLock

Définition à la ligne 100 du fichier mainWindow.py.

7.14.4.4 src.mainWindow.mainWindow.flashTimer

Définition à la ligne 98 du fichier mainWindow.py.

7.14.4.5 src.mainWindow.mainWindow.header

Définition à la ligne 256 du fichier mainWindow.py.

7.14.4.6 src.mainWindow.mainWindow.iconRedo

Définition à la ligne 206 du fichier mainWindow.py.

7.14.4.7 src.mainWindow.mainWindow.iconStop

Définition à la ligne 208 du fichier mainWindow.py.

7.14.4.8 src.mainWindow.mainWindow.listener

Définition à la ligne 92 du fichier mainWindow.py.

7.14.4.9 src.mainWindow.mainWindow.locale

Définition à la ligne 72 du fichier mainWindow.py.

7.14.4.10 src.mainWindow.mainWindow.manFileLocation

Définition à la ligne 249 du fichier mainWindow.py.

7.14.4.11 src.mainWindow.mainWindow.mv

Définition à la ligne 253 du fichier mainWindow.py.

7.14.4.12 src.mainWindow.mainWindow.namesDialog

Définition à la ligne 81 du fichier mainWindow.py.

7.14.4.13 src.mainWindow.mainWindow.namesEmptyIcon

Définition à la ligne 78 du fichier mainWindow.py.

7.14.4.14 src.mainWindow.mainWindow.namesEmptyTip

Définition à la ligne 80 du fichier mainWindow.py.

7.14.4.15 src.mainWindow.mainWindow.namesFullIcon

Définition à la ligne 77 du fichier mainWindow.py.

7.14.4.16 src.mainWindow.mainWindow.namesFullTip

Définition à la ligne 79 du fichier mainWindow.py.

7.14.4.17 src.mainWindow.mainWindow.oldThreads

Définition à la ligne 97 du fichier mainWindow.py.

7.14.4.18 src.mainWindow.mainWindow.operations

Définition à la ligne 96 du fichier mainWindow.py.

7.14.4.19 src.mainWindow.mainWindow.opts

Définition à la ligne 89 du fichier mainWindow.py.

7.14.4.20 src.mainWindow.mainWindow.proxy

Définition à la ligne 87 du fichier mainWindow.py.

7.14.4.21 src.mainWindow.mainWindow.recentConnect

Définition à la ligne 82 du fichier mainWindow.py.

7.14.4.22 src.mainWindow.mainWindow.redoStatusTip

Définition à la ligne 212 du fichier mainWindow.py.

7.14.4.23 src.mainWindow.mainWindow.redoToolTip

Définition à la ligne 211 du fichier mainWindow.py.

7.14.4.24 src.mainWindow.mainWindow.refreshDelay

Définition à la ligne 247 du fichier mainWindow.py.

7.14.4.25 src.mainWindow.mainWindow.refreshEnabled

Définition à la ligne 246 du fichier mainWindow.py.

7.14.4.26 src.mainWindow.mainWindow.schoolFile

Définition à la ligne 244 du fichier mainWindow.py.

7.14.4.27 src.mainWindow.mainWindow.stopStatusTip

Définition à la ligne 214 du fichier mainWindow.py.

7.14.4.28 src.mainWindow.mainWindow.stopToolTip

Définition à la ligne 213 du fichier mainWindow.py.

7.14.4.29 src.mainWindow.mainWindow.t

Définition à la ligne 86 du fichier mainWindow.py.

7.14.4.30 src.mainWindow.mainWindow.timer

Définition à la ligne 90 du fichier mainWindow.py.

7.14.4.31 src.mainWindow.mainWindow.tm

Définition à la ligne 651 du fichier mainWindow.py.

7.14.4.32 src.mainWindow.mainWindow.ui

Définition à la ligne 74 du fichier mainWindow.py.

7.14.4.33 src.mainWindow.mainWindow.visibleheader

Définition à la ligne 645 du fichier mainWindow.py.

7.14.4.34 src.mainWindow.mainWindow.workdir

Définition à la ligne 245 du fichier mainWindow.py.

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

7.15 Référence de la classe src.diskFull.mainWindow

Graphe d'héritage de src.diskFull.mainWindow :

Graphe de collaboration de src.diskFull.mainWindow :

Fonctions membres publiques

– def __init__
Le constructeur.

Attributs publics

– ui
– v
– total
– used

7.15.1 Description détaillée

Définition à la ligne 29 du fichier diskFull.py.

7.15.2 Documentation des constructeurs et destructeur

7.15.2.1 def src.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.

7.15.3 Documentation des données membres

7.15.3.1 src.diskFull.mainWindow.total

Définition à la ligne 47 du fichier diskFull.py.

7.15.3.2 src.diskFull.mainWindow.ui

Définition à la ligne 43 du fichier diskFull.py.

7.15.3.3 src.diskFull.mainWindow.used

Définition à la ligne 48 du fichier diskFull.py.

7.15.3.4 src.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

7.16 Référence de la classe src.mytextbrowser.myTextBrowser

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

Graphe d'héritage de src.mytextbrowser.myTextBrowser :

Graphe de collaboration de src.mytextbrowser.myTextBrowser :

Fonctions membres publiques

– def setSource
  lance Firefox en tâche de fond.
– def setHtml
  lien vers la méthode setSource originale

7.16.1 Description détaillée

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

Définition à la ligne 34 du fichier mytextbrowser.py.

7.16.2 Documentation des fonctions membres

7.16.2.1 def src.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.

