description: | API documentation for modules: GesMag, GesMag.db, GesMag.main, GesMag.users.

lang: en

classoption: oneside geometry: margin=1in papersize: a4

linkcolor: blue links-as-notes: true ...

Namespace `GesMag` {#id}

Sub-modules

Module `GesMag.db` {#id}

Classes

Class `BaseDeDonnees` {#id}

```
class BaseDeDonnees(
    urlBaseDeDonnee: str
)
```

Gère la base de donnée.

Methods

Method `affichageResultat` {#id}

```
def affichageResultat(
    self,
    curseur: sqlite3.Cursor
) -> list
```

Affiche le résultat d'une requête.

Method `creerConnexion` {#id}

```
def creerConnexion(
    self,
```

```
    path: str
)
```

Connexion à une base de donnée SQLite.

Method `fichierExiste` {#id}

```
def fichierExiste(
    self,
    path: str
) -> bool
```

Vérifie qu'un fichier existe.

Method `requete` {#id}

```
def requete(
    self,
    requete: str,
    valeurs=None
)
```

Envois une requête vers la base de données.

Module `GesMag.main` {#id}

Functions

Function `dimensionsFenetre` {#id}

```
def dimensionsFenetre(
    fenetre,
    taille: tuple
)
```

Permet de définir une fenetre centrer sur l'écran

Classes

Class `GesMag` {#id}

```
class GesMag
```

Programme de Gestion d'une caise de magasin.

Methods

Method `connexion` {#id}

```
def connexion(
    self,
    utilisateur: str,
    motDePasse: str
)
```

Gère la connexion aux différentes interfaces de l'application.

Method `demarrer` {#id}

```
def demarrer(
    self
) -> None
```

Lance le programme GesMag.

Method `motDePasseCorrect` {#id}

```
def motDePasseCorrect(
    self,
    motDPasse: str
) -> tuple
```

Détermine si un mot de passe suit la politique du programme ou non.

Module `GesMag.users` {#id}

Classes

Class `Utilisateurs` {#id}

```
class Utilisateurs
```

Gère une table "utilisateurs" pour une base de donnée donné.

Ancestors (in MRO)

- db.BaseDeDonnees

Methods

Method `ajoutUtilisateurs` {#id}

```
def ajoutUtilisateurs(
    self,
    pseudo: str,
    passe: str,
    metier: int,
    nom: str,
    prenom: str,
    naissance: str,
    adresse: str,
    postal: str
) -> list
```

Ajoute un utilisateur et retourne l'ID de ce dernier.

Method `creationTable` {#id}

```
def creationTable(
    self
) -> None
```

Créer la table qui stocker les utilisateurs.

Method `listUtilisateurs` {#id}

```
def listUtilisateurs(
    self
) -> list
```

Retourne la liste des nom d'utilisateurs.

Method `suppressionUtilisateurs` {#id}

```
def suppressionUtilisateurs(
    self,
    pseudo: str
) -> None
```

Supprime un utilisateur.

Method `verificationIdentifiants` {#id}

```
def verificationIdentifiants(
    self,
    pseudo: str,
    motDePasse: str
) -> bool
```

Renvoie vrai ou faux si les identifiants données sont bons.

Generated by *pdoc* 0.10.0 (https://pdoc3.github.io).