Projet d'application : P4++

Généré par Doxygen 1.9.1

Chapitre 1

ProjetApplicationConceptionLogicielle

Lors de ce projet, nous devons réaliser un puissance 4. Cependant, ce puissance 4 sera amélioré, les joueurs pourront changer la taille de la grille et le nombre de jetons requis pour gagner.

De plus, le joueur aura des bonus permettant de renverser la partie ou de se donner un avantage, par exemple, il pourra avoir une bombe pour supprimer une ligne ou une colonne, ou alors retourner la grille. Nous allons aussi apporter de la personnalisation au joueur, il pourra changer la couleur des jetons et de la grille.

Ce projet se divise en trois parties : l'algorithmie, le génie logiciel ainsi que la réalisation de l'interface graphique. Ces trois parties sont essentielles à ce projet.

1.1 Modules externes utilisés :

- `Numpy`
- `Inspect`
- `Tkinter`
- `argparse`

1.2 Installation

- Pour installer le jeu, il suffit de cloner le dépôt git avec la commande suivante :
  `git clone https://github.com/matteolanglois/projetApplicationConceptionLogicielle.git`
- Ensuite, il faut se placer dans le dossier du jeu :
  `cd projetApplicationConceptionLogicielle`
- Enfin, il faut installer les modules externes utilisés :
  `pip3 install -r requirements.txt`

1.3 Installation bis

- Après avoir cloné le dépôt git et vous être placé dans le dossier du jeu, il est possible d'exécuter le fichier install.bat (si vous êtes sous windows) ou install.sh (si vous êtes sous linux)
- Ce fichier va installer les modules externes utilisés et lancer le jeu.

1.4 Lancement

- Pour lancer le jeu, il suffit de lancer le fichier `main.py` avec python3.
- Si vous lancez le jeu avec l'argument '–cli', le jeu se lancera en ligne de commande.
- Si vous lancez le jeu sans argument, le jeu se lancera avec l'interface graphique.

1.5 Fonctionnalités

1.5.1 Algorithmie

- [x] Vérification de la victoire
- [x] Vérification de la jouabilité
- [x] Algorithme MinMax
- [x] Ajout de pièce
- [x] Algorithme de gestion de partie en ligne de commande
- [x] Implémentation du undo/redo

1.5.2 Interface graphique

- [x] Page d'accueil
- [x] Page de paramètres
- [x] Sauvegarde des paramètres
- [x] Page de jeu
- [x] Affichage de la grille
- [x] Affichage des jetons
- [x] Utilisation des bonus
- [x] Affichage de la victoire/Défaite
- [ ] Affichage de l'aide
- [x] Affichage du tour du joueur
- [x] Interface responsive

1.5.3 Personnalisation

- [x] Personnalisation des jetons
- [x] Personnalisation de la couleur de la grille
- [x] Personnalisation de la taille de la grille
- [x] Personnalisation du nombre de jetons à aligner pour gagner
- [x] Personnalisation de la difficulté de l'IA

1.5.4 Bonus

- [x] Utilisation des bonus en ligne de commande
- [x] Utilisation des bonus dans la fenêtre
- [x] Bonus d'inversion de la grille
- [x] Bonus de suppression de ligne pleine
- [x] Bonus d'aide avec MinMax
- [x] Bonus de retournement de la grille

1.6 Expliquation du code

1.6.1 Documentation

- La documentation du code se trouve dans le dossier `docs` à la racine du projet.
- Elle peut être générée avec Doxygen, le fichier de configuration doxyFile se trouve à la racine du projet.
- Pour générer la documentation, il suffit de lancer la commande suivante :
  ```
  doxygen doxyFile
  ```

1.6.2 Convention de nommage

- Les variables et fonctions sont nommées en snake_case.
- Les fonctions commencent par un acronyme désignant le fichier et le module dans lequel elles se trouvent.
- Les noms de variables commencent par leur type :
  - `b_` pour les booléens
  - `i_` pour les entiers
  - `f_` pour les flottants
  - `s_` pour les chaînes de caractères
  - `t_` pour les listes
  - `t_` pour les tuples
  - `npa_` pour les tableaux numpy
  - `tkl_` pour les labels tkinter
  - `tkb_` pour les boutons tkinter
  - `tkc_` pour les canvas tkinter
  - `tksb_` pour les spinbox tkinter
  - `tkf_` pour les frames tkinter
  - `tks_` pour les scales tkinter
  - `tksv_` pour les stringvar tkinter

Chapitre 2

Liste des choses à faire

Membre src::controller::ctrl_pageJeu.cpj_use_bonus (tk.Frame tkf_page_jeu)
Vérifier si le bonus a entraîné une victoire

Chapitre 3

Liste des tests

Membre tests::test_grid.tg_init_grille ()
Vérifie que la npa_grille est bien initialisée avec des 0 partout avec la bonne taille
Vérifie que toutes les combinaisons de 2 nombres de la liste *liste_tailles* sont testées

Membre tests::test_puissanceQuatre.tp_ajout_piece ()
Vérifie que la fonction renvoie les bonnes coordonnées quand le joueur joue
Vérifie que la fonction renvoie les bonnes coordonnées quand le bot joue
Vérifie que la fonction renvoie les bonnes coordonnées quand le joueur joue normalement
Vérifie que la fonction ne renvoie pas de coordonnées quand on ne peut pas ajouter de pièce

Membre tests::test_puissanceQuatre.tp_verif_colonne ()
Vérifie que la fonction renvoie True si la colonne est vide
Vérifie que la fonction renvoie False si la colonne est pleine
Vérifie que la fonction renvoie True si la colonne est presque pleine
Vérifie que la fonction renvoie True si la colonne est un peu remplie
Vérifie que la fonction renvoie False si la npa_grille est pleine

Membre tests::test_puissanceQuatre.tp_victoire_colonne ()
Vérifie que la fonction renvoie True si le joueur gagne
Vérifie que la fonction renvoie True si le bot gagne
Vérifie que la fonction renvoie False si personne ne gagne

Membre tests::test_puissanceQuatre.tp_victoire_diago ()
Vérifie que la fonction renvoie True si le joueur gagne
Vérifie que la fonction renvoie True si le bot gagne
Vérifie que la fonction renvoie False si personne ne gagne

Membre tests::test_puissanceQuatre.tp_victoire_ligne ()
Vérifie que la fonction renvoie True si le joueur gagne
Vérifie que la fonction renvoie True si le bot gagne
Vérifie que la fonction renvoie False si personne ne gagne

Chapitre 4

Index des espaces de nommage

4.1 Paquetages

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

Chapitre 5

Documentation des espaces de nommage

5.1 Référence de l'espace de nommage src

Espaces de nommage

- controller
- puissanceQuatre
- utils
- view

5.2 Référence de l'espace de nommage src.controller

Espaces de nommage

- ctrl_main
  Un programme qui joue au jeu puissance 4++.
- ctrl_pageAccueil
  Un programme qui joue au jeu puissance 4++.
- ctrl_PageBonus
  Un programme qui joue au jeu puissance 4++.
- ctrl_pageJeu
  Un programme qui joue au jeu puissance 4++.
- ctrl_pageParametres
  Un programme qui joue au jeu puissance 4++.

5.3 Référence de l'espace de nommage src.controller.ctrl_main

Un programme qui joue au jeu puissance 4++.

Fonctions

- def cm_init ()
  Initialise la fenêtre de jeu.
- def cm_quit (tk.Tk tk_root)
  Ferme la fenêtre de jeu.
- tk.Menu cm_menu (tk.Frame tk_frame, bool b_in_game)
  Crée le menu de la fenêtre.
- def cm_page_play (tk.Tk tk_root, tk.Frame tkf_old_frame)
  Fonction permettant de passer à la fenêtre de jeu.
- def cm_page_bonus (tk.Tk tk_root, tk.Frame tkf_old_frame)
  Fonction permettant de passer à la fenêtre de jeu.
- def cm_page_parameters (tk.Tk tk_root, tk.Frame tkf_old_frame)
  Fonction permettant de passer à la fenêtre de paramètres.
- def cm_page_accueil (tk.Tk tk_root, tk.Frame tkf_old_frame)
  Fonction permettant de passer à la fenêtre d'accueil.
- def cm_ended_game (str str_message, tk.Frame tkf_old_frame)
  Fonction permettant de passer à la fenêtre de fin de partie.
- def cm_warning (str str_message)

Affiche un message d'avertissement.
- def cm_info (str str_message)
  Affiche un message d'information.
- def cm_update (tk.Tk tk_root)
  Met à jour la fenêtre.

5.3.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Un module qui gère la fenêtre principale du jeu.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :
- tkinter
- numpy
- inspect

  Ce module gère la fenêtre principale du jeu et les transitions entre les différentes fenêtres.

5.3.2 Documentation des fonctions

5.3.2.1 cm_ended_game()

```
def src.controller.ctrl_main.cm_ended_game (
            str str_message,
            tk.Frame tkf_old_frame )
```

Fonction permettant de passer à la fenêtre de fin de partie.
Cette fonction appelle la fonction de suppression du cadre de la dernière fenêtre et la fonction d'initialisation de la page de fin de partie. Elle permet de passer à la fenêtre de fin de partie.

Précondition

tkf_old_frame initialisé

Paramètres

str_message	Message à afficher
tkf_old_frame	Le cadre de la dernière fenêtre

5.3.2.2 cm_info()

```
def src.controller.ctrl_main.cm_info (
            str str_message )
```

Affiche un message d'information.
Cette fonction appelle la fonction d'affichage d'un message d'information.

Paramètres

str_message	Message à afficher

5.3.2.3 cm_init()

```
def src.controller.ctrl_main.cm_init ( )
```

Initialise la fenêtre de jeu.

Cette fonction initialise la fenêtre de jeu et lance la boucle principale.

Précondition

tk_root initialisé

Postcondition

boucle principale lancée

Variables :

— tk_root : Fenêtre principale

5.3.2.4 cm_menu()

```
tk.Menu src.controller.ctrl_main.cm_menu (
            tk.Frame tk_frame,
            bool b_in_game )
```

Crée le menu de la fenêtre.
Cette fonction appelle la fonction de création du menu de la fenêtre. Elle renvoie le menu créé. Elle prend en paramètre le cadre de la dernière fenêtre et un booléen indiquant si le joueur est en jeu ou non.

Précondition

tkf_old_frame initialisé

Paramètres

tk_frame	Le cadre de la dernière fenêtre
b_in_game	Booléen indiquant si le joueur est en jeu ou non

Renvoie

Menu de la fenêtre

Postcondition

Menu créé

5.3.2.5 cm_page_accueil()

```
def src.controller.ctrl_main.cm_page_accueil (
            tk.Tk tk_root,
            tk.Frame tkf_old_frame )
```

Fonction permettant de passer à la fenêtre d'accueil.
Cette fonction appelle la fonction de suppression du cadre de la dernière fenêtre et la fonction d'initialisation de la page d'accueil. Elle permet de passer à la fenêtre d'accueil.

Précondition

tk_root initialisé

tkf_old_frame initialisé

Paramètres

tk_root	La fenêtre principale
tkf_old_frame	Le cadre de la dernière fenêtre

Postcondition

tkf_old_frame détruit
fenêtre d'accueil initialisée

5.3.2.6 cm_page_bonus()

```
def src.controller.ctrl_main.cm_page_bonus (
            tk.Tk tk_root,
            tk.Frame tkf_old_frame )
```

Fonction permettant de passer à la fenêtre de jeu.
Cette fonction appelle la fonction de suppression du cadre de la dernière fenêtre et la fonction d'initialisation de la page de jeu. Elle permet de passer à la fenêtre de jeu.

Précondition

tk_root initialisé
tkf_old_frame initialisé

Paramètres

tk_root	La fenêtre principale
tkf_old_frame	Le cadre de la dernière fenêtre

Postcondition

tkf_old_frame détruit
fenêtre de jeu initialisée

5.3.2.7 cm_page_parameters()

```
def src.controller.ctrl_main.cm_page_parameters (
            tk.Tk tk_root,
            tk.Frame tkf_old_frame )
```

Fonction permettant de passer à la fenêtre de paramètres.
Cette fonction appelle la fonction de suppression du cadre de la dernière fenêtre et la fonction d'initialisation de la page de paramètres. Elle permet de passer à la fenêtre de paramètres.

Précondition

tk_root initialisé
tkf_old_frame initialisé

Paramètres

tk_root	La fenêtre principale
tkf_old_frame	Le cadre de la dernière fenêtre

Postcondition

tkf_old_frame détruit
fenêtre de paramètres initialisée

5.3.2.8 cm_page_play()

```
def src.controller.ctrl_main.cm_page_play (
```

```
            tk.Tk tk_root,
            tk.Frame tkf_old_frame )
```

Fonction permettant de passer à la fenêtre de jeu.
Cette fonction appelle la fonction de suppression du cadre de la dernière fenêtre et la fonction d'initialisation de la page de jeu. Elle permet de passer à la fenêtre de jeu.

Précondition

tk_root initialisé
tkf_old_frame initialisé

Paramètres

tk_root	La fenêtre principale
tkf_old_frame	Le cadre de la dernière fenêtre

Postcondition

tkf_old_frame détruit
fenêtre de jeu initialisée

5.3.2.9 cm_quit()

```
def src.controller.ctrl_main.cm_quit (
            tk.Tk tk_root )
```

Ferme la fenêtre de jeu.
Cette fonction appelle la fonction de fermeture de la fenêtre de jeu.

