PyEngine

Version 1.0.0

juil. 28, 2019

Informations

Bienvenue sur la documentation de la bibliothèque PyEngine.

PyEngine est constamment en développement, la documentation est donc susceptible d'être changée. N'hésitez pas à y revenir dès que vous avez un problème.

Note : Il est important de rappeler que PyEngine est un projet OpenSource et développé par des personnes non professionnelles. Vous pouvez, vous aussi, y participer via le github.

Sommaire :

CHAPITRE 1

Introduction

PyEngine a été créé par LavaPower.

PyEngine se base sur PyGame pour fonctionner. Il a été fait pour être utilisé sur des jeux 2D de tout type : Platformer, Pong, Casse bricks...

Vous pouvez retrouver des tutoriels, des exemples et la documentation des différentes classes.

Note : PyEngine est encore très jeune et encore très limité.

Note : Cette documentation liste les méthodes des classes qui peuvent être utilisées sans problème et sans risque.

CHAPITRE 2

Téléchargement et Installation

2.1 Dernière release (Méthode simple)

— Avoir Python et Pip installés
— Faire dans une console : *pip install PyEngine-2D*
— PyEngine est téléchargé et installé

2.2 Version en développement (Méthode moins simple)

— Avoir Python et Pip installés
— Télécharger et décompresser les fichiers du github (http://github.com/pyengine-2D/PyEngine)
— **Faire dans une console à l'endroit où sont les fichiers :** *python setup.py install*.
— PyEngine est téléchargé et installé

CHAPITRE 3

FAQ

3.1 Qu'est-ce que PyEngine ?

PyEngine est une bibliothèque python permettant de créer des jeux vidéos 2D plus facilement. C'est une sorte de moteur de jeu très simplifié sans interface.

3.2 Pourquoi créer PyEngine ?

Pour créer un jeu vidéo en python, il existe déjà le très bon PyGame.

Mais en créant mon jeu, je devais créer des systèmes (comme le système d'entité) qui sont pourtant utiles pour tous. J'ai donc fait le choix de créer PyEngine (qui utilise PyGame lui même) (Et puis ça permet un bon entrainement en Python).

3.3 Quelles sont les dépendances de PyEngine ?

Si ce n'est Python, PyEngine utilise PyGame.

3.4 Quelles sont les plateformes où PyEngine est utilisable ?

Si vous pouvez utiliser PyGame et Python, vous pouvez utiliser PyEngine.

3.5 Je souhaite participer au développement de PyEngine, comment faire ?

Envoyez moi un message par Discord (LavaPower#2480) pour voir ce que vous pouvez faire.

CHAPITRE 4

Hello World

Dans ce premier tutoriel, nous allons créer un programme très connu : le fameux Hello World.

Grace à ce tutoriel, vous saurez créer une fenêtre graphique avec un état de jeu. De plus, vous saurez utiliser le widget Label venant du système d'UI.

4.1 Création de la fenêtre

La première étape est de créer la fenêtre graphique. Ici on va créer une fenêtre de 500 par 300 pixels avec un fond blanc.

Tout d'abord, il faut importer la classe de la fenêtre via :

```
from pyengine import Window # Window étant la classe de notre fenêtre.
```

Ensuite, il faut l'initialiser :

```
fenetre = Window(500, 300, (255, 255, 255))
# 500 : Largeur
# 300 : Longueur
# (255, 255, 255) : Couleur blanche
```

Si vous lancez ce code, vous verrez la fenêtre se lancer puis se fermer directement.

Pour régler ce problème, il faut lancer la boucle de la fenêtre. Pour cela, il suffit de faire :

```
fenetre.run()
```

Mais ici, vous avez une erreur. Plus précisément, une NoObjectError. Ceci s'explique par le fait que vous essayez de lancer la boucle d'une fenêtre qui n'a pas de GameState.