7.16.2.2 def src.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

7.17 Référence de la classe src.nameAdrive.nameAdriveDialog

un dialogue pour renommer un baladeur, compte tenu d'une liste de noms disponibles

Graphe d'héritage de src.nameAdrive.nameAdriveDialog :

Graphe de collaboration de src.nameAdrive.nameAdriveDialog :

Fonctions membres publiques

– def __init__
  Le constructeur.
– def makeSelection
  Si l'ancien nom commence par un numéro, sélectionne le premier élément de la liste commençant par le même, sinon sélectionne le tout premier élément de la liste.
– def selectionChanged
  fonction de rappel quand la sélection change dans la liste ; recopie l'élément sélectionné comme nouveau nom de baladeur
– def ok
  fonction de rappel quand l'utilisateur valide le choix
– def esc
  fonction de rappel quand l'utilisateur cherche à échapper au choix

Attributs publics

– oldName
– nameList
– tattoo
– ui

– numPattern

7.17.1 Description détaillée

un dialogue pour renommer un baladeur, compte tenu d'une liste de noms disponibles

Définition à la ligne 36 du fichier nameAdrive.py.

7.17.2 Documentation des constructeurs et destructeur

7.17.2.1 def src.nameAdrive.nameAdriveDialog.__init__ (*self, parent* = None, *oldName* = "", *nameList* = [], *driveIdent* = None)

Le constructeur.

Paramètres

parent	le widget parent
oldName	le nom précédent du baladeur
nameList	une liste de noms disponibles
driveIdent	identité d'un baladeur sous forme d'un triplet (stickId, Uuid, Tattoo)

Définition à la ligne 46 du fichier nameAdrive.py.

7.17.3 Documentation des fonctions membres

7.17.3.1 def src.nameAdrive.nameAdriveDialog.esc (*self*)

fonction de rappel quand l'utilisateur cherche à échapper au choix

Définition à la ligne 114 du fichier nameAdrive.py.

7.17.3.2 def src.nameAdrive.nameAdriveDialog.makeSelection (*self*)

Si l'ancien nom commence par un numéro, sélectionne le premier élément de la liste commençant par le même, sinon sélectionne le tout premier élément de la liste.

Définition à la ligne 70 du fichier nameAdrive.py.

7.17.3.3 def src.nameAdrive.nameAdriveDialog.ok (*self*)

fonction de rappel quand l'utilisateur valide le choix

Définition à la ligne 101 du fichier nameAdrive.py.

7.17.3.4 def src.nameAdrive.nameAdriveDialog.selectionChanged (*self*)

fonction de rappel quand la sélection change dans la liste ; recopie l'élément sélectionné comme nouveau nom de baladeur

Définition à la ligne 90 du fichier nameAdrive.py.

7.17.4 Documentation des données membres

7.17.4.1 src.nameAdrive.nameAdriveDialog.nameList

Définition à la ligne 49 du fichier nameAdrive.py.

7.17.4.2 src.nameAdrive.nameAdriveDialog.numPattern

Définition à la ligne 57 du fichier nameAdrive.py.

7.17.4.3 src.nameAdrive.nameAdriveDialog.oldName

Définition à la ligne 48 du fichier nameAdrive.py.

7.17.4.4 src.nameAdrive.nameAdriveDialog.tattoo

Définition à la ligne 51 du fichier nameAdrive.py.

7.17.4.5 src.nameAdrive.nameAdriveDialog.ui

Définition à la ligne 52 du fichier nameAdrive.py.

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

7.18 Référence de la classe src.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

7.18.1 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.

7.18.2 Documentation des constructeurs et destructeur

7.18.2.1 def src.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 53 du fichier notification.py.

7.18.3 Documentation des fonctions membres

7.18.3.1 def src.notification.Notification.notify (*self*)

Définition à la ligne 70 du fichier notification.py.

7.18.4 Documentation des données membres

7.18.4.1 src.notification.Notification.actions

Définition à la ligne 59 du fichier notification.py.

7.18.4.2 src.notification.Notification.app_icon

Définition à la ligne 56 du fichier notification.py.

7.18.4.3 src.notification.Notification.app_name

Définition à la ligne 54 du fichier notification.py.

7.18.4.4 src.notification.Notification.body

Définition à la ligne 58 du fichier notification.py.

7.18.4.5 src.notification.Notification.expire_timeout

Définition à la ligne 61 du fichier notification.py.

7.18.4.6 src.notification.Notification.hints

Définition à la ligne 60 du fichier notification.py.

7.18.4.7 src.notification.Notification.interface

Définition à la ligne 66 du fichier notification.py.

7.18.4.8 src.notification.Notification.replaces_id

Définition à la ligne 55 du fichier notification.py.

7.18.4.9 src.notification.Notification.summary

Définition à la ligne 57 du fichier notification.py.

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

7.19 Référence de la classe src.preferences.preferenceWindow

Graphe d'héritage de src.preferences.preferenceWindow :

Graphe de collaboration de src.preferences.preferenceWindow :

Fonctions membres publiques

– def __init__
  Le constructeur.
– def enableDelay
  active ou désactive le glisseur pour modifier le délai de rafraichissement
– def updateRefreshLabel
  Met à jour l'affichage de la valeur du délai de rafraichissement.
– def values
– def setValues
  Met en place les préférences dans le dialogue.

Attributs publics

– ui

7.19.1 Description détaillée

Définition à la ligne 29 du fichier preferences.py.

7.19.2 Documentation des constructeurs et destructeur

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

Le constructeur.

Définition à la ligne 34 du fichier preferences.py.

7.19.3 Documentation des fonctions membres

7.19.3.1 def src.preferences.preferenceWindow.enableDelay (*self, state*)

active ou désactive le glisseur pour modifier le délai de rafraichissement

Paramètres

state	l'état coché ou décoché de la boîte qui contrôle le rafraichissement

Définition à la ligne 47 du fichier preferences.py.

7.19.3.2 def src.preferences.preferenceWindow.setValues (*self, prefs*)

Met en place les préférences dans le dialogue.

Paramètres

prefs	un dictionnaire de préférences

Définition à la ligne 81 du fichier preferences.py.

7.19.3.3 def src.preferences.preferenceWindow.updateRefreshLabel (*self, val*)

Met à jour l'affichage de la valeur du délai de rafraichissement.