Précondition

tk_root initialisé

Paramètres

tk_root	Fenêtre principale

5.3.2.10 cm_update()

```
def src.controller.ctrl_main.cm_update (
            tk.Tk tk_root )
```

Met à jour la fenêtre.
Cette fonction appelle la fonction de mise à jour de la fenêtre principale.

Précondition

tk_root initialisé

Paramètres

tk_root	Fenêtre principale

5.3.2.11 cm_warning()

```
def src.controller.ctrl_main.cm_warning (
            str str_message )
```

Affiche un message d'avertissement.
Cette fonction appelle la fonction d'affichage d'un message d'avertissement.

Paramètres

str_message	Message à afficher

5.4 Référence de l'espace de nommage src.controller.ctrl_pageAccueil

Un programme qui joue au jeu puissance 4++.

Fonctions

- def cpa_init (tk.Tk tk_root)
  Initialise la page d'accueil.
- def cpa_play (tk.Tk tk_root, tk.Frame tkf_frame)
  Lance une partie.

5.4.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Un module qui gère la page d'accueil du jeu.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

- tkinter
- numpy
- inspect

  Ce module gère la page d'accueil du jeu

5.4.2 Documentation des fonctions

5.4.2.1 cpa_init()

```
def src.controller.ctrl_pageAccueil.cpa_init (
            tk.Tk tk_root )
```

Initialise la page d'accueil.
Cette fonction initialise la page d'accueil du jeu.

Précondition

tk_root initialisé

Paramètres

tk_root	Fenêtre principale

5.4.2.2 cpa_play()

```
def src.controller.ctrl_pageAccueil.cpa_play (
            tk.Tk tk_root,
            tk.Frame tkf_frame )
```

Lance une partie.

Cette fonction lance une partie de puissance 4++. Elle est appelée lorsque l'utilisateur clique sur le bouton "Jouer".

Précondition

tk_root initialisé
tkf_frame initialisé

Paramètres

tk_root	Fenêtre principale
tkf_frame	Frame de la page d'accueil

Postcondition

Fenêtre de choix du bonus ouverte

5.5 Référence de l'espace de nommage src.controller.ctrl_PageBonus

Un programme qui joue au jeu puissance 4++.

Fonctions

- def cpb_init (tk.Tk tk_win_root)
  Initialise la page de choix du bonus.
- list[str] cpb_get_bonuses ()
  Récupère les bonus disponibles.
- def cpb_valider_bonus ()
  Récupère les bonus sélectionnés par le joueur.
- def cpb_show_bonus_description (str s_bonus)
  Affiche la description du bonus sélectionné.
- str cpb_get_chosen_bonus ()
  Accesseur du bonus choisi par l'utilisateur.

5.5.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Un module qui gère la page de bonus.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

- tkinter
- numpy
- inspect
  Ce module contient les fonctions permettant de gérer la page de choix du bonus.

5.5.2 Documentation des fonctions

5.5.2.1 cpb_get_bonuses()

```
list[str] src.controller.ctrl_PageBonus.cpb_get_bonuses ( )
```

Récupère les bonus disponibles.
Cette fonction récupère les bonus disponibles et les formate pour l'affichage.

Renvoie

: Liste des bonus disponibles

5.5.2.2 cpb_get_chosen_bonus()

```
str src.controller.ctrl_PageBonus.cpb_get_chosen_bonus ( )
```

Accesseur du bonus choisi par l'utilisateur.
Cette fonction retourne le bonus choisi par l'utilisateur.

Précondition

S_BONUS non nul

Renvoie

: Bonus choisi par l'utilisateur

Variables :

— S_BONUS : nom du bonus sélectionné

5.5.2.3 cpb_init()

```
def src.controller.ctrl_PageBonus.cpb_init (
            tk.Tk tk_win_root )
```

Initialise la page de choix du bonus.
Cette fonction initialise la page de choix du bonus en affichant la fenêtre principale et en initialisant la page de choix du bonus.

Précondition

tk_root initialisé

Paramètres

tk_win_root	Fenêtre principale

Postcondition

page de choix du bonus initialisée

Variables :

— NPA_GRID : Grille de jeu
— TK_ROOT : Fenêtre principale

5.5.2.4 cpb_show_bonus_description()

```
def src.controller.ctrl_PageBonus.cpb_show_bonus_description (
            str s_bonus )
```

Affiche la description du bonus sélectionné.
Cette fonction affiche la description du bonus sélectionné. Pour cela, elle récupère la description du bonus et l'affiche.

Précondition

s_bonus non vide

Paramètres

s_bonus	Nom du bonus

Postcondition

Description du bonus affichée

Variables :

— s_desc : Description du bonus

5.5.2.5 cpb_valider_bonus()

```
def src.controller.ctrl_PageBonus.cpb_valider_bonus ( )
```

Récupère les bonus sélectionnés par le joueur.
Cette fonction récupère les bonus sélectionnés par le joueur et passe à la fenêtre de jeu.

Précondition

fenêtre de choix du bonus affichée

Postcondition

Choix du bonus enregistré

Variables :

- S_BONUS : Nom du bonus sélectionné
- TK_ROOT : Fenêtre principale

5.6 Référence de l'espace de nommage src.controller.ctrl_pageJeu

Un programme qui joue au jeu puissance 4++.

Fonctions

- def cpj_init (tk.Tk tk_win_root)
  Affiche la page de jeu.
- def cpj_draw_grid (int i_nb_rows, int i_nb_columns)
  Dessine la grille de jeu.
- def cpj_put_coin (int i_row, int i_cols, int i_joueur)
  Place un jeton dans la grille de jeu.
- def cpj_undo ()
  Annule le dernier coup.
- def cpj_redo ()
  Refait le dernier coup.
- def cpj_quit ()
  Quitte la partie.
- def cpj_play (tk.Event event, tk.Frame tkf_page_jeu)
  Joue un coup dans la grille de jeu.
- def cpj_use_bonus (tk.Frame tkf_page_jeu)
  Utilise un bonus puis met à jour la grille de jeu.
- def cpj_bot_play (tk.Frame tkf_page_jeu)
  Fait jouer le bot.
- def cpj_update_grid ()
  Réinitialise la grille de jeu.
- def cpj_info_turn (bool b_is_player)
  Affiche le joueur qui doit jouer.

5.6.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Un module qui gère la page de jeu.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

- tkinter
- numpy
- inspect
  Ce module contient les fonctions permettant de gérer la page de jeu.

5.6.2 Documentation des fonctions

5.6.2.1 cpj_bot_play()

```
def src.controller.ctrl_pageJeu.cpj_bot_play (
            tk.Frame tkf_page_jeu )
```

Fait jouer le bot.
Cette fonction fait jouer le bot puis met à jour la grille de jeu.

Précondition

tkf_page_jeu initialisé

Paramètres

tkf_page_jeu	: Frame de la page de jeu

Variables :

- I_NB_JETONS : Nombre de jetons à aligner pour gagner
- I_DIFFICULTY : Difficulté du bot
- i_grid_x : Colonne de la grille de jeu
- i_grid_y : Ligne de la grille de jeu

5.6.2.2 cpj_draw_grid()

```
def src.controller.ctrl_pageJeu.cpj_draw_grid (
            int i_nb_rows,
            int i_nb_columns )
```

Dessine la grille de jeu.
Cette fonction appelle la fonction de la vue permettant de dessiner la grille de jeu.

Précondition

i_nb_rows $>$ 0
i_nb_columns $>$ 0

Paramètres

i_nb_rows	Nombre de lignes de la grille de jeu
i_nb_columns	Nombre de colonnes de la grille de jeu

Postcondition

grille de jeu dessinée

5.6.2.3 cpj_info_turn()

```
def src.controller.ctrl_pageJeu.cpj_info_turn (
            bool b_is_player )
```

Affiche le joueur qui doit jouer.
Cette fonction affiche le joueur qui doit jouer.

Paramètres

b_is_player	: Booléen indiquant si c'est au joueur de jouer

Postcondition

Affichage du joueur qui doit jouer

Variables :

— b_is_player : Booléen indiquant si c'est au joueur de jouer

5.6.2.4 cpj_init()

```
def src.controller.ctrl_pageJeu.cpj_init (
            tk.Tk tk_win_root )
```

Affiche la page de jeu.
Cette fonction affiche la page de jeu et initialise la grille de jeu. Elle récupère également les paramètres de la partie. Elle initialise également les variables globales.

Précondition

tk_root initialisé

Postcondition

page de jeu affichée

Variables :

— TK_ROOT : Fenêtre principale
— ST_COLOR_JOUEUR : Couleur des jetons du joueur
— ST_COLOR_BOT : Couleur des jetons du bot
— I_NB_ROWS : Nombre de lignes de la grille de jeu
— I_NB_COLS : Nombre de colonnes de la grille de jeu
— NPA_GRID : Grille de jeu
— T_UNDO_REDO : Liste des coups joués
— T_REDO : Liste des coups annulés
— I_DIFFICULTY : Difficulté du bot
— I_NB_JETONS : Nombre de jetons à aligner pour gagner
— st_color_grid : Couleur de la grille de jeu

5.6.2.5 cpj_play()

```
def src.controller.ctrl_pageJeu.cpj_play (
            tk.Event event,
            tk.Frame tkf_page_jeu )
```

Joue un coup dans la grille de jeu.
Cette fonction joue un coup dans la grille de jeu et met à jour la grille de jeu.

Précondition

event est un évènement de la souris

tkf_page_jeu initialisé

Paramètres

event	Évènement de la souris sur la grille de jeu
tkf_page_jeu	: Frame de la page de jeu

Variables :

— I_NB_JETONS : Nombre de jetons à aligner pour gagner
— i_grid_x : Colonne de la grille de jeu
— i_grid_y : Ligne de la grille de jeu
— b_joueur_gagne : Booléen indiquant si le joueur a gagné
— b_joueur_joue : Booléen indiquant si le joueur a joué

5.6.2.6 cpj_put_coin()

```
def src.controller.ctrl_pageJeu.cpj_put_coin (
            int i_row,
            int i_cols,
            int i_joueur )
```

Place un jeton dans la grille de jeu.
Cette fonction appelle la fonction de la vue permettant de placer un jeton dans la grille de jeu.

Précondition

i_row >= 0
i_cols >= 0
i_joueur >= 1 et i_joueur <= 2

Paramètres

i_row	Ligne de la grille de jeu
i_cols	Colonne de la grille de jeu
i_joueur	Joueur qui joue

Postcondition

jeton placé dans la grille de jeu

Variables :
- ST_COLOR_JOUEUR : Couleur des jetons du joueur
- ST_COLOR_BOT : Couleur des jetons du bot

5.6.2.7 cpj_quit()

```
def src.controller.ctrl_pageJeu.cpj_quit ( )
```

Quitte la partie.
Cette fonction détruit la page de jeu et ferme la fenêtre principale.

Précondition

tk_root initialisé

Postcondition

page de jeu détruite

5.6.2.8 cpj_redo()

```
def src.controller.ctrl_pageJeu.cpj_redo ( )
```

Refait le dernier coup.
Cette fonction refait le dernier coup annulé et met à jour la grille de jeu.

Précondition

tk_root initialisé

Postcondition

dernier coup refait

5.6.2.9 cpj_undo()

```
def src.controller.ctrl_pageJeu.cpj_undo ( )
```

Annule le dernier coup.
Cette fonction annule le dernier coup joué et met à jour la grille de jeu.

Précondition

tk_root initialisé

Postcondition

dernier coup annulé

5.6.2.10 cpj_update_grid()

```
def src.controller.ctrl_pageJeu.cpj_update_grid ( )
```

Réinitialise la grille de jeu.
Cette fonction réinitialise la grille de jeu puis met à jour la grille de jeu. Elle permet de mettre à jour la grille de jeu après un undo ou un redo.

Précondition

NPA_GRID initialisé

Postcondition

Grille de jeu mise à jour

Variables :

- I_NB_ROWS : Nombre de lignes de la grille de jeu
- I_NB_COLS : Nombre de colonnes de la grille de jeu
- NPA_GRID : Grille de jeu
- i_boucle_row : Ligne de la grille de jeu
- i_boucle_col : Colonne de la grille de jeu

5.6.2.11 cpj_use_bonus()

```
def src.controller.ctrl_pageJeu.cpj_use_bonus (
            tk.Frame tkf_page_jeu )
```

Utilise un bonus puis met à jour la grille de jeu.
Cette fonction utilise un bonus puis met à jour la grille de jeu.

Précondition

S_BONUS est un bonus
NPA_GRID est une grille de jeu

Paramètres

tkf_page_jeu	: Frame de la page de jeu

Postcondition

Bonus utilisé si B_BONUS_USED est faux

Variables :

- S_BONUS : Bonus sélectionné
- NPA_GRID : Grille de jeu
- B_BONUS_USED : Booléen indiquant si le bonus a été utilisé
- m_module : Module du bonus
- f_bonus : Fonction du bonus

A faire Vérifier si le bonus a entraîné une victoire

5.7 Référence de l'espace de nommage src.controller.ctrl_pageParametres

Un programme qui joue au jeu puissance 4++.