Note : Une GameState est un état de votre jeu/programme. Par exemple, dans un flappy bird, il y a plusieurs états : le moment où il y a le menu, le moment où on jeu, le moment de fin de jeu...

4.2 Création de la GameState

Actuellement vous avez ceci :

```
from pyengine import Window

fenetre = Window(500, 300, (255, 255, 255))
fenetre.run()
```

Pour créer votre GameState, il va falloir importer puis utiliser sa classe :

```
from pyengine import GameState

state = GameState("HelloWorld")
# "HelloWorld" correspond au nom de votre state.
```

Ensuite, il faut l'ajouter à votre fenêtre via un :

```
fenetre.add_state(state)
```

En organisant bien votre code, vous devriez avoir quelque chose de ce style :

```
from pyengine import Window, GameState

fenetre = Window(500, 300, (255, 255, 255))
state = GameState("HelloWorld")

fenetre.add_state(state)
fenetre.run()
```

Lancez le programme et vous devriez avoir ceci :

Tutoriels/../images/HelloWorld1.png

4.3 Création du texte

Maintenant, nous allons afficher notre texte.

Pour cela, nous allons utiliser le monde de notre GameState afin de récupérer le système qui gère l'ui.

```
from pyengine.Systems import UISystem

uisystem = state.get_world().get_system(UISystem)
```

Ensuite, nous devons créer notre widget et l'ajouter à notre système :

```
from pyengine.Widgets import Label

hello = Label([0, 0], "Hello World !", (0, 0, 0))
# [0, 0] : Position x, y
```

(suite sur la page suivante)

(suite de la page précédente)

```
# "Hello World !" : Texte
# (0, 0, 0) : Couleur noire
uisystem.add_widget(hello)
```

Ce qui nous donne au final :

```
from pyengine import Window, GameState
from pyengine.Systems import UISystem
from pyengine.Widgets import Label

fenetre = Window(500, 300, (255, 255, 255))
state = GameState("HelloWorld")

fenetre.add_state(state)

uisystem = state.get_world().get_system(UISystem)

hello = Label([0, 0], "Hello World !", (0, 0, 0))
uisystem.add_widget(hello)

fenetre.run()
```

Avec comme résultat :

Tutoriels/../images/HelloWorld2.png

CHAPITRE 5

Enumérations

PyEngine incorpore beaucoup d'énumérations utilisées dans ses différentes classes. En voici la liste :

5.1 ControlType

Description Correspond aux différents types de contrôles de la classe ControlComponent

Valeurs

- FOURDIRECTION : L'objet se déplace dans les quatres directions.
- CLASSICJUMP : L'objet se déplace latéralement et fait un saut simple
- DOUBLEJUMP : Comme le CLASSICJUMP mais avec un double saut
- CLICKFOLLOW : L'objet se déplace vers l'endroit du clic de la souris
- LEFTRIGHT : L'objet se déplace latéralement, sans saut
- UPDOWN : L'objet se déplace de bas en haut

5.2 Controls

Description Correspond aux contrôles utilisables dans PyEngine

Valeurs

- UPJUMP : Direction vers le haut (sert aussi au saut)
- LEFT : Direction vers la gauche
- RIGHT : Direction vers la droite
- DOWN : Direction vers le bas

5.3 MouseButton

Description Correspond aux différents boutons de la souris

Valeurs

— LEFTCLICK : Clic gauche
— MIDDLECLICK : Clic molette
— RIGHTCLICK : Clic droit

5.4 CollisionCauses

Description Correspond aux différentes causes d'une collision

Valeurs

— UNKNOWN : Cause inconnu
— GRAVITY : Causée par la gravité
— LEFTCONTROL : Causée par un déplacement à gauche du ControlComponent
— RIGHTCONTROL : Causée par un déplacement à droite du ControlComponent
— UPCONTROL : Causée par un déplacement en haut du ControlComponent
— DOWNCONTROL : Causée par un déplacement en bas du ControlComponent
— MOVECOMPONENT : Causée par un déplacement du MoveComponent

Note : Si une collision à lieu à cause d'un saut, c'est la cause GRAVITY qui sera affichée.