Paramètres

val	un nombre entier qui exprime le délai en secondes

Définition à la ligne 55 du fichier preferences.py.

7.19.3.4 def src.preferences.preferenceWindow.values (*self*)

Renvoie

un dictionnaire de préférences

Définition à la ligne 65 du fichier preferences.py.

7.19.4 Documentation des données membres

7.19.4.1 src.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

7.20 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

7.21 Référence de la classe QDialog

Graphe d'héritage de QDialog :

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

7.22 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

7.23 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

7.24 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

7.25 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

7.26 Référence de la classe QTreeView

Graphe d'héritage de QTreeView :

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

7.27 Référence de la classe src.gestClasse.Sconet

Une classe pour travailler avec des données Sconet.

Graphe d'héritage de src.gestClasse.Sconet :

Graphe de collaboration de src.gestClasse.Sconet :

Fonctions membres publiques

– def __init__
  Le constructeur.
– def makeCompact
  removes useless thext nodes containing only spaces.
– def collectNullTexts
– def collectClasses
– def elevesDeClasse
– def eleveParID
  appends the "eleve" element to the list self.currentResult if self.currentID is matched

- def unIDEleveDeClasse
  appends the ID of an "eleve" to self.currentResult if he belongs to the class self.currentClassName
- def collectOneClass
  adds one class name to the set self.classes
- def unique_name
  a unique name for an "eleve", based on a few fields and on the ID
- def showable_name
- def elementsWalk
  implemente un parcour des éléments d'un arbre, pour y appliquer une procédure
- def __str__

Attributs publics

- donnees
- nullTexts
- classes
- currentResult
- currentClassName
- currentID

7.27.1 Description détaillée

Une classe pour travailler avec des données Sconet.

Définition à la ligne 59 du fichier gestClasse.py.

7.27.2 Documentation des constructeurs et destructeur

7.27.2.1 def src.gestClasse.Sconet.__init__ (*self, f*)

Le constructeur.

Paramètres

f	le nom d'un fichier, ou un fichier ouvert en lecture

Réimplémentée à partir de src.gestClasse.AbstractGestClasse.

Définition à la ligne 66 du fichier gestClasse.py.

7.27.3 Documentation des fonctions membres

7.27.3.1 def src.gestClasse.Sconet.__str__ (*self*)

Définition à la ligne 190 du fichier gestClasse.py.

7.27.3.2 def src.gestClasse.Sconet.collectClasses (*self*)

Renvoie

the list of classes containg students

Réimplémentée à partir de src.gestClasse.AbstractGestClasse.

Définition à la ligne 94 du fichier gestClasse.py.

7.27.3.3 def src.gestClasse.Sconet.collectNullTexts (*self, el*)

Définition à la ligne 84 du fichier gestClasse.py.

7.27.3.4 def src.gestClasse.Sconet.collectOneClass (*self, el*)

adds one class name to the set self.classes

Paramètres

el	an element

Définition à la ligne 145 du fichier gestClasse.py.

7.27.3.5 def src.gestClasse.Sconet.elementsWalk (*self, el, proc*)

implemente un parcour des éléments d'un arbre, pour y appliquer une procédure

Paramètres

el	un élément
proc	la procédure à appliquer (paramètres : l'élément)

Définition à la ligne 185 du fichier gestClasse.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.27.3.6 def src.gestClasse.Sconet.eleveParID (*self, el*)

appends the "eleve" element to the list self.currentResult if self.currentID is matched

Définition à la ligne 121 du fichier gestClasse.py.

7.27.3.7 def src.gestClasse.Sconet.elevesDeClasse (*self, className*)

Paramètres

className	name of a school class

Renvoie

list of "eleve" elements

Réimplémentée à partir de src.gestClasse.AbstractGestClasse.

Définition à la ligne 104 du fichier gestClasse.py.

7.27.3.8 def src.gestClasse.Sconet.makeCompact (*self*)

removes useless thext nodes containing only spaces.

Définition à la ligne 77 du fichier gestClasse.py.

7.27.3.9 def src.gestClasse.Sconet.showable_name (*self, el, fields* = ["NOM", *PRENOM*)

Paramètres

el	un objet élève
fields	les champs de donnée à exploiter

Renvoie

une chaîne unicode, pour nommer l'élève

Définition à la ligne 171 du fichier gestClasse.py.

7.27.3.10 def src.gestClasse.Sconet.unIDEleveDeClasse (*self, el*)

appends the ID of an "eleve" to self.currentResult if he belongs to the class self.currentClassName

Paramètres

el	an element

Définition à la ligne 131 du fichier gestClasse.py.

7.27.3.11 def src.gestClasse.Sconet.unique_name (*self, el, fields* = ["NOM", *PRENOM*)

a unique name for an "eleve", based on a few fields and on the ID

Paramètres

el	en "eleve" element
fields	the fields used to build the result a printable unique id

Définition à la ligne 158 du fichier gestClasse.py.

7.27.4 Documentation des données membres

7.27.4.1 src.gestClasse.Sconet.classes

Définition à la ligne 95 du fichier gestClasse.py.

7.27.4.2 src.gestClasse.Sconet.currentClassName

Définition à la ligne 106 du fichier gestClasse.py.

7.27.4.3 src.gestClasse.Sconet.currentID

Définition à la ligne 111 du fichier gestClasse.py.

7.27.4.4 src.gestClasse.Sconet.currentResult

Définition à la ligne 105 du fichier gestClasse.py.

7.27.4.5 src.gestClasse.Sconet.donnees

Définition à la ligne 70 du fichier gestClasse.py.

7.27.4.6 src.gestClasse.Sconet.nullTexts

Définition à la ligne 78 du fichier gestClasse.py.

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

7.28 Référence de la classe src.sconet.Sconet

Une classe pour travailler avec des données Sconet.