Fonctions

- def cpp_init (tk.Tk tk_win_root)
  Initialise la page des paramètres.
- def cpp_settings_save ()
  Sauvegarde les paramètres.
- def cpp_settings_reset ()
  Réinitialise les paramètres.
- def cpp_custom_save ()
  Sauvegarde des paramètres de personnalisation.
- (int, int, int, int) cpp_settings_load ()
  Charge les paramètres.
- (str, str, str) cpp_custom_load ()
  Charge les paramètres de personnalisation.
- def cpp_askcolor (str s_element)
  Ouvre un sélecteur de couleur.
- def cpp_custom_reset ()
  Réinitialise les paramètres de personnalisation.

5.7.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Un module qui gère la page des paramètres.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

- tkinter
- numpy
- inspect
  Contient les fonctions permettant de gérer la page des paramètres

5.7.2 Documentation des fonctions

5.7.2.1 cpp_askcolor()

```
def src.controller.ctrl_pageParametres.cpp_askcolor (
            str s_element )
```

Ouvre un sélecteur de couleur.
Cette fonction ouvre un sélecteur de couleur pour l'élément passé en paramètre. L'élément peut être le joueur, le bot ou la grille.

Précondition

tk_root initialisé
view_pp initialisé

Paramètres

s_element	Élément à colorer (joueur, bot ou grille)

5.7.2.2 cpp_custom_load()

```
(str, str, str) src.controller.ctrl_pageParametres.cpp_custom_load ( )
```

Charge les paramètres de personnalisation.
Cette fonction charge les paramètres de personnalisation sauvegardés dans un fichier texte. Elle récupère la couleur des jetons du joueur, la couleur des jetons du bot et la couleur de la grille. Si le fichier n'existe pas, les paramètres par défaut sont retournés.

Précondition

res/custom.txt existant

Renvoie

st_color_joueur: Couleur des jetons du joueur
st_color_bot: Couleur des jetons du bot
st_color_grid: Couleur de la grille

Variables :

- st_color_joueur : Couleur des jetons du joueur
- st_color_bot : Couleur des jetons du bot
- st_color_grid : Couleur de la grille

5.7.2.3 cpp_custom_reset()

```
def src.controller.ctrl_pageParametres.cpp_custom_reset ( )
```

Réinitialise les paramètres de personnalisation.
Cette fonction réinitialise les paramètres de personnalisation par défaut.

Précondition

tk_root initialisé
view_pp initialisé

5.7.2.4 cpp_custom_save()

```
def src.controller.ctrl_pageParametres.cpp_custom_save ( )
```

Sauvegarde des paramètres de personnalisation.
Cette fonction sauvegarde les paramètres de personnalisation dans un fichier texte. Si les paramètres ne sont pas valides, un message d'erreur est affiché. Les paramètres sont valides si deux couleurs ne sont pas trop proches. Deux couleurs sont trop proches si la différence entre les composantes rouges, vertes et bleues est inférieure à 50.

Précondition

tk_root initialisé
res/custom.txt existant

Postcondition

paramètres de personnalisation sauvegardés ou str_message d'erreur affiché

Variables :

- st_color_joueur : Couleur des jetons du joueur
- st_color_bot : Couleur des jetons du bot
- st_color_grid : Couleur de la grille
- str_message : Message d'information
- f_custom : Fichier de sauvegarde des paramètres de personnalisation

5.7.2.5 cpp_init()

```
def src.controller.ctrl_pageParametres.cpp_init (
            tk.Tk tk_win_root )
```

Initialise la page des paramètres.
Cette fonction initialise la page des paramètres en chargeant les paramètres sauvegardés et en les affichant.

Précondition

tk_root initialisé

Paramètres

tk_win_root	Fenêtre principale

Postcondition

page des paramètres initialisée

Variables :

- tk_root : Fenêtre principale
- i_rows : Nombre de lignes de la grille
- i_columns : Nombre de colonnes de la grille
- i_nb_jetons : Nombre de jetons à aligner pour gagner
- i_difficulty : Difficulté du bot
- st_color_joueur : Couleur des jetons du joueur
- st_color_bot : Couleur des jetons du bot
- st_color_grid : Couleur de la grille

5.7.2.6 cpp_settings_load()

```
(int, int, int, int) src.controller.ctrl_pageParametres.cpp_settings_load ( )
```

Charge les paramètres.
Cette fonction charge les paramètres sauvegardés dans un fichier texte. Elle récupère le nombre de lignes, le nombre de colonnes, le nombre de jetons à aligner et la difficulté du bot. Si le fichier n'existe pas, les paramètres par défaut sont retournés.

Précondition

res/settings.txt existant

Renvoie

i_rows: Nombre de lignes de la grille

i_columns: Nombre de colonnes de la grille

i_nb_jetons: Nombre de jetons à aligner pour gagner

i_difficulty: Difficulté du bot

Variables :

- i_rows : Nombre de lignes de la grille
- i_columns : Nombre de colonnes de la grille
- i_nb_jetons : Nombre de jetons à aligner pour gagner
- i_difficulty : Difficulté du bot

5.7.2.7 cpp_settings_reset()

```
def src.controller.ctrl_pageParametres.cpp_settings_reset ( )
```

Réinitialise les paramètres.
Cette fonction réinitialise les paramètres par défaut.

Précondition

tk_root initialisé
view_pp initialisé

5.7.2.8 cpp_settings_save()

```
def src.controller.ctrl_pageParametres.cpp_settings_save ( )
```

Sauvegarde les paramètres.
Cette fonction sauvegarde les paramètres dans un fichier texte. Si les paramètres ne sont pas valides, un message d'erreur est affiché. Les paramètres sont valides si le nombre de jetons à aligner est inférieur ou égal au nombre de lignes ou au nombre de colonnes.

Précondition

tk_root initialisé
res/settings.txt existant

Postcondition

paramètres sauvegardés ou str_message d'erreur affiché

Variables :

- i_rows : Nombre de lignes de la grille
- i_columns : Nombre de colonnes de la grille
- i_nb_jetons : Nombre de jetons à aligner pour gagner
- i_difficulty : Difficulté du bot
- str_message : Message d'information
- f_settings : Fichier de sauvegarde des paramètres

5.8 Référence de l'espace de nommage src.puissanceQuatre

Espaces de nommage

- bonus
  Un programme qui joue au jeu puissance 4++.
- gestionPartie
  Un programme qui joue au jeu puissance 4++.
- grid
  Un programme qui joue au jeu puissance 4++.
- puissanceQuatre
  Ce module contient l'implémentation des règles du puissance 4.

5.9 Référence de l'espace de nommage src.puissanceQuatre.bonus

Un programme qui joue au jeu puissance 4++.

Fonctions

- np.array p4b_no_bonus (np.array npa_grid)
  Bonus permettant de ne pas jouer de bonus.
- np.array p4b_invert_grid (np.array npa_grid)
  Echange les pions des joueurs.
- np.array p4b_remove_full_line (np.array npa_grid)
  Supprime une ligne pleine.
- np.array p4b_use_min_max (np.array npa_grid)
  Bonus permettant au joueur d'utiliser l'algorithme min max pour son prochain coup.
- np.array p4b_flip_grid (np.array npa_grid)
  Bonus permettant de retourner la grille.
- np.array p4b_random_placement (np.array npa_grid)
  Bonus permettant de placer un jeton aléatoirement.
- np.array p4b_random_bonus (np.array npa_grid)
  Bonus permettant de choisir un bonus aléatoirement.

5.9.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Ce module contient les fonctions relatives aux bonus.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

- tkinter
- numpy
- inspect
  Ce module contient les fonctions de bonus.

5.9.2 Documentation des fonctions

5.9.2.1 p4b_flip_grid()

```
np.array src.puissanceQuatre.bonus.p4b_flip_grid (
            np.array npa_grid )
```

Bonus permettant de retourner la grille.

Précondition

npa_grid initialisé

Paramètres

npa_grid	Grille de jeu

Renvoie

npa_grid : Grille retournée

5.9.2.2 p4b_invert_grid()

```
np.array src.puissanceQuatre.bonus.p4b_invert_grid (
            np.array npa_grid )
```

Echange les pions des joueurs.

Précondition

npa_grid initialisé

Paramètres

npa_grid	Grille

Renvoie

npa_grid: Grille inversée

Variables :

- i_nb_rows : Nombre de lignes de la grille
- i_nb_cols : Nombre de colonnes de la grille
- i_row : Indice de ligne
- i_col : Indice de colonne

5.9.2.3 p4b_no_bonus()

```
np.array src.puissanceQuatre.bonus.p4b_no_bonus (
            np.array npa_grid )
```

Bonus permettant de ne pas jouer de bonus.

Précondition

npa_grid initialisé

Paramètres

npa_grid	Grille de jeu

Renvoie

npa_grid : Grille retournée

5.9.2.4 p4b_random_bonus()

```
np.array src.puissanceQuatre.bonus.p4b_random_bonus (
            np.array npa_grid )
```

Bonus permettant de choisir un bonus aléatoirement.

Précondition

npa_grid initialisé

Paramètres

npa_grid	Grille de jeu

Renvoie

npa_grid : Grille retournée

5.9.2.5 p4b_random_placement()

```
np.array src.puissanceQuatre.bonus.p4b_random_placement (
            np.array npa_grid )
```

Bonus permettant de placer un jeton aléatoirement.

Précondition

npa_grid initialisé

Paramètres

npa_grid	Grille de jeu

Renvoie

npa_grid : Grille retournée

5.9.2.6 p4b_remove_full_line()

```
np.array src.puissanceQuatre.bonus.p4b_remove_full_line (
            np.array npa_grid )
```

Supprime une ligne pleine.

Précondition

npa_grid initialisé

Paramètres

npa_grid	Grille

Renvoie

npa_grid: Grille avec une ligne pleine en moins

Variables :

- i_nb_rows : Nombre de lignes de la grille
- i_nb_cols : Nombre de colonnes de la grille
- i_row : Indice de ligne
- i_col : Indice de colonne
- b_full : Booléen indiquant si la ligne est pleine
- i_row2 : Indice de ligne

5.9.2.7 p4b_use_min_max()

```
np.array src.puissanceQuatre.bonus.p4b_use_min_max (
            np.array npa_grid )
```

Bonus permettant au joueur d'utiliser l'algorithme min max pour son prochain coup.

Précondition

npa_grid initialisé

Paramètres

npa_grid	Grille de jeu

Renvoie

npa_grid : Grille avec un coup de plus de joué

Variables :

- i_col : La colonne qui va être jouée avec l'algorithme min max

5.10 Référence de l'espace de nommage src.puissanceQuatre.gestionPartie

Un programme qui joue au jeu puissance 4++.

Fonctions

- def gp_choose_bonus ()
  Récupère le bonus choisi par le joueur.
- def gp_show_rules ()
  Affiche les règles du jeu.
- str gp_get_player_choice (int i_nb_colonnes, np.array npa_grille)
  Récupère le choix du joueur lors de son tour.
- np.array gp_handle_undo_redo (bool b_undo, np.array npa_grille)
  Méthode permettant au joueur d'annuler ou de refaire son dernier coup.
- def gp_use_bonus (str s_bonus, np.array npa_grille)
  Méthode permettant au joueur d'utiliser son bonus.

- def gp_handle_player_turn (np.array npa_grille, str s_bonus)
  Méthode permettant de gérer le tour du joueur.
- def gp_handle_bot_turn (np.array npa_grille, str s_bonus, int i_nb_jeton_victoire)
  Méthode permettant de gérer le tour du bot.
- def gp_handle_victory (np.array npa_grille, int i_ligne_joueur, int i_joueur, int i_colonne_joueur, int i_nb_↩ jeton_victoire)
  Méthode permettant de gérer la victoire d'un joueur.
- def gp_gestion_partie (int i_nb_lignes=6, int i_nb_colonnes=7, int i_nb_jeton_victoire=4)
  Gère le déroulement d'une partie de puissance 4.
- def gp_start_game ()
  Lance une partie normale en ligne de commande.

5.10.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Un module qui gère la partie de puissance 4 en ligne de commande.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

- tkinter
- numpy
- inspect
  Ce module contient les fonctions permettant de gérer la partie en ligne de commande.

5.10.2 Documentation des fonctions

5.10.2.1 gp_choose_bonus()

```
def src.puissanceQuatre.gestionPartie.gp_choose_bonus ( )
```

Récupère le bonus choisi par le joueur.
Cette fonction permet de récupérer le bonus que le joueur a choisi avant le début de la partie. Elle renvoie le bonus choisi.

Renvoie

Le bonus choisi par le joueur

Variables :

- ts_bonuses : Liste, contient les fonctions des bonus
- i_bonus : Entier, le bonus choisi par le joueur

5.10.2.2 gp_gestion_partie()

```
def src.puissanceQuatre.gestionPartie.gp_gestion_partie (
            int  i_nb_lignes = 6,
            int  i_nb_colonnes = 7,
            int  i_nb_jeton_victoire = 4 )
```

Gère le déroulement d'une partie de puissance 4.
Cette fonction gère l'entièreté du déroulement d'une partie de puissance 4. Elle prend en paramètre la taille de la grille en lignes, la taille de la grille en colonnes et le nombre de jetons à aligner pour gagner.
Méthode gérant le déroulement d'une partie de puissance 4 en ligne de commande.
Variables :

- b_victoire : Booléen, True si un joueur a gagné, False sinon
- b_bonus_utilise : Booléen, True si le joueur a utilisé son bonus, False sinon
- t_undo_redo : Liste, contient les grilles pour l'undo et le redo
- npa_grille : np.array, la grille de jeu
- i_colonne_joueur : Entier, la colonne où le joueur veut jouer
- i_ligne_joueur : Entier, la ligne où le joueur veut jouer

Paramètres

i_nb_lignes	Taille de la grille en lignes
i_nb_colonnes	Taille de la grille en colonnes
i_nb_jeton_victoire	Nombre de jetons à aligner pour gagner

5.10.2.3 gp_get_player_choice()

```
str src.puissanceQuatre.gestionPartie.gp_get_player_choice (
            int i_nb_colonnes,
            np.array npa_grille )
```

Récupère le choix du joueur lors de son tour.
Cette fonction permet de récupérer le choix du joueur lors de son tour. Elle renvoie le choix du joueur.