5.5 WorldCallbacks

Description Correspond aux différents types de callback renvoyé par le monde

Valeurs

— OUTOFWINDOW : Activé quand un élément dépasse les bords de l'écran

Note : Un callback est simplement une fonction lancée suivant des évènements précis.

CHAPITRE 6

Exceptions

PyEngine incorpore beaucoup d'exceptions utilisées dans ses différentes classes. En voici la liste :

6.1 NoObjectError

Description L'objet n'existe pas.

6.2 WrongObjectError

Description L'objet ne peut pas être utilisé à cet endroit.

6.3 CompatibilityError

Description L'objet n'est pas compatible avec un autre objet déjà mis en place.

CHAPITRE 7

Window

Cette classe correspond à la fenêtre ouverte par votre jeu.

7.1 Constructeur

Description Crée l'objet Window

Paramètres

- width <integer> : Largeur de la fenêtre
- height <integer> : Hauteur de la fenêtre
- color <list> (None) : Couleur de fond
- debug <boolean> (False) : Mode debug

Note : Si color est égal à None, color vaut (0, 0, 0) (soit noir)

Voici ces méthodes :

7.2 set_color

Description Change la couleur de la fenêtre

Retourne Rien

Paramètre color <list> : Couleur de la fenêtre

7.3 get_color

Description Récupère la couleur de la fenêtre

Retourne <list> : Couleur de la fenêtre

Paramètre Rien

7.4 set_debug

Description Change si la fenêtre est en mode debug
Retourne Rien
Paramètre debug <boolean> : Mode debug de la fenêtre

7.5 get_debug

Description Vérifie que la fenêtre est en mode debug
Retourne <boolean> : Mode debug de la fenêtre
Paramètre Rien

7.6 set_title

Description Change le titre de la fenêtre
Retourne Rien
Paramètre title <string> : Nouveau titre de la fenêtre

7.7 get_title

Description Récupère le titre de la fenêtre
Retourne <string> : Titre de la fenêtre
Paramètre Rien

7.8 add_state

Description Ajoute un GameState à la fenêtre
Retourne Rien
Paramètre state <GameState> : GameState à ajouter

7.9 set_current_state

Description Définit la GameState actuelle
Retourne Rien
Paramètre name <string> : Nom de la GameState à définir comme actuelle

7.10 get_current_state

Description Récupère la GameState actuelle
Retourne <GameState> : GameState actuelle
Paramètre Rien

7.11 get_state

Description Récupère une GameState à partir de son nom
Retourne <GameState|None> : GameState dont le nom est <name> ou Rien si elle n'existe pas.
Paramètre name <string> : Nom de la GameState à récupérer

7.12 stop

Description Arrête le jeu
Retourne Rien
Paramètre Rien

7.13 run

Description Lance le jeu
Retourne Rien
Paramètre Rien

Avertissement : Peut retourner les exceptions : NoObjectError

CHAPITRE 8

GameState

Cette classe correspond à un état précis de votre jeu

8.1 Constructeur

Description Crée l'objet GameState
Paramètre name <string> : Nom de la GameState

Voici ces méthodes :

8.2 set_world

Description Définit le monde de la GameState
Retourne Rien
Paramètre world <World> : Nouveau monde de la GameState

8.3 get_world

Description Récupère le monde de la GameState
Retourne <World> : Monde de la GameState
Paramètres Rien

CHAPITRE 9

World

Cette classe correspond au monde d'une GameState

9.1 Constructeur

Description Crée l'objet World

Paramètre Rien

Voici ces méthodes :

9.2 set_callback

Description Définit un Callback du monde

Retourne Rien

Paramètres

— callback <WorldCallbacks> : Callback à définir
— function <Function> : Fonction lancée au moment du callback

Les callsbacks peuvent demander des paramètres. Il faut donc les fournir dans la fonction lancée.