Fonctions membres publiques

- def __init__
  Le constructeur.
- def makeCompact
  removes useless thext nodes containing only spaces.
- def collectNullTexts
- def collectClasses
- def collectOneClass
- def elementsWalk
  implemente un parcour des éléments d'un arbre, pour y appliquer une procédure
- def __str__

Attributs publics

- donnees
- nullTexts
- classes

7.28.1 Description détaillée

Une classe pour travailler avec des données Sconet.

Définition à la ligne 10 du fichier sconet.py.

7.28.2 Documentation des constructeurs et destructeur

7.28.2.1 def src.sconet.Sconet.__init__ (*self, file*)

Le constructeur.

Paramètres

file	le nom d'un fichier, ou un fichier ouvert en lecture

Définition à la ligne 17 du fichier sconet.py.

7.28.3 Documentation des fonctions membres

7.28.3.1 def src.sconet.Sconet.__str__ (*self*)

Définition à la ligne 72 du fichier sconet.py.

7.28.3.2 def src.sconet.Sconet.collectClasses (*self*)

Renvoie

the list of classes containg students

Définition à la ligne 44 du fichier sconet.py.

7.28.3.3 def src.sconet.Sconet.collectNullTexts (*self, el*)

Définition à la ligne 34 du fichier sconet.py.

7.28.3.4 def src.sconet.Sconet.collectOneClass (*self, el*)

Renvoie

the name of a class if it is a class with students

Définition à la ligne 53 du fichier sconet.py.

7.28.3.5 def src.sconet.Sconet.elementsWalk (*self, el, proc*)

implemente un parcour des éléments d'un arbre, pour y appliquer une procédure

Paramètres

el	un élément
proc	la procédure à appliquer (paramètres : l'élément)

Définition à la ligne 67 du fichier sconet.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.28.3.6 def src.sconet.Sconet.makeCompact (*self*)

removes useless thext nodes containing only spaces.

Définition à la ligne 27 du fichier sconet.py.

7.28.4 Documentation des données membres

7.28.4.1 src.sconet.Sconet.classes

Définition à la ligne 45 du fichier sconet.py.

7.28.4.2 src.sconet.Sconet.donnees

Définition à la ligne 20 du fichier sconet.py.

7.28.4.3 src.sconet.Sconet.nullTexts

Définition à la ligne 28 du fichier sconet.py.

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

7.29 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

7.30 Référence de la classe src.usbThread.threadCopyFromUSB

Classe pour les threads copiant depuis les clés USB.

Graphe d'héritage de src.usbThread.threadCopyFromUSB :

Graphe de collaboration de src.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

7.30.1 Description détaillée

Classe pour les threads copiant depuis les clés USB.

Définition à la ligne 331 du fichier usbThread.py.

7.30.2 Documentation des constructeurs et destructeur

7.30.2.1 def src.usbThread.threadCopyFromUSB.__init__ (*self, ud, fileList, subdir* = " . ", *dest* = "/tmp", *rootPath* = "/", *logfile* = "/dev/null", *parent* = None)

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
parent	un widget qui recevra de signaux en début et en fin d'exécution

Définition à la ligne 346 du fichier usbThread.py.

7.30.3 Documentation des fonctions membres

7.30.3.1 def src.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, qui peut contenir des jokers
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 src.usbThread.abstractThreadUSB.

Définition à la ligne 363 du fichier usbThread.py.

Voici le graphe d'appel pour cette fonction :

7.30.4 Documentation des données membres

7.30.4.1 src.usbThread.threadCopyFromUSB.cmd

Réimplémentée à partir de src.usbThread.abstractThreadUSB.

Définition à la ligne 350 du fichier usbThread.py.

7.30.4.2 src.usbThread.threadCopyFromUSB.rootPath

Définition à la ligne 349 du fichier usbThread.py.

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

7.31 Référence de la classe src.usbThread.threadCopyToUSB

Classe pour les threads copiant vers les clés USB.

Graphe d'héritage de src.usbThread.threadCopyToUSB :

Graphe de collaboration de src.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

7.31.1 Description détaillée

Classe pour les threads copiant vers les clés USB.

Définition à la ligne 261 du fichier usbThread.py.

7.31.2 Documentation des constructeurs et destructeur

7.31.2.1 def src.usbThread.threadCopyToUSB.__init__ (*self, ud, fileList, subdir, logfile* = "/dev/null", *parent* = None)

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
parent	un widget qui recevra de signaux en début et en fin d'exécution

Définition à la ligne 274 du fichier usbThread.py.

7.31.3 Documentation des fonctions membres

7.31.3.1 def src.usbThread.threadCopyToUSB.threadType (*self*)

Renvoie

une chaîne courte qui informe sur le type de thread

Réimplémentée à partir de src.usbThread.abstractThreadUSB.

Définition à la ligne 282 du fichier usbThread.py.

7.31.3.2 def src.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 src.usbThread.abstractThreadUSB.

Définition à la ligne 297 du fichier usbThread.py.

Voici le graphe d'appel pour cette fonction :

7.31.4 Documentation des données membres

7.31.4.1 src.usbThread.threadCopyToUSB.cmd

Réimplémentée à partir de src.usbThread.abstractThreadUSB.

Définition à la ligne 276 du fichier usbThread.py.

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

7.32 Référence de la classe src.usbThread.threadDeleteInUSB

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

Graphe d'héritage de src.usbThread.threadDeleteInUSB :

Graphe de collaboration de src.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

7.32.1 Description détaillée

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

Définition à la ligne 478 du fichier usbThread.py.

7.32.2 Documentation des constructeurs et destructeur

7.32.2.1 def src.usbThread.threadDeleteInUSB.__init__ (*self, ud, fileList, subdir, logfile* = "/dev/null", *parent* = None)

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
parent	un widget qui recevra de signaux en début et en fin d'exécution

Définition à la ligne 491 du fichier usbThread.py.

7.32.3 Documentation des fonctions membres

7.32.3.1 def src.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 src.usbThread.abstractThreadUSB.

Définition à la ligne 509 du fichier usbThread.py.