Paramètres

i_nb_colonnes	Nombre de colonnes de la grille
npa_grille	Grille de jeu

Précondition

i_nb_colonnes $>$ 0
npa_grille initialisé

Renvoie

Le choix du joueur

Postcondition

Le choix du joueur récupéré

Variables :

- i_colonne_joueur : Entier, la colonne où le joueur veut jouer
- s_colonne_joueur : Chaine de caractères, le choix du joueur

5.10.2.4 gp_handle_bot_turn()

```
def src.puissanceQuatre.gestionPartie.gp_handle_bot_turn (
            np.array npa_grille,
            str s_bonus,
            int i_nb_jeton_victoire )
```

Méthode permettant de gérer le tour du bot.
Cette fonction permet de gérer le tour du bot. Elle renvoie la grille modifiée.

Paramètres

npa_grille	np.array, la grille de jeu
s_bonus	str, le nom du bonus
i_nb_jeton_victoire	int, le nombre de jetons à aligner pour gagner

Variables :

- npa_grille : np.array, la grille de jeu
- s_bonus : str, le nom du bonus choisi par le joueur
- i_colonne_joueur : Entier, la colonne où le joueur veut jouer
- i_ligne_joueur : Entier, la ligne où le joueur veut jouer

5.10.2.5 gp_handle_player_turn()

```
def src.puissanceQuatre.gestionPartie.gp_handle_player_turn (
            np.array npa_grille,
            str s_bonus )
```

Méthode permettant de gérer le tour du joueur.
Cette fonction permet de gérer le tour du joueur. Elle renvoie la grille modifiée.

Paramètres

npa_grille	np.array, la grille de jeu
s_bonus	str, le nom du bonus

Renvoie

: np.array, la grille de jeu modifiée

Variables :

- npa_grille : np.array, la grille de jeu
- s_bonus : str, le nom du bonus choisi par le joueur

5.10.2.6 gp_handle_undo_redo()

```
np.array src.puissanceQuatre.gestionPartie.gp_handle_undo_redo (
            bool b_undo,
            np.array npa_grille )
```

Méthode permettant au joueur d'annuler ou de refaire son dernier coup.
Cette fonction permet au joueur d'annuler ou de refaire son dernier coup.

Paramètres

b_undo	booléen indiquant si c'est un undo ou un redo (True pour undo, False pour redo)
npa_grille	np.array, la grille de jeu

Renvoie

: np.array, la grille de jeu modifiée

5.10.2.7 gp_handle_victory()

```
def src.puissanceQuatre.gestionPartie.gp_handle_victory (
            np.array npa_grille,
            int i_ligne_joueur,
            int i_joueur,
            int i_colonne_joueur,
            int i_nb_jeton_victoire )
```

Méthode permettant de gérer la victoire d'un joueur.
Cette fonction permet de gérer la victoire d'un joueur. Elle renvoie un booléen indiquant si un joueur a gagné ou non.

Paramètres

npa_grille	np.array, la grille de jeu
i_ligne_joueur	Entier, la ligne où le joueur veut jouer
i_joueur	Entier, le numéro du joueur

Paramètres

i_colonne_joueur	Entier, la colonne où le joueur veut jouer
i_nb_jeton_victoire	Entier, le nombre de jetons à aligner pour gagner

Renvoie

: Booléen, True si un joueur a gagné, False sinon

Variables :

— b_victoire : Booléen, True si un joueur a gagné, False sinon

5.10.2.8 gp_show_rules()

```
def src.puissanceQuatre.gestionPartie.gp_show_rules ( )
```

Affiche les règles du jeu.
Cette fonction permet d'afficher les règles dans la console python.

5.10.2.9 gp_start_game()

```
def src.puissanceQuatre.gestionPartie.gp_start_game ( )
```

Lance une partie normale en ligne de commande.
Cette fonction démarre une partie de puissance 4 dans la console python.
Cette fonction lance une partie normale en ligne de commande avec une grille de 6 lignes, 7 colonnes et 4 jetons à aligner pour gagner.

5.10.2.10 gp_use_bonus()

```
def src.puissanceQuatre.gestionPartie.gp_use_bonus (
            str s_bonus,
            np.array npa_grille )
```

Méthode permettant au joueur d'utiliser son bonus.
Cette fonction permet au joueur d'utiliser son bonus. Elle renvoie la grille modifiée.

Paramètres

s_bonus	str, le nom du bonus
npa_grille	np.array, la grille de jeu

Renvoie

: np.array, la grille de jeu modifiée

5.11 Référence de l'espace de nommage src.puissanceQuatre.grid

Un programme qui joue au jeu puissance 4++.

Fonctions

— np.array pq_init_grille (int i_max_ligne, int i_max_colonne)
L'initiateur de la grille.
— np.array pq_reset_grille (np.array npa_grille)
Réinitialise la grille.
— def pq_print_grille (np.array npa_grille)
Affiche la grille.
— np.array pq_apply_gravity (np.array npa_grille)
Applique la gravité sur la grille.

5.11.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Ce module contient les fonctions relatives à la grille de jeu.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

- tkinter
- numpy
- inspect

  Ce module contient les fonctions de gestion de la grille de jeu. Notamment l'initialisation de la grille, l'affichage de la grille et la réinitialisation de la grille.

5.11.2 Documentation des fonctions

5.11.2.1 pq_apply_gravity()

```
np.array src.puissanceQuatre.grid.pq_apply_gravity (
            np.array npa_grille )
```

Applique la gravité sur la grille.
Cette méthode permet d'appliquer la gravité sur la grille de jeu. Elle prend en paramètre la grille à modifier et renvoie la grille modifiée.

Précondition

npa_grille initialisé

Paramètres

npa_grille	La grille à modifier

Postcondition

npa_grille contient des 0 et des 1 ou 2

il n'y a pas de 0 sous un 1 ou un 2

Variables :

- i_nb_ligne : Nombre de lignes de la grille
- i_nb_colonne : Nombre de colonnes de la grille
- i_boucle_colonne : Compteur de boucle pour les colonnes de la grille
- i_boucle_ligne : Compteur de boucle pour les lignes de la grille
- i_compt : Compteur de sécurité

5.11.2.2 pq_init_grille()

```
np.array src.puissanceQuatre.grid.pq_init_grille (
            int i_max_ligne,
            int i_max_colonne )
```

L'initiateur de la grille.
Cette méthode permet d'initialiser la grille de jeu. Elle prend en paramètre le nombre de lignes et de colonnes de la grille et renvoie la grille initialisée.

Précondition

i_max_ligne $>$ 1 et i_max_colonne $>$ 1

Paramètres

i_max_ligne	Le nombre de lignes de la grille
i_max_colonne	Le nombre de colonnes de la grille

Postcondition

npa_grille initialisé

Renvoie

La grille créée

Variables :

— npa_grille : np.array

5.11.2.3 pq_print_grille()

```
def src.puissanceQuatre.grid.pq_print_grille (
            np.array npa_grille )
```

Affiche la grille.

Cette méthode permet d'afficher la grille de jeu. Elle prend en paramètre la grille à afficher.

Variables :

— char_joueur : Le caractère du jeton du joueur
— char_bot : Le caractère du jeton du bot
— char_vide : Le caractère représentant une case vide
— i_max_ligne : Le nombre de lignes de la grille
— i_max_colonne : Le nombre de colonnes de la grille
— i_boucle_colonne : Le compteur de boucle pour les colonnes de la grille
— i_boucle_ligne : Le compteur de boucle pour les lignes de la grille

Paramètres

npa_grille	La grille à afficher

5.11.2.4 pq_reset_grille()

```
np.array src.puissanceQuatre.grid.pq_reset_grille (
            np.array npa_grille )
```

Réinitialise la grille.

Cette méthode permet de réinitialiser la grille de jeu. Elle prend en paramètre la grille à réinitialiser et renvoie la grille réinitialisée.

Variables :

— i_boucle : Entier
— i_max_ligne : Entier
— i_max_colonne : Entier

Précondition

npa_grille initialisé

Paramètres

npa_grille	La grille à réinitialiser

Postcondition

npa_grille contient des 0

Renvoie

La grille réinitialisée

5.12 Référence de l'espace de nommage src.puissanceQuatre.puissanceQuatre

Ce module contient l'implémentation des règles du puissance 4.

Fonctions

- bool pq_verif_colonne (int i_colonne, np.array npa_grille)
  Vérifie si on peut poser un jeton dans cette colonne.
- (int, int) pq_ajout_piece (np.array npa_grille, int i_colonne, int i_joueur)
  La méthode qui gère le placement de jetons.
- float pq_minmax (i_joueur, npa_grille_copy, i_nb_victoire, s_bonus, b_bonus_used, i_colonne=0, b_is_↩ first=False, i_tour=0, b_is_the_bonus=False)
  Méthode implémentant l'algorithme minmax.
- bool pq_victoire (np.array npa_grille, int i_ligne, int i_colonne, int i_joueur, int i_nb_victoire)
- bool pq_victoire_ligne (np.array npa_grille, int i_ligne, int i_colonne, int i_joueur, int i_nb_victoire)
  Verification de la victoire sur la ligne.
- bool pq_victoire_colonne (np.array npa_grille, int i_ligne, int i_colonne, int i_joueur, int i_nb_victoire)
  Verification de la victoire sur une colonne.
- bool pq_victoire_diago (np.array npa_grille, int i_ligne, int i_colonne, int i_joueur, int i_nb_victoire)
  Verification de la victoire sur les diagonales.
- np.array pq_undo (np.array npa_grille, list t_undo_redo)
  Méthode permettant de revenir en arrière dans le jeu.
- np.array pq_redo (np.array npa_grille, list t_redo)
  Méthode permettant de revenir en avant dans le jeu.
- bool pq_partie_finie (np.array npa_grille, bool b_bonus_utilise)
  Vérification de si la partie est finie ou non.

5.12.1 Description détaillée

Ce module contient l'implémentation des règles du puissance 4.
Ce module contient la gestion de la structure du puissance 4 et la gestion du jeu.

5.12.2 Documentation des fonctions

5.12.2.1 pq_ajout_piece()

```
(int, int) src.puissanceQuatre.puissanceQuatre.pq_ajout_piece (
            np.array npa_grille,
            int i_colonne,
            int i_joueur )
```

La méthode qui gère le placement de jetons.
Cette méthode permet d'ajouter une pièce dans la colonne indiquée. Elle vérifie si la colonne est valide et si elle n'est pas pleine. Elle renvoie les coordonnées du nouveau jeton.
Cette méthode permet de placer un jeton dans une colonne donnée. Elle vérifie si la colonne est valide et si elle n'est pas pleine. Elle renvoie les coordonnées du nouveau jeton.

Précondition

0 < i_colonne <= npa_grille.shape[0]
npa_grille initialisé

Paramètres

i_colonne	La colonne où le joueur pose le jeton
i_joueur	Le joueur qui joue (1 pour le joueur, 2 pour le bot)
npa_grille	La grille du puissance 4

Postcondition

npa_grille contient un nouvel entier

Renvoie

Les coordonnées du nouveau jeton

Variables :
- i_boucle **Entier** : Compteur de boucle
- i_max_ligne **Entier** : Nombre de lignes dans la grille
- ti_coords **Tuple d'entiers** : Coordonnées du nouveau jeton

5.12.2.2 pq_minmax()

```
float src.puissanceQuatre.puissanceQuatre.pq_minmax (
              i_joueur,
              npa_grille_copy,
              i_nb_victoire,
              s_bonus,
              b_bonus_used,
              i_colonne = 0,
              b_is_first = False,
              i_tour = 0,
              b_is_the_bonus = False )
```

Méthode implémentant l'algorithme minmax.
Cette méthode permet de jouer un coup en utilisant l'algorithme minmax. Elle prend en paramètre le joueur qui joue, la grille de jeu, le nombre de jetons à aligner pour gagner, le bonus à jouer, un booléen indiquant si le bonus a déjà été joué, la colonne où jouer le bonus, un booléen indiquant si c'est le premier appel de la méthode, le nombre de tours effectués et un booléen indiquant si le bonus est utilisé ou non.
Variables :
- m_module : Module, Le module du bonus
- f_bonus : Fonction, La fonction du bonus
- tf_result : Liste, La liste contenant les résultats
- i_maximum : Entier, Le i_maximum de la liste
- i_max_index : Entier, L'indice du i_maximum de la liste
- ligne : Entier, La ligne où le jeton a été posé

Précondition

npa_grille initialisé

1 <= i_joueur <= 2

0 <= i_colonne <= npa_grille.shape[0]

Paramètres

i_joueur	Le joueur qui joue (1 pour le joueur, 2 pour le bot)
npa_grille_copy	La grille du puissance 4
i_nb_victoire	Le nombre de jetons à aligner pour gagner
s_bonus	Le bonus à jouer
b_bonus_used	Un booléen indiquant si le bonus a déjà été joué
i_colonne	La colonne où jouer le bonus

Paramètres

b_is_first	Un booléen indiquant si c'est le premier appel de la méthode
i_tour	Le nombre de tours effectués
b_is_the_bonus	Un booléen indiquant si le bonus est utilisé ou non

Renvoie

La colonne où jouer le jeton

5.12.2.3 pq_partie_finie()

```
bool src.puissanceQuatre.puissanceQuatre.pq_partie_finie (
            np.array npa_grille,
            bool b_bonus_utilise )
```

Vérification de si la partie est finie ou non.
La vérification se fait avec deux critères : Si la grille est pleine ou non, ainsi que si le joueur peut encore utiliser son bonus.
Variables :

- i_nb_lignes : Le nombre de lignes de la grille
- i_nb_colonnes : Le nombre de colonnes de la grille
- b_tableau_plein : Booléen, True si la grille est pleine, False sinon
- i_boucle_ligne : Entier, Compteur de boucle pour les lignes
- i_boucle_colonne : Entier, Compteur de boucle pour les colonnes

Préconditions :

- npa_grille initialisé
- 2 <= i_nb_lignes
- 2 <= i_nb_colonnes

Paramètres

npa_grille	La grille du puissance 4
b_bonus_utilise	Un booléen permettant de savoir si le joueur a utilisé son bonus ou non.