OUTOFWINDOW

— <Entity> - Entité qui dépasse les bords
— <List> - position de l'entité

Note : Un callback est simplement une fonction lancée suivant des évènements précis.

Avertissement : Peut retourner une exception : TypeError

9.3 get_system

Description Récupère un système du monde

Retourne <EntitySystem|MusicSystem|UISystem|None> : Système du type <classe> s'il existe

Paramètre <EntitySystem|MusicSystem|UISystem> : Classe du système à récupérer

9.4 has_system

Description Vérifie l'existence d'un système dans le monde

Retourne <bool> : Vrai si le monde a le système. Sinon Faux

Paramètre <EntitySystem|MusicSystem|UISystem> : Classe du système à récupérer

CHAPITRE 10

Entity

Cette classe correspond à une entité de votre jeu.

10.1 Constructeur

Description Crée l'objet Entity
Paramètre Rien

Voici ces méthodes :

10.2 get_id

Description Récupère l'id de l'entité
Retourne <int> : Id de l'entité
Paramètre Rien

10.3 attach_entity

Description Attache une entité à l'entité
Retourne Rien
Paramètre entity <Entity> : Entité à attacher

10.4 add_component

Description Ajoute un composant à l'entité

Retourne <Components> : Composant ajouté
Paramètre component <Components> : Composant à ajouter

Note : Components fait référence à toutes les classes étant des composants.

Avertissement : Peut retourner une exception : WrongObjectError

10.5 has_component

Description Vérifie l'existence d'un composant dans l'entité
Retourne <bool> : Vrai si l'entité a le composant. Sinon Faux
Paramètre component <Components> : Composant à ajouter

Note : Components fait référence à toutes les classes étant des composants.

10.6 get_component

Description Récupère un composant de l'entité
Retourne <Components|None> : Composant dont le type est <component> s'il existe
Paramètre component <Components> : Composant à récupérer

Note : Components fait référence à toutes les classes étant des composants.

CHAPITRE 11

MusicSystem

Cette classe correspond au système de musique d'un monde

Avertissement : La gestion des .mp3 est un peu bugguée. Si votre mp3 ne fonctionne pas correctement, utilisez une autre extension (exemple : .wav)

Voici ses méthodes :

11.1 next_song

Description Passe à la musique suivante
Retourne Rien
Paramètre Rien

11.2 clear_queue

Description Vide la queue
Retourne Rien
Paramètre Rien

11.3 set_loop

Description Définit si la queue se rejoue
Retourne Rien
Paramètre loop <bool> : Vrai si la queue se rejoue. Sinon faux

11.4 play

Description Lance la musique
Retourne Rien
Paramètre Rien

Avertissement : Peut retourner une exception : NoObjectError

11.5 add

Description Ajoute une musique à la queue
Retourne Rien
Paramètre file <string> : Chemin vers le fichier de la musique

11.6 set_volume

Description Définit le volume du système
Retourne Rien
Paramètre volume <int> : Volume du système

Note : Le volume doit être compris entre 0 et 100

Avertissement : Peut retourner une exception : ValueError

11.7 stop

Description Arrête la musique
Retourne Rien
Paramètre Rien

11.8 pause

Description Met en pause la musique
Retourne Rien
Paramètre Rien

11.9 unpause

Description Relance la musique
Retourne Rien
Paramètre Rien

CHAPITRE 12

UISystem

Cette classe correspond au système d'interface utilisateur du monde

Voici ses méthodes :

12.1 add_widget

Description Ajoute un widget au système

Retourne <Widgets> : Widget ajouté

Paramètre widget <Widgets> : Widget à ajouter

Note : Widgets fait référence à toutes les classes étant des widgets.