Voici le graphe d'appel pour cette fonction :

7.32.4 Documentation des données membres

7.32.4.1 src.usbThread.threadDeleteInUSB.cmd

Réimplémentée à partir de src.usbThread.abstractThreadUSB.

Définition à la ligne 494 du fichier usbThread.py.

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

7.33 Référence de la classe src.usbThread.threadMoveFromUSB

Classe pour les threads déplaçant des fichiers depuis les clés USB.

Graphe d'héritage de src.usbThread.threadMoveFromUSB :

Graphe de collaboration de src.usbThread.threadMoveFromUSB :

Fonctions membres publiques

– def __init__
Constructeur Crée un thread pour déplacer 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

7.33.1 Description détaillée

Classe pour les threads déplaçant des fichiers depuis les clés USB.

Définition à la ligne 402 du fichier usbThread.py.

7.33.2 Documentation des constructeurs et destructeur

7.33.2.1 def src.usbThread.threadMoveFromUSB.__init__ (*self, ud, fileList, subdir* = " . ", *dest* = "/tmp", *rootPath* = "/", *logfile* = "/dev/null", *parent* = None)

Constructeur Crée un thread pour déplacer 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
parent	un widget qui recevra de signaux en début et en fin d'exécution

Définition à la ligne 417 du fichier usbThread.py.

7.33.3 Documentation des fonctions membres

7.33.3.1 def src.usbThread.threadMoveFromUSB.toDo (*self, ud, fileList, subdir, dest, logfile*)

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

Après chaque copie réussie la source est effacée. 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 src.usbThread.abstractThreadUSB.

Définition à la ligne 435 du fichier usbThread.py.

Voici le graphe d'appel pour cette fonction :

7.33.4 Documentation des données membres

7.33.4.1 src.usbThread.threadMoveFromUSB.cmd

Réimplémentée à partir de src.usbThread.abstractThreadUSB.

Définition à la ligne 421 du fichier usbThread.py.

7.33.4.2 src.usbThread.threadMoveFromUSB.rootPath

Définition à la ligne 420 du fichier usbThread.py.

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

7.34 Référence de la classe src.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.
– def threadSet
  renvoie l'ensemble des threads actifs

Attributs publics

– dico

7.34.1 Description détaillée

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

Définition à la ligne 33 du fichier usbThread.py.

7.34.2 Documentation des constructeurs et destructeur

7.34.2.1 def src.usbThread.ThreadRegister.__init__ (*self*)

Le constructure met en place un dictionnaire.

Définition à la ligne 39 du fichier usbThread.py.

7.34.3 Documentation des fonctions membres

7.34.3.1 def src.usbThread.ThreadRegister.__str__ (*self*)

Définition à la ligne 42 du fichier usbThread.py.

7.34.3.2 def src.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 72 du fichier usbThread.py.

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

Paramètres

ud	un disque
thread	un thread Dépile un thread pour le baladeur ud

Définition à la ligne 63 du fichier usbThread.py.

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

Paramètres

ud	un disque
thread	un thread Empile un thread pour le baladeur ud

Définition à la ligne 51 du fichier usbThread.py.

7.34.3.5 def src.usbThread.ThreadRegister.threadSet (*self*)

renvoie l'ensemble des threads actifs

Définition à la ligne 81 du fichier usbThread.py.

7.34.4 Documentation des données membres

7.34.4.1 src.usbThread.ThreadRegister.dico

Définition à la ligne 40 du fichier usbThread.py.

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

7.35 Référence de la classe src.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 src.ownedUsbDisk.uDisk :

Graphe de collaboration de src.ownedUsbDisk.uDisk :

Fonctions membres publiques

– def __init__
– def uniqueId
– 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)

7.35.1 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 85 du fichier ownedUsbDisk.py.

7.35.2 Documentation des constructeurs et destructeur

7.35.2.1 def src.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

Réimplémentée à partir de src.usbDisk.uDisk.

Définition à la ligne 92 du fichier ownedUsbDisk.py.

7.35.3 Documentation des fonctions membres

7.35.3.1 def src.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 n==-1, renvoie self ; si checkable est vrai, renvoie 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

Réimplémentée à partir de src.usbDisk.uDisk.

Définition à la ligne 190 du fichier ownedUsbDisk.py.

Voici le graphe d'appel pour cette fonction :

7.35.3.2 def src.ownedUsbDisk.uDisk.ensureOwner (*self, noLoop*)

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

Paramètres

noLoop	si True : ne fait pas de dialogue interactif

Renvoie

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

Définition à la ligne 221 du fichier ownedUsbDisk.py.

Voici le graphe d'appel pour cette fonction :

7.35.3.3 def src.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

Réimplémentée à partir de src.usbDisk.uDisk.

Définition à la ligne 161 du fichier ownedUsbDisk.py.

7.35.3.4 def src.ownedUsbDisk.uDisk.ownerByDb (*self*)

renvoie un nom de propriétaire dans tous les cas.

Définition à la ligne 171 du fichier ownedUsbDisk.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.35.3.5 def src.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 129 du fichier ownedUsbDisk.py.

7.35.3.6 def src.ownedUsbDisk.uDisk.tattoo (*self*)

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

Renvoie

un tatouage, supposément unique.

Définition à la ligne 113 du fichier ownedUsbDisk.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.35.3.7 def src.ownedUsbDisk.uDisk.uniqueId (*self*)

Renvoie

un identifiant unique, composé du nom du propriétaire suivi du tatouage

Réimplémentée à partir de src.usbDisk.uDisk.

Définition à la ligne 105 du fichier ownedUsbDisk.py.

Voici le graphe d'appel pour cette fonction :

7.35.3.8 def src.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 145 du fichier ownedUsbDisk.py.

7.35.4 Documentation des données membres

7.35.4.1 tuple src.ownedUsbDisk.uDisk.headers = staticmethod(headers) [static]

Réimplémentée à partir de src.usbDisk.uDisk.