Renvoie

True si la partie est finie, False sinon

5.12.2.4 pq_redo()

```
np.array src.puissanceQuatre.puissanceQuatre.pq_redo (
            np.array npa_grille,
            list t_redo )
```

Méthode permettant de revenir en avant dans le jeu.
Cette méthode permet de revenir en avant dans le jeu. Elle prend en paramètre la grille du puissance 4 et la liste contenant les grilles pour l'undo et le redo. Elle renvoie la grille du puissance 4 après le redo.

Paramètres

npa_grille	La grille du puissance 4
t_redo	La liste contenant les grilles pour l'undo et le redo

Renvoie

La grille du puissance 4 après le redo

Variables :

— npa : np.array, la grille du puissance 4 au coup annulé

5.12.2.5 pq_undo()

```
np.array src.puissanceQuatre.puissanceQuatre.pq_undo (
            np.array npa_grille,
            list t_undo_redo )
```

Méthode permettant de revenir en arrière dans le jeu.
Cette méthode permet de revenir en arrière dans le jeu. Elle prend en paramètre la grille du puissance 4 et la liste contenant les grilles pour l'undo et le redo. Elle renvoie la grille du puissance 4 après l'undo.

Paramètres

npa_grille	La grille du puissance 4
t_undo_redo	La liste contenant les grilles pour l'undo et le redo

Renvoie

La grille du puissance 4 après l'undo

Variables :

— npa : np.array, la grille du puissance 4 au coup précédent

5.12.2.6 pq_verif_colonne()

```
bool src.puissanceQuatre.puissanceQuatre.pq_verif_colonne (
            int i_colonne,
            np.array npa_grille )
```

Vérifie si on peut poser un jeton dans cette colonne.
Cette fonction permet de vérifier si on peut poser un jeton dans une colonne donnée. Elle vérifie si la colonne est valide et si elle n'est pas pleine. Elle renvoie un booléen indiquant si on peut poser un jeton dans cette colonne ou non.

Précondition

0 < i_colonne <= npa_grille.shape[0]

npa_grille initialisé

Paramètres

i_colonne	La colonne où on souhaite poser un jeton
npa_grille	La grille de jeu

Postcondition

npa_grille[i_colonne] contient au moins un 0

Renvoie

True si on peut poser le jeton, False sinon

Variable :

— b_resultat : Booléen
— i_boucle : Entier

5.12.2.7 pq_victoire()

```
bool src.puissanceQuatre.puissanceQuatre.pq_victoire (
            np.array npa_grille,
            int i_ligne,
            int i_colonne,
            int i_joueur,
            int i_nb_victoire )

: Méthode appelant les trois vérifications de victoire.

Cette méthode permet de vérifier si le joueur a gagné ou non. Elle prend en
paramètre la grille de jeu, la ligne et la colonne où le jeton a été posé,
le joueur qui a joué et le nombre de jetons à aligner pour gagner. Elle
renvoie un booléen indiquant si le joueur a gagné ou non.

@param npa_grille: La grille du puissance 4
@param i_ligne: La ligne où le jeton a été posé
@param i_colonne: La colonne où le jeton a été posé
@param i_joueur: Le joueur qui a joué (1 pour le joueur humain, 2 pour le
                bot)
@param i_nb_victoire: Nombre de jetons à combiner pour gagner
@return True si le joueur i_joueur a gagné, False sinon
```

5.12.2.8 pq_victoire_colonne()

```
bool src.puissanceQuatre.puissanceQuatre.pq_victoire_colonne (
            np.array npa_grille,
            int i_ligne,
            int i_colonne,
            int i_joueur,
            int i_nb_victoire )
```

Verification de la victoire sur une colonne.
Cette méthode permet de vérifier si le joueur a gagné sur la colonne où il a joué. Elle prend en paramètre la grille de jeu, la ligne et la colonne où le jeton a été posé, le joueur qui a joué et le nombre de jetons à aligner pour gagner. Elle renvoie un booléen indiquant si le joueur a gagné ou non.
Variables :

- i_compteur : Entier, Le nombre de jetons du joueur dans la ligne
- i_nb_lignes : Entier, Nombre de lignes dans la grille
- b_victoire : Booléen, Indique si le joueur a gagné ou non

Préconditions :

- npa_grille initialisé
- npa_grille contient un jeton en i_ligne, i_colonne
- 1 <= i_joueur <= 2

Paramètres

npa_grille	La grille du puissance 4
i_ligne	La ligne où le jeton a été posé
i_colonne	La colonne où le jeton a été posé
i_joueur	Le joueur qui a joué
i_nb_victoire	Le nombre de jetons nécessaire pour la victoire

Renvoie

True si le joueur i_joueur a gagné, False sinon

5.12.2.9 pq_victoire_diago()

```
bool src.puissanceQuatre.puissanceQuatre.pq_victoire_diago (
            np.array npa_grille,
            int i_ligne,
            int i_colonne,
            int i_joueur,
            int i_nb_victoire )
```

Verification de la victoire sur les diagonales.
Cette méthode permet de vérifier si le joueur a gagné sur les diagonales où il a joué. Elle prend en paramètre la grille de jeu, la ligne et la colonne où le jeton a été posé, le joueur qui a joué et le nombre de jetons à aligner pour gagner. Elle renvoie un booléen indiquant si le joueur a gagné ou non.
Variables :

- i_compteur : Entier, Le nombre de jetons du joueur dans la ligne
- i_nb_lignes : Entier, Nombre de lignes dans la grille
- i_nb_colonnes : Entier, Nombre de colonnes dans la grille
- tti_directions : Tableau de tuples d'entiers, Les directions à vérifier
- i_dx : Entier, Composante x de la direction
- i_dy : Entier, Composante y de la direction

Préconditions :

- npa_grille initialisé
- npa_grille contient un jeton en i_ligne, i_colonne
- 1 <= i_joueur <= 2

Paramètres

npa_grille	La grille du puissance 4
i_ligne	La ligne où le jeton a été posé
i_colonne	La colonne où le jeton a été posé
i_joueur	Le joueur qui a joué
i_nb_victoire	Le nombre de jetons nécessaire pour la victoire

Renvoie

True si le joueur i_joueur a gagné, False sinon

5.12.2.10 pq_victoire_ligne()

```
bool src.puissanceQuatre.puissanceQuatre.pq_victoire_ligne (
            np.array npa_grille,
            int i_ligne,
            int i_colonne,
            int i_joueur,
            int i_nb_victoire )
```

Verification de la victoire sur la ligne.
Cette méthode permet de vérifier si le joueur a gagné sur la ligne où il a joué. Elle prend en paramètre la grille de jeu, la ligne et la colonne où le jeton a été posé, le joueur qui a joué et le nombre de jetons à aligner pour gagner. Elle renvoie un booléen indiquant si le joueur a gagné ou non.
Variables :

- i_compteur : Entier, Le nombre de jetons du joueur dans la ligne
- b_vu : Booléen, ajouter explication
- b_suite : Booléen, ajouter explication

- i_nb_colonnes : Entier, Nombre de colonnes dans la grille
- i_boucle : Entier, Compteur de boucle
- i_debut : Entier, premier emplacement possible pour la combinaison de victoire dans la ligne
- i_fin : Entier, dernier emplacement possible pour la combinaison de victoire dans la ligne

Préconditions :

- npa_grille initialisé
- npa_grille contient un jeton en i_ligne, i_colonne
- 1 <= i_joueur <= 2

```
@param npa_grille: La grille du jeu
@param i_ligne: La ligne où le jeton a été posé
@param i_colonne: La colonne où le jeton a été posé
@param i_joueur: Le joueur qui a joué
@param i_nb_victoire: Le nombre de jetons nécessaire pour la victoire

@return True si le joueur a gagné, False sinon
```

5.13 Référence de l'espace de nommage src.utils

Espaces de nommage

- bonus_utils
  Un programme qui joue au jeu puissance 4++.
- colors_utils
  Un programme qui joue au jeu puissance 4++.
- widget_utils
  Un programme qui joue au jeu puissance 4++.

5.14 Référence de l'espace de nommage src.utils.bonus_utils

Un programme qui joue au jeu puissance 4++.

Fonctions

- list[tuple[str,...]] bu_get_bonuses ()
  Retourne la liste des noms des fonctions bonus.
- str bu_get_bonus_name (tuple t_function)
  Retourne le nom d'une fonction bonus.
- str bu_format_bonus_name (str s_bonus_name)
  Formate le nom d'un bonus.
- str bu_unformat_bonus_name (str s_bonus_name)
  Déformate le nom d'un bonus.
- str bu_get_bonus_description (str s_bonus_name)
  Retourne la description d'un bonus.

5.14.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Ce module contient les fonctions utiles aux bonus.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

- tkinter
- numpy
- inspect
  Ce module contient les fonctions utilitaires relatives aux bonus.

5.14.2 Documentation des fonctions

5.14.2.1 bu_format_bonus_name()

```
str src.utils.bonus_utils.bu_format_bonus_name (
            str s_bonus_name )
```

Formate le nom d'un bonus.
Cette fonction formate le nom d'un bonus pour l'afficher dans le menu. Elle enlève le préfixe "p4b_" et remplace-les "_" par des espaces.

5.14.2.2 bu_get_bonus_description()

```
str src.utils.bonus_utils.bu_get_bonus_description (
            str s_bonus_name )
```

Retourne la description d'un bonus.
Cette fonction retourne la description d'un bonus. Elle utilise pour cela la documentation de la fonction bonus.

5.14.2.3 bu_get_bonus_name()

```
str src.utils.bonus_utils.bu_get_bonus_name (
            tuple t_function )
```

Retourne le nom d'une fonction bonus.
Cette fonction retourne le nom d'une fonction bonus.

5.14.2.4 bu_get_bonuses()

```
list[tuple[str, ...]] src.utils.bonus_utils.bu_get_bonuses ( )
```

Retourne la liste des noms des fonctions bonus.
Cette fonction retourne la liste des noms des fonctions bonus. Elle utilise le module bonus pour récupérer les fonctions bonus. Elle utilise aussi l'inspecteur pour récupérer le nom des fonctions bonus.

Postcondition

La liste des noms des fonctions bonus est retournée

Variables :

— ts_functions : Liste des fonctions du module bonus

5.14.2.5 bu_unformat_bonus_name()

```
str src.utils.bonus_utils.bu_unformat_bonus_name (
            str s_bonus_name )
```

Déformate le nom d'un bonus.
Cette fonction déformate le nom d'un bonus pour l'utiliser dans le code. Elle ajoute le préfixe "p4b_" et remplace-les espaces par des "_" afin de retrouver le nom de la fonction.

5.15 Référence de l'espace de nommage src.utils.colors_utils

Un programme qui joue au jeu puissance 4++.

Fonctions

— (int, int, int) cu_hex_to_rgb (str s_color)
Convertit une couleur hexadécimale en RGB.
— int cu_rgb_distance ((int, int, int) rgb1,(int, int, int) rgb2)
Calcule la distance entre deux couleurs RGB.
— bool cu_colors_too_close (str color1, str color2)
Vérifie si deux couleurs sont trop proches.

5.15.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Ce module contient les fonctions relatives aux couleurs.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

- tkinter
- numpy
- inspect

  Ce module contient les fonctions de gestion des couleurs. Notamment la conversion d'une couleur hexadécimale en RGB et la vérification de la distance entre deux couleurs.

5.15.2 Documentation des fonctions

5.15.2.1 cu_colors_too_close()

```
bool src.utils.colors_utils.cu_colors_too_close (
            str color1,
            str color2 )
```

Vérifie si deux couleurs sont trop proches.
Cette fonction vérifie si deux couleurs sont trop proches. Elle utilise la fonction cu_rgb_distance pour calculer la distance entre les deux couleurs. Si la distance est inférieure à 50, les couleurs sont trop proches.

Paramètres

color1	Couleur 1 au format hexadécimal
color2	Couleur 2 au format hexadécimal

Renvoie

True si les couleurs sont trop proches, False sinon

5.15.2.2 cu_hex_to_rgb()

```
(int, int, int) src.utils.colors_utils.cu_hex_to_rgb (
            str s_color )
```

Convertit une couleur hexadécimale en RGB.
Cette fonction convertit une couleur hexadécimale en RGB.

Paramètres

s_color	Couleur hexadécimale

Renvoie

Couleur RGB

5.15.2.3 cu_rgb_distance()

```
int src.utils.colors_utils.cu_rgb_distance (
            (int, int, int) rgb1,
            (int, int, int) rgb2 )
```

Calcule la distance entre deux couleurs RGB.
Cette fonction calcule la distance entre deux couleurs RGB. Elle utilise la formule de la distance euclidienne.

Paramètres

rgb1	Couleur 1
rgb2	Couleur 2

Renvoie

Distance entre les deux couleurs

5.16 Référence de l'espace de nommage src.utils.widget_utils

Un programme qui joue au jeu puissance 4++.

Fonctions

- (int, int) wu_get_screen_size (tk.Frame tkf_frame)
  Récupère la taille de l'écran.
- (int, int) wu_get_grid_size (tk.Frame tkf_frame)
  Récupère la taille de la grille.
- int wu_get_font_size (tk.Frame tkf_frame, bool b_title)
  Récupère la taille de la police.
- int wu_get_font_size_window (tk.Tk tk_window, bool b_title)
  Récupère la taille de la police en fonction de la fenêtre.

5.16.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Ce module contient les fonctions relatives aux widgets.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

- tkinter
- numpy
- inspect
  Ce module contient les fonctions de gestion des widgets. Il permet de gérer la taille de la fenêtre en fonction de la résolution de l'écran.

5.16.2 Documentation des fonctions

5.16.2.1 wu_get_font_size()

```
int src.utils.widget_utils.wu_get_font_size (
            tk.Frame tkf_frame,
            bool b_title )
```

Récupère la taille de la police.
Cette fonction calcule la taille de la police et la renvoie.

Précondition

Le cadre doit être initialisé

Paramètres

tkf_frame	Frame tkinter
b_title	True si la police est pour un titre, False sinon

Renvoie

Taille de la police

5.16.2.2 wu_get_font_size_window()

```
int src.utils.widget_utils.wu_get_font_size_window (
            tk.Tk tk_window,
            bool b_title )
```

Récupère la taille de la police en fonction de la fenêtre.
Cette fonction calcule la taille de la police et la renvoie. Elle prend en paramètre la fenêtre principale.

Précondition

Le cadre doit être initialisé

Paramètres

tk_window	Fenêtre principale
b_title	True si la police est pour un titre, False sinon

Renvoie

Taille de la police

5.16.2.3 wu_get_grid_size()

```
(int, int) src.utils.widget_utils.wu_get_grid_size (
            tk.Frame tkf_frame )
```

Récupère la taille de la grille.
Cette fonction calcule la taille de la grille et la renvoie sous la forme d'un tuple (largeur, hauteur).

Paramètres

tkf_frame	Frame tkinte

Précondition

Le cadre doit être initialisé

Renvoie

Tuple (largeur, hauteur)

5.16.2.4 wu_get_screen_size()

```
(int, int) src.utils.widget_utils.wu_get_screen_size (
            tk.Frame tkf_frame )
```

Récupère la taille de l'écran.
Cette fonction récupère la taille de l'écran et la renvoie sous la forme d'un tuple (largeur, hauteur).

Paramètres

tkf_frame	Frame tkinter

Précondition

Le cadre doit être initialisé

Renvoie

Tuple (largeur, hauteur)

5.17 Référence de l'espace de nommage src.view

Espaces de nommage

- view_main
  Un programme qui joue au jeu puissance 4++.
- view_pageAccueil
  Un programme qui joue au jeu puissance 4++.
- view_pageBonus
  Un programme qui joue au jeu puissance 4++.
- view_pageJeu
  Un programme qui joue au jeu puissance 4++.
- view_pageParametres
  Un programme qui joue au jeu puissance 4++.

5.18 Référence de l'espace de nommage src.view.view_main

Un programme qui joue au jeu puissance 4++.

Fonctions

- tk.Tk vm_init ()
  Initialise la fenêtre de jeu.
- def vm_quit (tk.Tk tk_win_root)
  Ferme la fenêtre de jeu.
- tk.Menu vm_menu (tk.Frame tk_old_frame, bool b_in_game)
  Initialise le menu de la fenêtre de jeu.
- def vm_message_game_ended (str s_message, tk.Frame tkf_page_jeu)
  Affiche un message de fin de partie.
- def vm_message_warning (str str_message)
  Affiche un message d'avertissement.
- def vm_message_info (str str_message)
  Affiche un message d'information.
- def vm_remove_frame (tk.Frame frame)
  Supprime un cadre.
- def vm_update (tk.Tk tk_root)
  Met à jour la fenêtre principale.

5.18.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Ce module contient les fonctions de base utile à la vue du jeu.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

- tkinter
- numpy
- inspect
  Ce module permet de facilement avoir le menu sur toutes les fenêtres. De quitter le jeu, de recommencer une partie, etc. Il permet aussi d'afficher des messages d'information ou d'avertissement.

5.18.2 Documentation des fonctions

5.18.2.1 vm_init()

```
tk.Tk src.view.view_main.vm_init ( )
```

Initialise la fenêtre de jeu.

Cette fonction initialise la fenêtre principale du jeu. Elle crée la fenêtre, lui donne un titre, un logo, la rend non redimensionnable et renvoie la fenêtre créée.

Variables :

- tk_root : Fenêtre principale
- tkfo_default_font : Police par défaut

Renvoie

Fenêtre principale

5.18.2.2 vm_menu()

```
tk.Menu src.view.view_main.vm_menu (
            tk.Frame tk_old_frame,
            bool b_in_game )
```

Initialise le menu de la fenêtre de jeu.

Cette fonction initialise le menu de la fenêtre principale du jeu. Elle crée le menu, les sous-menus, les commandes et renvoie le menu créé. Ce menu est affiché dans toutes les fenêtres.

Variables :

- tkm_menu_bar : Menu de la fenêtre de jeu
- tkm_menu_partie : sous-menu permettant de gérer la partie
- tkm_menu_param : sous-menu permettant d'accéder aux paramètres
- tkm_menu_a_propos : sous-menu permettant d'accéder à la page "À propos"

Renvoie

Menu de la fenêtre de jeu

5.18.2.3 vm_message_game_ended()

```
def src.view.view_main.vm_message_game_ended (
            str s_message,
            tk.Frame tkf_page_jeu )
```

Affiche un message de fin de partie.

Cette fonction affiche un message de fin de partie. Elle demande à l'utilisateur s'il veut rejouer ou non. Si oui, elle relance une partie sinon elle revient à la page d'accueil. Elle est appelée lorsque la partie est terminée.

Préconditions :

- tk_root initialisé

Paramètres

s_message	Message à afficher
tkf_page_jeu	La page de jeu

5.18.2.4 vm_message_info()

```
def src.view.view_main.vm_message_info (
            str str_message )
```

Affiche un message d'information.

Cette fonction affiche un message d'information. Elle est appelée lorsque l'utilisateur fait une action qui est autorisée. Par exemple, lorsque l'utilisateur change les paramètres du jeu.

Paramètres

str_message	Message à afficher

5.18.2.5 vm_message_warning()

```
def src.view.view_main.vm_message_warning (
            str str_message )
```

Affiche un message d'avertissement.
Cette fonction affiche un message d'avertissement. Elle est appelée lorsque l'utilisateur fait une action qui n'est pas autorisée. Par exemple, si l'utilisateur choisit des paramètres qui ne sont pas compatibles avec le jeu.

Paramètres

str_message	Message à afficher

5.18.2.6 vm_quit()

```
def src.view.view_main.vm_quit (
            tk.Tk tk_win_root )
```

Ferme la fenêtre de jeu.
Cette fonction ferme la fenêtre principale du jeu.

```
**Préconditions :**
* tk_root initialisé

@param tk_win_root: Fenêtre principale
```

5.18.2.7 vm_remove_frame()

```
def src.view.view_main.vm_remove_frame (
            tk.Frame frame )
```

Supprime un cadre.
Cette fonction supprime un cadre. Elle est appelée lorsque l'on veut changer de page.

Précondition

frame existe

Paramètres

frame	Le cadre à supprimer

Postcondition

frame n'existe plus

5.18.2.8 vm_update()

```
def src.view.view_main.vm_update (
            tk.Tk tk_root )
```

Met à jour la fenêtre principale.
Cette fonction met à jour la fenêtre principale. Elle est appelée lorsque l'on veut mettre à jour la fenêtre principale.

Paramètres

tk_root	Fenêtre principale

5.19 Référence de l'espace de nommage src.view.view_pageAccueil

Un programme qui joue au jeu puissance 4++.

Fonctions

- def vpa_init (tk.Tk tk_root)
  Initialise la page d'accueil.
- def vpa_destroy ()
  Détruit la page d'accueil.

5.19.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Ce module représente la vue de la page d'accueil.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

- tkinter
- numpy
- inspect

  Ce module contient les fonctions d'initialisation et de destruction de la page d'accueil.

5.19.2 Documentation des fonctions

5.19.2.1 vpa_destroy()

```
def src.view.view_pageAccueil.vpa_destroy ( )
```

Détruit la page d'accueil.
Cette fonction détruit la page d'accueil. Elle efface le cadre et le supprime.
Variables :

- tkf_page_accueil : Frame de la page d'accueil

5.19.2.2 vpa_init()

```
def src.view.view_pageAccueil.vpa_init (
            tk.Tk tk_root )
```

Initialise la page d'accueil.
Cette fonction initialise la page d'accueil. Elle crée un cadre, un label contenant le titre et un bouton pour lancer une partie.
Variables :

- tkf_page_accueil : Frame de la page d'accueil
- tkl_title : Label contenant le titre
- tkB_play : Bouton pour lancer une partie

Préconditions :

- tk_root initialisé

Paramètres

tk_root	Fenêtre principale

Voir également

src/controller/ctrl_pageAccueil.py

5.20 Référence de l'espace de nommage src.view.view_pageBonus

Un programme qui joue au jeu puissance 4++.

Fonctions

- None vpb_init (tk.Tk tk_root)
  Initialisation de la fenêtre de choix du bonus.
- tuple[str,...] vpb_get_bonus ()
  Récupère le nom du bonus sélectionné par le joueur.
- def vpb_show_bonus_description (str s_description)
  Affiche la description d'un bonus.
- tk.Frame vpb_get_frame ()
  Accesseur du cadre de la fenêtre de choix du bonus.

5.20.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Ce module représente la vue de la page de choix du bonus.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

- tkinter
- numpy
- inspect

  Ce module contient les fonctions permettant de gérer la vue de la page de choix du bonus.

5.20.2 Documentation des fonctions

5.20.2.1 vpb_get_bonus()

```
tuple[str, ...] src.view.view_pageBonus.vpb_get_bonus ( )
```

Récupère le nom du bonus sélectionné par le joueur.
Cette fonction récupère le nom du bonus sélectionné par le joueur. Elle renvoie le nom du bonus sélectionné.

Précondition

TKS_BONUS initialisé

Renvoie

: Le nom du bonus sélectionné par le joueur

Variables :

- TKS_BONUS : Variable de choix du bonus.

5.20.2.2 vpb_get_frame()

```
tk.Frame src.view.view_pageBonus.vpb_get_frame ( )
```

Accesseur du cadre de la fenêtre de choix du bonus.
Cette fonction renvoie le cadre de la fenêtre de choix du bonus. Elle est utilisée par le contrôleur principal pour afficher la fenêtre de choix du bonus.

Précondition

Cadre initialisé

Renvoie

: Cadre de la fenêtre de choix du bonus.

Variables :

— TKF_PAGE_CHOIX : Cadre de la fenêtre de choix du bonus.

5.20.2.3 vpb_init()

```
None src.view.view_pageBonus.vpb_init (
            tk.Tk tk_root )
```

Initialisation de la fenêtre de choix du bonus.
Cette fonction initialise la fenêtre de choix du bonus. Elle crée un cadre, un titre, un menu déroulant pour le choix du bonus, un label pour la description du bonus et un bouton pour valider le bonus. Elle affiche aussi la description du premier bonus. Elle affiche aussi le menu sur la fenêtre.

Précondition

tk_root initialisé

Paramètres

tk_root	la fenêtre de base

Postcondition

Fenêtre de choix du bonus initialisée

Variables :

— TKS_BONUS : Variable de choix du bonus.
— TKL_DESCRIPTION_BONUS : Label de la description du bonus.
— TKF_PAGE_CHOIX : Cadre de la fenêtre de choix du bonus.
— tkL_titre : Label du titre de la fenêtre de choix du bonus.
— tkC_bonus : Menu déroulant pour le choix du bonus.
— tkL_description : Label de la description du bonus.
— tkB_valider : Bouton de validation du bonus.

5.20.2.4 vpb_show_bonus_description()

```
def src.view.view_pageBonus.vpb_show_bonus_description (
            str s_description )
```

Affiche la description d'un bonus.
Cette fonction affiche la description d'un bonus. Elle prend en paramètre la description du bonus à afficher. Elle affiche la description du bonus dans le label de la description du bonus.

Précondition

TKL_DESCRIPTION_BONUS initialisé

Paramètres

s_description	Description du bonus

Variables :

— TKL_DESCRIPTION_BONUS : Label de la description du bonus.