Définition à la ligne 212 du fichier ownedUsbDisk.py.

7.35.4.2 src.ownedUsbDisk.uDisk.model

Définition à la ligne 97 du fichier ownedUsbDisk.py.

7.35.4.3 src.ownedUsbDisk.uDisk.owner

Définition à la ligne 95 du fichier ownedUsbDisk.py.

7.35.4.4 src.ownedUsbDisk.uDisk.vendor

Définition à la ligne 96 du fichier ownedUsbDisk.py.

7.35.4.5 src.ownedUsbDisk.uDisk.visibleDirs

Définition à la ligne 98 du fichier ownedUsbDisk.py.

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

7.36 Référence de la classe src.usbDisk.uDisk

une classe pour représenter un disque ou une partition.

Graphe d'héritage de src.usbDisk.uDisk :

Graphe de collaboration de src.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 isMounted
- 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)

7.36.1 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.

7.36.2 Documentation des constructeurs et destructeur

7.36.2.1 def src.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

Réimplémentée dans src.ownedUsbDisk.uDisk.

Définition à la ligne 51 du fichier usbDisk.py.

7.36.3 Documentation des fonctions membres

7.36.3.1 def src.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

Réimplémentée dans src.ownedUsbDisk.uDisk.

Définition à la ligne 286 du fichier usbDisk.py.

Voici le graphe d'appel pour cette fonction :

7.36.3.2 def src.usbDisk.uDisk.__str__ (*self*)

Fournit une représentation imprimable.

Renvoie

une représentation imprimable de l'instance

Définition à la ligne 148 du fichier usbDisk.py.

Voici le graphe d'appel pour cette fonction :

7.36.3.3 def src.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.

7.36.3.4 def src.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 332 du fichier usbDisk.py.

Voici le graphe d'appel pour cette fonction :

7.36.3.5 def src.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.

Voici le graphe d'appel pour cette fonction :

7.36.3.6 def src.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.

Voici le graphe des appelants de cette fonction :

7.36.3.7 def src.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 323 du fichier usbDisk.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.36.3.8 def src.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.

Voici le graphe des appelants de cette fonction :

7.36.3.9 def src.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

Réimplémentée dans src.ownedUsbDisk.uDisk.

Définition à la ligne 104 du fichier usbDisk.py.

7.36.3.10 def src.usbDisk.uDisk.isDosFat (*self*)

Permet de reconnaitre les partitions DOS-FAT.

Renvoie

True dans le cas d'une partition FAT16 ou FAT32

Définition à la ligne 199 du fichier usbDisk.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.36.3.11 def src.usbDisk.uDisk.isMounted (*self*)

Renvoie

True si le disque ou la partion est montée

Définition à la ligne 206 du fichier usbDisk.py.

Voici le graphe d'appel pour cette fonction :

7.36.3.12 def src.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.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.36.3.13 def src.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.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.36.3.14 def src.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 260 du fichier usbDisk.py.

Voici le graphe d'appel pour cette fonction :

7.36.3.15 def src.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.

Voici le graphe d'appel pour cette fonction :

7.36.3.16 def src.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 306 du fichier usbDisk.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.36.3.17 def src.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.

Voici le graphe des appelants de cette fonction :

7.36.3.18 def src.usbDisk.uDisk.uniqueId (*self*)

renvoie un identifiant unique.

Dans cette classe, cette fonction est synonyme de getFatUuid

Renvoie

un identifiant supposément unique

Réimplémentée dans src.ownedUsbDisk.uDisk.

Définition à la ligne 91 du fichier usbDisk.py.

Voici le graphe d'appel pour cette fonction :

7.36.3.19 def src.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 270 du fichier usbDisk.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.36.3.20 def src.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 214 du fichier usbDisk.py.

Voici le graphe d'appel pour cette fonction :

Voici le graphe des appelants de cette fonction :

7.36.4 Documentation des données membres

7.36.4.1 src.usbDisk.uDisk.checkable

Définition à la ligne 56 du fichier usbDisk.py.

7.36.4.2 src.usbDisk.uDisk.device

Définition à la ligne 53 du fichier usbDisk.py.

7.36.4.3 src.usbDisk.uDisk.device_prop

Définition à la ligne 54 du fichier usbDisk.py.

7.36.4.4 src.usbDisk.uDisk.fatuuid

Définition à la ligne 59 du fichier usbDisk.py.

7.36.4.5 src.usbDisk.uDisk.firstFat

Définition à la ligne 60 du fichier usbDisk.py.

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

Réimplémentée dans src.ownedUsbDisk.uDisk.

Définition à la ligne 111 du fichier usbDisk.py.

7.36.4.7 src.usbDisk.uDisk.path

Définition à la ligne 52 du fichier usbDisk.py.

7.36.4.8 src.usbDisk.uDisk.selected

Définition à la ligne 55 du fichier usbDisk.py.

7.36.4.9 src.usbDisk.uDisk.stickid

Définition à la ligne 57 du fichier usbDisk.py.

7.36.4.10 src.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

7.37 Référence de la classe src.mainWindow.UsbDiskDelegate

Classe pour identifier le baladeur dans le tableau.

Graphe d'héritage de src.mainWindow.UsbDiskDelegate :

Graphe de collaboration de src.mainWindow.UsbDiskDelegate :

Fonctions membres publiques

– def __init__
– def paint

Attributs publics

– okPixmap
– busyPixmap

7.37.1 Description détaillée

Classe pour identifier le baladeur dans le tableau.

La routine de rendu à l'écran trace une petite icône et le nom du propriétaire à côté.

Définition à la ligne 931 du fichier mainWindow.py.

7.37.2 Documentation des constructeurs et destructeur

7.37.2.1 def src.mainWindow.UsbDiskDelegate.__init__ (*self, parent*)

Définition à la ligne 932 du fichier mainWindow.py.

7.37.3 Documentation des fonctions membres

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

Définition à la ligne 937 du fichier mainWindow.py.