5.21 Référence de l'espace de nommage src.view.view_pageJeu

Un programme qui joue au jeu puissance 4++.

Fonctions

- def vpj_init_page_jeu (tk.Tk tk_root, str st_color_grid)
  Initialise la page de jeu.
- def vpj_destroy ()
  Détruit la page de jeu.
- def vpj_draw_grid (int rows, int columns)
  Dessine la grille de jeu.
- def vpj_show_coin (int i_row, int i_column, str color)
  Dessine un jeton dans une cellule.
- (int, int) vpj_get_grid_cell (int i_x, int i_y)
  Récupère les coordonnées dans la grille de la cellule cliquée.
- def vpj_disable_bonus ()
  Désactive le bouton bonus.
- tk.Frame vpj_get_frame ()
  Accesseur de la frame de la page de jeu.
- def vpj_set_info (str st_info)
  Modifie le texte du label d'information.

5.21.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Ce module représente la vue de la page de jeu.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

- tkinter
- numpy
- inspect

  Ce module contient les fonctions permettant de gérer la vue de la page de jeu.

5.21.2 Documentation des fonctions

5.21.2.1 vpj_destroy()

```
def src.view.view_pageJeu.vpj_destroy ( )
```

Détruit la page de jeu.
Cette fonction détruit la page de jeu. Elle efface le cadre et le supprime.
Variables :

- tkf_page_jeu : Frame de la page de jeu

Précondition

tkf_page_jeu initialisé

5.21.2.2 vpj_disable_bonus()

```
def src.view.view_pageJeu.vpj_disable_bonus ( )
```

Désactive le bouton bonus.
Cette fonction désactive le bouton bonus et change son relief. Elle est utilisée par le contrôleur de la page de jeu pour désactiver le bouton.

Précondition

TKS_BONUS initialisé

5.21.2.3 vpj_draw_grid()

```
def src.view.view_pageJeu.vpj_draw_grid (
            int rows,
            int columns )
```

Dessine la grille de jeu.
Cette fonction dessine la grille de jeu. Elle prend en paramètre le nombre de lignes et de colonnes de la grille. Elle dessine la grille dans le canvas.
Variables :

- tkc_grid : Canvas de la page de jeu
- i_canvas_width : Largeur du canvas
- i_canvas_height : Hauteur du canvas
- cell_width : Largeur d'une cellule
- cell_height : Hauteur d'une cellule
- ti_upper_left : Coordonnées du coin supérieur gauche d'une cellule
- ti_lower_right : Coordonnées du coin inférieur droit d'une cellule

Paramètres

rows	Nombre de lignes de la grille
columns	Nombre de colonnes de la grille

5.21.2.4 vpj_get_frame()

```
tk.Frame src.view.view_pageJeu.vpj_get_frame ( )
```

Accesseur de la frame de la page de jeu.
Cette fonction renvoie la frame de la page de jeu. Elle est utilisée par le contrôleur principal pour afficher la page de jeu.

Précondition

TKF_PAGE_JEU initialisé

Renvoie

: Frame de la page de jeu

Variables :

- TKF_PAGE_JEU : Frame de la page de jeu

5.21.2.5 vpj_get_grid_cell()

```
(int, int) src.view.view_pageJeu.vpj_get_grid_cell (
            int i_x,
            int i_y )
```

Récupère les coordonnées dans la grille de la cellule cliquée.
Récupère les coordonnées dans la grille de la cellule cliquée en fonction des coordonnées du clic dans le canvas.
Variables :

- i_canvas_width : Largeur du canvas
- i_canvas_height : Hauteur du canvas

Paramètres

i_x	Coordonnée x du clic dans le canvas
i_y	Coordonnée y du clic dans le canvas

Renvoie

: Coordonnées de la cellule cliquée dans la grille

5.21.2.6 vpj_init_page_jeu()

```
def src.view.view_pageJeu.vpj_init_page_jeu (
            tk.Tk tk_root,
            str st_color_grid )
```

Initialise la page de jeu.
Cette fonction initialise la page de jeu. Elle crée un cadre, un canvas pour afficher la grille, un bouton pour annuler le dernier coup, un bouton pour refaire le dernier coup, un bouton pour utiliser un bonus et un bouton pour quitter la partie. Elle affiche aussi le menu sur la fenêtre.
Variables :

- tkf_page_jeu : Frame de la page de jeu
- tkc_grid : Canvas de la page de jeu
- i_canvas_width : Largeur du canvas
- i_canvas_height : Hauteur du canvas
- tkB_undo : Bouton pour annuler le dernier coup
- tkB_redo : Bouton pour refaire le dernier coup
- tkB_bonus : Bouton pour utiliser un bonus
- tkB_quit : Bouton pour quitter la partie

Préconditions :

- tk_root initialisé

Paramètres

tk_root	Fenêtre principale
st_color_grid	Couleur de la grille au format hexadécimal

5.21.2.7 vpj_set_info()

```
def src.view.view_pageJeu.vpj_set_info (
            str st_info )
```

Modifie le texte du label d'information.
Cette fonction modifie le texte du label d'information. Elle prend en paramètre le texte à afficher dans le label d'information. Elle affiche le texte dans le label d'information.

Paramètres

st_info	Texte à afficher dans le label d'information

5.21.2.8 vpj_show_coin()

```
def src.view.view_pageJeu.vpj_show_coin (
            int i_row,
            int i_column,
            str color )
```

Dessine un jeton dans une cellule.
Cette fonction dessine un jeton dans une cellule. Elle prend en paramètre la ligne et la colonne de la cellule où l'on va dessiner le jeton et la couleur du jeton. Elle dessine le jeton dans la cellule.

Paramètres

i_row	Ligne de la cellule où l'on va dessiner le jeton
i_column	Colonne de la cellule où l'on va dessiner le jeton
color	Couleur du jeton

5.22 Référence de l'espace de nommage src.view.view_pageParametres

Un programme qui joue au jeu puissance 4++.

Fonctions

- def vpp_init (tk.Tk tk_root)
  Fonction initialisant la page des paramètres.
- def vpp_init_custom ()
  Fonction initialisant la partie personnalisation des paramètres.
- def vpp_init_settings ()
  Fonction initialisant la partie paramètres du jeu.
- def vpp_get_nb_rows ()
  Accesseur retournant le nombre de lignes sélectionné
- def vpp_get_nb_columns ()
  Accesseur retournant le nombre de colonnes sélectionné
- def vpp_get_nb_jetons ()
  Accesseur retournant le nombre de jetons requis sélectionné
- def vpp_get_difficulty ()
  Accesseur retournant la difficulté sélectionnée.
- def vpp_set_nb_rows (int i_rows)
  Mutateur pour le nombre de lignes.
- def vpp_set_nb_columns (int i_columns)
  Mutateur pour le nombre de colonnes.
- def vpp_set_nb_jetons (int i_nb_jetons)
  Mutateur pour le nombre de jetons requis.
- def vpp_set_difficulty (i_difficulty)
  Mutateur pour la difficulté
- def vpp_reset_settings ()
  Réinitialise les paramètres du jeu.
- def vpp_get_joueur_color ()
  Accesseur retournant la couleur des jetons du joueur.
- def vpp_get_bot_color ()
  Accesseur retournant la couleur des jetons du bot.
- def vpp_get_grid_color ()
  Accesseur retournant la couleur de la grille.
- def vpp_set_joueur_color (str s_color)
  Mutateur pour la couleur des jetons du joueur.
- def vpp_set_bot_color (str s_color)
  Mutateur pour la couleur des jetons du bot.
- def vpp_set_grid_color (str s_color)
  Mutateur pour la couleur de la grille.
- def vpp_reset_customs ()
  Réinitialise les paramètres de personnalisation.
- def vpp_askcolor (str s_element)
  Ouvre un sélecteur de couleur.

5.22.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Vue de la page des paramètres.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

- tkinter
- numpy

— inspect
La vue de la page des paramètres permet de gérer l'affichage de la page des paramètres et traiter les paramètres.

5.22.2 Documentation des fonctions

5.22.2.1 vpp_askcolor()

```
def src.view.view_pageParametres.vpp_askcolor (
            str s_element )
```

Ouvre un sélecteur de couleur.
Cette fonction ouvre un sélecteur de couleur. Elle est appelée lorsque l'utilisateur clique sur le bouton pour choisir la couleur.

Précondition

s_element est soit "joueur", "bot" ou "grille"

Paramètres

s_element	L'élément dont on veut changer la couleur

Postcondition

La couleur de l'élément est modifiée

5.22.2.2 vpp_get_bot_color()

```
def src.view.view_pageParametres.vpp_get_bot_color ( )
```

Accesseur retournant la couleur des jetons du bot.
Cette fonction retourne la couleur des jetons du bot. Elle est appelée lorsque l'utilisateur clique sur le bouton "↩ Enregistrer".

Précondition

TIS_CUSTOM_COLOR_BOT initialisé

Renvoie

La couleur des jetons du bot

Variables :
— TIS_CUSTOM_COLOR_BOT : Tableau d'entier pour la couleur des jetons du bot

5.22.2.3 vpp_get_difficulty()

```
def src.view.view_pageParametres.vpp_get_difficulty ( )
```

Accesseur retournant la difficulté sélectionnée.
Cette fonction retourne la difficulté sélectionnée par l'utilisateur. Elle est appelée lorsque l'utilisateur clique sur le bouton "Enregistrer".

Précondition

TKS_SCALE initialisé

Renvoie

La difficulté sélectionnée

Variables :
— TKS_SCALE : Slider pour la difficulté

5.22.2.4 vpp_get_grid_color()

```
def src.view.view_pageParametres.vpp_get_grid_color ( )
```

Accesseur retournant la couleur de la grille.
Cette fonction retourne la couleur de la grille. Elle est appelée lorsque l'utilisateur clique sur le bouton "Enregistrer".

Précondition

TIS_CUSTOM_COLOR_GRID initialisé

Renvoie

La couleur de la grille

Variables :

— TIS_CUSTOM_COLOR_GRID : Tableau d'entier pour la couleur de la grille

5.22.2.5 vpp_get_joueur_color()

```
def src.view.view_pageParametres.vpp_get_joueur_color ( )
```

Accesseur retournant la couleur des jetons du joueur.
Cette fonction retourne la couleur des jetons du joueur. Elle est appelée lorsque l'utilisateur clique sur le bouton "Enregistrer".

Précondition

TIS_CUSTOM_COLOR_JOUEUR initialisé

Renvoie

La couleur des jetons du joueur

Variables :

— TIS_CUSTOM_COLOR_JOUEUR : Tableau d'entier pour la couleur des jetons du joueur

5.22.2.6 vpp_get_nb_columns()

```
def src.view.view_pageParametres.vpp_get_nb_columns ( )
```

Accesseur retournant le nombre de colonnes sélectionné
Cette fonction retourne le nombre de colonnes sélectionné par l'utilisateur. Elle est appelée lorsque l'utilisateur clique sur le bouton "Enregistrer"

Précondition

STV_COLUMNS initialisé

Renvoie

Le nombre de colonnes sélectionné

Variables :

— STV_COLUMNS : StringVar pour récupérer la valeur du nombre de colonnes

5.22.2.7 vpp_get_nb_jetons()

```
def src.view.view_pageParametres.vpp_get_nb_jetons ( )
```

Accesseur retournant le nombre de jetons requis sélectionné
Cette fonction retourne le nombre de jetons requis sélectionné par l'utilisateur. Elle est appelée lorsque l'utilisateur clique sur le bouton "Enregistrer".

Précondition

STV_NB_JETONS initialisé

Renvoie

Le nombre de jetons requis sélectionné

Variables :

— STV_NB_JETONS : StringVar pour récupérer la valeur du nombre de jetons requis

5.22.2.8 vpp_get_nb_rows()

```
def src.view.view_pageParametres.vpp_get_nb_rows ( )
```

Accesseur retournant le nombre de lignes sélectionné
Cette fonction retourne le nombre de lignes sélectionné par l'utilisateur. Elle est appelée lorsque l'utilisateur clique sur le bouton "Enregistrer"

Précondition

STV_ROWS initialisé

Renvoie

Le nombre de lignes sélectionné

Variables :

— STV_ROWS : StringVar pour récupérer la valeur du nombre de lignes

5.22.2.9 vpp_init()

```
def src.view.view_pageParametres.vpp_init (
            tk.Tk tk_root )
```

Fonction initialisant la page des paramètres.
Cette fonction initialise la page des paramètres. Elle crée un cadre et affiche le menu sur la fenêtre. Elle initialise aussi les paramètres de jeu et de personnalisation.

Variables :

— tkf_page_parameter : Frame de la page des paramètres

Préconditions :

— tk_root initialisé

Paramètres

tk_root	La fenêtre principale

5.22.2.10 vpp_init_custom()

```
def src.view.view_pageParametres.vpp_init_custom ( )
```

Fonction initialisant la partie personnalisation des paramètres.
Cette fonction initialise la partie personnalisation des paramètres. Elle crée des labels pour indiquer les choix de couleurs et des boutons pour ouvrir un sélecteur de couleur. Elle initialise aussi les couleurs par défaut.

Précondition

TKF_PAGE_PARAMETER initialisé

Postcondition

La partie personnalisation des paramètres est initialisée

TKB_PICKER_JOUEUR, TKB_PICKER_BOT et TKB_PICKER_GRID sont initialisés

Variables :

— tkl_perso : Label pour indiquer la seconde partie des paramètres
— tkl_color_joueur : Label pour indiquer le choix de la couleur des jetons du joueur
— TKB_PICKER_JOUEUR : Bouton pour ouvrir un sélectionneur de couleur pour les jetons du joueur

- tkl_color_bot : Label pour indiquer le choix de la couleur des jetons du bot
- TKB_PICKER_BOT : Bouton pour ouvrir un sélectionneur de couleur pour les jetons du bot
- tkl_color_grid : Label pour indiquer le choix de la couleur de la grille
- TKB_PICKER_GRID : Bouton pour ouvrir un sélectionneur de couleur pour la grille
- tkb_save : Bouton pour enregistrer les paramètres
- tkb_reset : Bouton pour réinitialiser les paramètres

5.22.2.11 vpp_init_settings()

```
def src.view.view_pageParametres.vpp_init_settings ( )
```

Fonction initialisant la partie paramètres du jeu.
Cette fonction initialise la partie paramètres du jeu. Elle crée des labels pour indiquer les choix de paramètres et des spinbox pour choisir les paramètres. Elle initialise aussi les paramètres par défaut.

Précondition

TKF_PAGE_PARAMETER initialisé

Postcondition

La partie paramètres du jeu est initialisée

Variables :

- tkl_param : Label pour indiquer la première partie des paramètres
- tkl_size : Label pour indiquer le choix de la taille de la grille
- STV_ROWS : StringVar pour récupérer la valeur du nombre de lignes
- tksb_nb_rows : Spinbox pour le nombre de lignes
- tkl_lines : Label pour indiquer le nombre de lignes
- STV_COLUMNS : StringVar pour récupérer la valeur du nombre de colonnes
- tksb_nb_columns : Spinbox pour le nombre de colonnes
- tkl_colonnes : Label pour indiquer le nombre de colonnes
- tkl_nb_jetons : Label pour indiquer le choix du nombre de jetons requis
- STV_NB_JETONS : StringVar pour récupérer la valeur du nombre de jetons requis
- tksb_nb_jetons : Spinbox pour le nombre de jetons requis
- tkl_difficulty : Label pour indiquer le choix de la difficulté
- TKS_SCALE : Slider pour la difficulté
- tkb_save : Bouton pour enregistrer les paramètres
- tkb_reset : Bouton pour réinitialiser les paramètres

5.22.2.12 vpp_reset_customs()

```
def src.view.view_pageParametres.vpp_reset_customs ( )
```

Réinitialise les paramètres de personnalisation.
Cette fonction réinitialise les paramètres de personnalisation. Elle est appelée lorsque l'utilisateur clique sur le bouton "Réinitialiser".

5.22.2.13 vpp_reset_settings()

```
def src.view.view_pageParametres.vpp_reset_settings ( )
```

Réinitialise les paramètres du jeu.
Cette fonction réinitialise les paramètres du jeu. Elle est appelée lorsque l'utilisateur clique sur le bouton "↩ Réinitialiser".

5.22.2.14 vpp_set_bot_color()

```
def src.view.view_pageParametres.vpp_set_bot_color (
            str s_color )
```

Mutateur pour la couleur des jetons du bot.
Cette fonction modifie la couleur des jetons du bot. Elle est appelée lorsque l'utilisateur clique sur le bouton "↩ Réinitialiser".

Précondition

TIS_CUSTOM_COLOR_BOT initialisé

Paramètres

s_color	La nouvelle couleur des jetons du bot

Postcondition

TIS_CUSTOM_COLOR_BOT est modifié

Variables :

- TIS_CUSTOM_COLOR_BOT : Tableau d'entier pour la couleur des jetons du bot
- s_color : La couleur des jetons du bot à modifier
- TKB_PICKER_BOT : Bouton pour ouvrir le sélecteur de couleur

5.22.2.15 vpp_set_difficulty()

```
def src.view.view_pageParametres.vpp_set_difficulty (
              i_difficulty )
```

Mutateur pour la difficulté
Cette fonction modifie la difficulté sélectionnée par l'utilisateur. Elle est appelée lorsque l'utilisateur clique sur le bouton "Réinitialiser".

Précondition

TKS_SCALE initialisé

Paramètres

i_difficulty	La nouvelle difficulté

Postcondition

TKS_SCALE est modifié

Variables :

- TKS_SCALE : Slider pour la difficulté
- i_difficulty : La difficulté à modifier

5.22.2.16 vpp_set_grid_color()

```
def src.view.view_pageParametres.vpp_set_grid_color (
             str s_color )
```

Mutateur pour la couleur de la grille.
Cette fonction modifie la couleur de la grille. Elle est appelée lorsque l'utilisateur clique sur le bouton "Réinitialiser".

Précondition

TIS_CUSTOM_COLOR_GRID initialisé

Paramètres

s_color	La nouvelle couleur de la grille

Postcondition

TIS_CUSTOM_COLOR_GRID est modifié

Variables :

— TIS_CUSTOM_COLOR_GRID : Tableau d'entier pour la couleur de la grille
— s_color : La couleur de la grille à modifier
— TKB_PICKER_GRID : Bouton pour ouvrir le sélecteur de couleur

5.22.2.17 vpp_set_joueur_color()

```
def src.view.view_pageParametres.vpp_set_joueur_color (
            str s_color )
```

Mutateur pour la couleur des jetons du joueur.
Cette fonction modifie la couleur des jetons du joueur. Elle est appelée lorsque l'utilisateur clique sur le bouton "Réinitialiser".

Précondition

TIS_CUSTOM_COLOR_JOUEUR initialisé

Paramètres

s_color	La nouvelle couleur des jetons du joueur

Postcondition

TIS_CUSTOM_COLOR_JOUEUR est modifié

Variables :
— TIS_CUSTOM_COLOR_JOUEUR : Tableau d'entier pour la couleur des jetons du joueur
— s_color : La couleur des jetons du joueur à modifier
— TKB_PICKER_JOUEUR : Bouton pour ouvrir le sélecteur de couleur

5.22.2.18 vpp_set_nb_columns()

```
def src.view.view_pageParametres.vpp_set_nb_columns (
            int i_columns )
```

Mutateur pour le nombre de colonnes.
Cette fonction modifie le nombre de colonnes sélectionné par l'utilisateur. Elle est appelée lorsque l'utilisateur clique sur le bouton "Réinitialiser".

Précondition

STV_COLUMNS initialisé∗

Paramètres

i_columns	Le nouveau nombre de colonnes

Postcondition

STV_COLUMNS est modifié

Variables :
— STV_COLUMNS : StringVar pour récupérer la valeur du nombre de colonnes
— i_columns : Le nombre de colonnes à modifier

5.22.2.19 vpp_set_nb_jetons()

```
def src.view.view_pageParametres.vpp_set_nb_jetons (
            int i_nb_jetons )
```

Mutateur pour le nombre de jetons requis.
Cette fonction modifie le nombre de jetons requis sélectionné par l'utilisateur. Elle est appelée lorsque l'utilisateur clique sur le bouton "Réinitialiser".

Précondition

STV_NB_JETONS initialisé

Paramètres

i_nb_jetons	Le nouveau nombre de jetons requis

Postcondition

STV_NB_JETONS est modifié

Variables :
- STV_NB_JETONS : StringVar pour récupérer la valeur du nombre de jetons requis
- i_nb_jetons : Le nombre de jetons requis à modifier

5.22.2.20 vpp_set_nb_rows()

```
def src.view.view_pageParametres.vpp_set_nb_rows (
            int i_rows )
```

Mutateur pour le nombre de lignes.
Cette fonction modifie le nombre de lignes sélectionné par l'utilisateur. Elle est appelée lorsque l'utilisateur clique sur le bouton "Réinitialiser".

Précondition

STV_ROWS initialisé

Paramètres

i_rows	Le nouveau nombre de lignes

Postcondition

STV_ROWS est modifié

Variables :
- STV_ROWS : StringVar pour récupérer la valeur du nombre de lignes
- i_rows : Le nombre de lignes à modifier

5.23 Référence de l'espace de nommage tests

Espaces de nommage

- test_grid
  Un programme qui joue au jeu puissance 4++.
- test_puissanceQuatre
  Un programme qui joue au jeu puissance 4++.

5.24 Référence de l'espace de nommage tests.test_grid

Un programme qui joue au jeu puissance 4++.

Fonctions

- def tg_init_grille ()
  Teste la fonction pq_init_grille.
- def tg_test_all ()
  Lance tous les tests.

5.24.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Teste le module puissanceQuatre.grid.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

- tkinter
- numpy
- inspect
  Ce module teste le module puissanceQuatre.grid.

5.24.2 Documentation des fonctions

5.24.2.1 tg_init_grille()

```
def tests.test_grid.tg_init_grille ( )
```

Teste la fonction pq_init_grille.
Variables :

- *liste_tailles* : liste des tailles de npa_grille à tester
- *i_boucle* : variable de boucle
- *i_boucle_2* : variable de boucle
- *npa_grille* : npa_grille de jeu

Test Vérifie que la npa_grille est bien initialisée avec des 0 partout avec la bonne taille
Vérifie que toutes les combinaisons de 2 nombres de la liste *liste_tailles* sont testées

5.24.2.2 tg_test_all()

```
def tests.test_grid.tg_test_all ( )
```

Lance tous les tests.

5.25 Référence de l'espace de nommage tests.test_puissanceQuatre

Un programme qui joue au jeu puissance 4++.

Fonctions

- def tp_verif_colonne ()
  Teste la fonction pq_verif_colonne.
- def tp_ajout_piece ()
  Teste la fonction pq_ajout_piece.
- def tp_victoire_ligne ()
  Test de la fonction pq_victoire_ligne.
- def tp_victoire_colonne ()
  Test de la fonction pq_victoire_colonne.
- def tp_victoire_diago ()
  Test de la fonction pq_victoire_diagonale.
- def tp_test_all ()
  Fonction qui lance tous les tests unitaires.

5.25.1 Description détaillée

Un programme qui joue au jeu puissance 4++.
Teste le module puissanceQuatre.puissanceQuatre.
Ce programme est un jeu de puissance 4++ avec une grille de taille variable, un nombre de pions à aligner variable, des bonus et un undo.
Ce programme utilise les modules externes suivants :

— tkinter
— numpy
— inspect
Ce module teste le module puissanceQuatre.puiissanceQuatre.

5.25.2 Documentation des fonctions

5.25.2.1 tp_ajout_piece()

```
def tests.test_puissanceQuatre.tp_ajout_piece ( )
```
Teste la fonction pq_ajout_piece.
Variables :
— *npa_grille* : npa_grille de jeu
— *ti_coords* : tuple de coordonnées
— *i_boucle* : variable de boucle

Test Vérifie que la fonction renvoie les bonnes coordonnées quand le joueur joue
Vérifie que la fonction renvoie les bonnes coordonnées quand le bot joue
Vérifie que la fonction renvoie les bonnes coordonnées quand le joueur joue normalement
Vérifie que la fonction ne renvoie pas de coordonnées quand on ne peut pas ajouter de pièce

5.25.2.2 tp_test_all()

```
def tests.test_puissanceQuatre.tp_test_all ( )
```
Fonction qui lance tous les tests unitaires.

5.25.2.3 tp_verif_colonne()

```
def tests.test_puissanceQuatre.tp_verif_colonne ( )
```
Teste la fonction pq_verif_colonne.
Variables :
— *npa_grille* : npa_grille de jeu
— *i_boucle* : variable de boucle
— *i_boucle_2* : variable de boucle

Test Vérifie que la fonction renvoie True si la colonne est vide
Vérifie que la fonction renvoie False si la colonne est pleine
Vérifie que la fonction renvoie True si la colonne est presque pleine
Vérifie que la fonction renvoie True si la colonne est un peu remplie
Vérifie que la fonction renvoie False si la npa_grille est pleine

5.25.2.4 tp_victoire_colonne()

```
def tests.test_puissanceQuatre.tp_victoire_colonne ( )
```
Test de la fonction pq_victoire_colonne.
Variables :
— *npa_grille* : Grille du jeu
— *i_boucle* : Variable de boucle
— *ti_coords* : Tuple de coordonnées

Test Vérifie que la fonction renvoie True si le joueur gagne
Vérifie que la fonction renvoie True si le bot gagne
Vérifie que la fonction renvoie False si personne ne gagne

5.25.2.5 tp_victoire_diago()

```
def tests.test_puissanceQuatre.tp_victoire_diago ( )
```

Test de la fonction pq_victoire_diagonale.

Variables :

- *npa_grille* : Grille du jeu
- *i_boucle* : Variable de boucle
- *i_boucle_2* : Variable de boucle
- *ti_coords* : Tuple de coordonnées

Test Vérifie que la fonction renvoie True si le joueur gagne
Vérifie que la fonction renvoie True si le bot gagne
Vérifie que la fonction renvoie False si personne ne gagne

5.25.2.6 tp_victoire_ligne()

```
def tests.test_puissanceQuatre.tp_victoire_ligne ( )
```

Test de la fonction pq_victoire_ligne.

Variables :

- *npa_grille* : Grille du jeu
- *i_boucle* : Variable de boucle
- *ti_coords* : Tuple de coordonnées

Test Vérifie que la fonction renvoie True si le joueur gagne
Vérifie que la fonction renvoie True si le bot gagne
Vérifie que la fonction renvoie False si personne ne gagne

Index