7.37.4 Documentation des données membres

7.37.4.1 src.mainWindow.UsbDiskDelegate.busyPixmap

Définition à la ligne 935 du fichier mainWindow.py.

7.37.4.2 src.mainWindow.UsbDiskDelegate.okPixmap

Définition à la ligne 934 du fichier mainWindow.py.

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

7.38 Référence de la classe src.mainWindow.usbTableModel

Un modèle de table pour des séries de clés USB.

Graphe d'héritage de src.mainWindow.usbTableModel :

Graphe de collaboration de src.mainWindow.usbTableModel :

Fonctions membres publiques

– def __init__

- def pushCmd
  fonction de rappel déclenchée par les threads (au commencement)
- def popCmd
  fonction de rappel déclenchée par les threads (à la fin)
- def updateOwnerColumn
  force la mise à jour de la colonne des propriétaires
- def rowCount
  un QModelIndex
- def columnCount
  un QModelIndex
- def setData
- def partition
- def data
- def headerData
- def sort
  Sort table by given column number.

Attributs publics

- header
- donnees
- checkable
- pere

7.38.1 Description détaillée

Un modèle de table pour des séries de clés USB.

Définition à la ligne 730 du fichier mainWindow.py.

7.38.2 Documentation des constructeurs et destructeur

7.38.2.1 def src.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 739 du fichier mainWindow.py.

7.38.3 Documentation des fonctions membres

7.38.3.1 def src.mainWindow.usbTableModel.columnCount (*self, parent*)

un QModelIndex

Définition à la ligne 816 du fichier mainWindow.py.

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

Définition à la ligne 834 du fichier mainWindow.py.

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

Définition à la ligne 863 du fichier mainWindow.py.

7.38.3.4 def src.mainWindow.usbTableModel.partition (*self, index*)

Paramètres

index	in QModelIndex

Renvoie

la partition pointée par index

Définition à la ligne 831 du fichier mainWindow.py.

7.38.3.5 def src.mainWindow.usbTableModel.popCmd (*self, owner, cmd*)

fonction de rappel déclenchée par les threads (à la fin)

Paramètres

owner	le propriétaire du baladeur associé au thread
cmd	la commande shell effectuée sur ce baladeur

Définition à la ligne 771 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

7.38.3.6 def src.mainWindow.usbTableModel.pushCmd (*self, owner, cmd*)

fonction de rappel déclenchée par les threads (au commencement)

Paramètres

owner	le propriétaire du baladeur associé au thread
cmd	la commande shell effectuée sur ce baladeur

Définition à la ligne 754 du fichier mainWindow.py.

Voici le graphe d'appel pour cette fonction :

7.38.3.7 def src.mainWindow.usbTableModel.rowCount (*self, parent*)

un QModelIndex

Définition à la ligne 809 du fichier mainWindow.py.

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

Définition à la ligne 819 du fichier mainWindow.py.

7.38.3.9 def src.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 875 du fichier mainWindow.py.

7.38.3.10 def src.mainWindow.usbTableModel.updateOwnerColumn (*self*)

force la mise à jour de la colonne des propriétaires

Définition à la ligne 797 du fichier mainWindow.py.

Voici le graphe des appelants de cette fonction :

7.38.4 Documentation des données membres

7.38.4.1 src.mainWindow.usbTableModel.checkable

Définition à la ligne 743 du fichier mainWindow.py.

7.38.4.2 src.mainWindow.usbTableModel.donnees

Définition à la ligne 742 du fichier mainWindow.py.

7.38.4.3 src.mainWindow.usbTableModel.header

Définition à la ligne 741 du fichier mainWindow.py.

7.38.4.4 src.mainWindow.usbTableModel.pere

Définition à la ligne 744 du fichier mainWindow.py.

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

Chapitre 8

Documentation des fichiers

8.1 Référence du fichier src/__init__.py

Paquetages

– namespace src

8.2 Référence du fichier src/checkBoxDialog.py

Classes

– class src.checkBoxDialog.CheckBoxDialog
Un dialogue pour gérer les cases à cocher de l'application.

Paquetages

– namespace src.checkBoxDialog

Variables

– string src.checkBoxDialog.licenceEn

8.3 Référence du fichier src/choixEleves.py

Classes

– class src.choixEleves.choixElevesDialog
implémente un dialogue permettant de choisir des élèves les propriétés importantes sont self.ok, vrai si on doit prendre en compte la liste sélectionnée, et le contenu de la liste des sélectionnés, dont on peut récupérer les élèves un par un à l'aide de self.pop()

Paquetages

– namespace src.choixEleves

Variables

– dictionary src.choixEleves.licence = {}

- tuple src.choixEleves.app = QApplication(sys.argv)
- tuple src.choixEleves.d = choixElevesDialog("../exemples/SCONET_test.xml", gestionnaire=gestClasse.Sconet)
- tuple src.choixEleves.i = d.pop()

8.4 Référence du fichier src/chooseInSticks.py

Classes

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

Paquetages

- namespace src.chooseInSticks

Variables

- string src.chooseInSticks.licenceEn

8.5 Référence du fichier src/copyToDialog1.py

Classes

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

Paquetages

- namespace src.copyToDialog1

Variables

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

8.6 Référence du fichier src/db.py

Paquetages

- namespace src.db

Fonctions

- def src.db.openDb
  Ouverture de la base de données de l'application, et création si nécessaire.
- def src.db.checkVersion
  Vérifie si la base de données reste compatible.
- def src.db.hasStudent
  vérifie qu'un étudiant est déjà connu
- def src.db.knowsId
  dit si une clé USB est déjà connue
- def src.db.tattooList
  Renvoie la liste des tatouages connus de la base de données.

- def src.db.readStudent
  renvoie l'étudiant qui possède une clé USB
- def src.db.readPrefs
  renvoie les préférences de ScolaSync
- def src.db.setWd
  définit le nouveau nom du répertoire de travail préféré.
- def src.db.writeStudent
  inscrit un étudiant comme propriétaire d'une clé USB
- def src.db.writePrefs
  inscrit les préférences

Variables

- dictionary src.db.licence = {}
- src.db.database = None
- src.db.cursor = None

8.7 Référence du fichier src/deviceListener.py

Classes

- class src.deviceListener.DeviceListener

Paquetages

- namespace src.deviceListener

Variables

- dictionary src.deviceListener.licence = {}

8.8 Référence du fichier src/diskFull.py

Classes

- class src.diskFull.mainWindow

Paquetages

- namespace src.diskFull

Fonctions

- def src.diskFull.sceneWithUsage

Variables

- dictionary src.diskFull.licence = {}

8.9 Référence du fichier src/gestClasse.py

Classes

- class src.gestClasse.AbstractGestClasse
- class src.gestClasse.Sconet
  Une classe pour travailler avec des données Sconet.

Paquetages

- namespace src.gestClasse

8.10 Référence du fichier src/gestclassetreeview.py

Classes

- class src.gestclassetreeview.gestClasseTreeView

Paquetages

- namespace src.gestclassetreeview

8.11 Référence du fichier src/globaldef.py

Paquetages

- namespace src.globaldef

Fonctions

- def src.globaldef.firstdir
  Renvoie le premier répertoire existant d'une liste de propositions.

Variables

- string src.globaldef.licenceEn
  globaldef.py is part of the package scolasync.
- string src.globaldef.userShareDir = "~/.scolasync"
- string src.globaldef.logFileName = "~/.scolasync/scolasync.log"
- string src.globaldef.markFileName = "~/.scolasync/marques.py"

8.12 Référence du fichier src/help.py

Classes

- class src.help.helpWindow

Paquetages

- namespace src.help

Variables

- dictionary src.help.licence = {}

8.13 Référence du fichier src/mainWindow.py

Classes

- class src.mainWindow.mainWindow
- class src.mainWindow.usbTableModel
  Un modèle de table pour des séries de clés USB.
- class src.mainWindow.CheckBoxDelegate
- class src.mainWindow.UsbDiskDelegate
  Classe pour identifier le baladeur dans le tableau.
- class src.mainWindow.DiskSizeDelegate
  Classe pour figurer la taille de la mémoire du baladeur.

Paquetages

- namespace src.mainWindow

Fonctions

- def src.mainWindow.registerCmd
  enregistre la commande cmd pour la partition donnée
- def src.mainWindow.CheckBoxRect

Variables

- dictionary src.mainWindow.licence = {}
- dictionary src.mainWindow.activeThreads = {}
- dictionary src.mainWindow.pastCommands = {}
- src.mainWindow.lastCommand = None

8.14 Référence du fichier src/marques.py

Paquetages

- namespace src.marques

8.15 Référence du fichier src/mytextbrowser.py

Classes

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

Paquetages

- namespace src.mytextbrowser

Variables

- dictionary src.mytextbrowser.licence = {}

8.16 Référence du fichier src/nameAdrive.py

Classes

– class src.nameAdrive.nameAdriveDialog
un dialogue pour renommer un baladeur, compte tenu d'une liste de noms disponibles

Paquetages

– namespace src.nameAdrive

Variables

– dictionary src.nameAdrive.licence = {}

8.17 Référence du fichier src/notification.py

Classes

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

Paquetages

– namespace src.notification

Variables

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

8.18 Référence du fichier src/ownedUsbDisk.py

Classes

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

Paquetages

– namespace src.ownedUsbDisk

Fonctions

– def src.ownedUsbDisk.tattooInDir
Renvoie le tatouage pour un point de montage donné, quitte à le créer si nécessaire.
– def src.ownedUsbDisk.editRecord
édition de la base de données.

Variables

– dictionary src.ownedUsbDisk.licence = {}

8.19 Référence du fichier src/preferences.py

Classes

– class src.preferences.preferenceWindow

Paquetages

– namespace src.preferences

Variables

– dictionary src.preferences.licence = {}

8.20 Référence du fichier src/scolasync.py

Paquetages

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

Fonctions

– def src.scolasync.usage
affiche le mode d'emploi à la console
– def src.scolasync.run
Le lancement de l'application.

Variables

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

8.21 Référence du fichier src/sconet.py

Classes

– class src.sconet.Sconet
Une classe pour travailler avec des données Sconet.

Paquetages

– namespace src.sconet

Variables

– tuple src.sconet.s = Sconet("SCONET_test.xml")

8.22 Référence du fichier src/usbDisk.py

Classes

- class src.usbDisk.uDisk
  une classe pour représenter un disque ou une partition.
- class src.usbDisk.Available
  une classe pour représenter la collection des disques USB connectés

Paquetages

- namespace src.usbDisk

Variables

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

8.23 Référence du fichier src/usbThread.py

Classes

- class src.usbThread.ThreadRegister
  Une classe pour tenir un registre des threads concernant les baladeurs.
- class src.usbThread.abstractThreadUSB
  Une classe abstraite Cette classe sert de creuset pour les classe servant aux copies et aux effacement.
- class src.usbThread.threadCopyToUSB
  Classe pour les threads copiant vers les clés USB.
- class src.usbThread.threadCopyFromUSB
  Classe pour les threads copiant depuis les clés USB.
- class src.usbThread.threadMoveFromUSB
  Classe pour les threads déplaçant des fichiers depuis les clés USB.
- class src.usbThread.threadDeleteInUSB
  Classe pour les threads effaçant des sous-arbres dans les clés USB.

Paquetages

- namespace src.usbThread

Variables

- string src.usbThread.licenceEn
- int src.usbThread._threadNumber = 0

8.24 Référence du fichier src/version.py

Paquetages

- namespace src.version

Fonctions

- def src.version.major
- def src.version.minor
- def src.version.version

Variables

– dictionary src.version.licence = {}