12.2 get_widget

Description Récupère un widget au système

Retourne <Widgets|None> : Widget s'il existe ou rien

Paramètre widget <Widgets> : Classe du widget à récupérer

CHAPITRE 13

EntitySystem

Cette classe correspond au système d'entité du monde

Voici ses méthodes :

13.1 get_entity

Description Récupère une entité

Retourne <Entity|None> : Entité si elle existe ou rien.

Paramètre id <int> : Id de l'entité à récupérer

13.2 add_entity

Description Ajoute une entité au système

Retourne <Entity> : Entité ajoutée

Paramètre entity <Entity> : Entité à ajouter

Avertissement : Peut retourner une exception : NoObjectError

CHAPITRE 14

PositionComponent

Cette classe permet de définir une position à l'entité

14.1 Constructeur

Description Crée l'objet PositionComponent

Paramètres

— position <list> : Position de l'entité
— offset <list> ([0, 0]) : Offset de l'entité

Note : L'offset n'est utile que dans le cas d'entité attachée à une autre entité.

Voici ses méthodes :

14.2 get_position

Description Récupère la position

Retourne <list> : Position de l'entité

Paramètre Rien

14.3 set_position

Description Définit la position

Retourne Rien

Paramètre position <list> : Position de l'entité

CHAPITRE 15

SpriteComponent

Cette classe permet de définir un sprite à l'entité

Avertissement : Non compatible avec le TextComponent

15.1 Constructeur

Description Crée l'objet SpriteComponent

Paramètres

- image <string> : Chemin vers le sprite
- scale <integer> (1) : « Zoom » sur le sprite
- rotation <integer> (0) : Rotation du sprite

Avertissement : Peut retourner une exception : CompatibilityError

Voici ses méthodes :

15.2 set_scale

Description Change le « zoom » du sprite

Retourne Rien

Paramètre scale <integer> : « Zoom » du sprite

15.3 set_size

Description Change la taille du sprite
Retourne Rien
Paramètre size <list> : taille du sprite

Note : Réinitialise la scale du sprite

15.4 set_rotation

Description Change la rotation du sprite
Retourne Rien
Paramètre rotation <integer> : Rotation du sprite

15.5 set_sprite

Description Change le sprite
Retourne Rien
Paramètre sprite <string> : Chemin vers le sprite

CHAPITRE 16

TextComponent

Cette classe permet de définir un texte à l'entité

Avertissement : Non compatible avec le SpriteComponent

16.1 Constructeur

Description Crée l'objet TextComponent

Paramètres

- texte <string> : Texte à afficher
- color <list> ((255, 255, 255)) : Couleur du texte
- font <list> ([« arial », 15, False, False]) : Police du texte

Note : La police est composée comme ceci : [Nom, Taille, Gras, Italique]. Vous pouvez omettre des éléments mais seulement dans l'ordre.

Exemple : Vous pouvez écrire [Nom, Taille] mais pas [Nom, Gras, Italique]

Avertissement : Peut retourner une exception : CompatibilityError

Note : Pas de méthode utilisable avec ce composant

CHAPITRE 17

PhysicsComponent

Cette classe permet de définir une physique à l'entité

17.1 Constructeur

Description Crée l'objet PhysicsComponent

Paramètres

— affectbygravity <bool> (True) : Affecté par la gravité ou non
— gravity_force <int> (5) : Puissance de la gravité

Voici ces méthodes :

17.2 get_gravity

Description Récupère la force de gravité

Retourne <int> : Force de gravité

Paramètre Rien

17.3 set_gravity

Description Définit la force de gravité

Retourne Rien

Paramètre gravity <int> : Force de gravité

17.4 set_callback

Description Définit le callback de la collision

Retourne Rien

Paramètres

— function <Function> : Fonction lancée au moment du callback

---

Note : Un callback est simplement une fonction lancée suivant des évènements précis (Ici une collision).

CHAPITRE 18

MoveComponent

Cette classe permet de faire bouger automatiquement une entité

18.1 Constructeur

Description Crée l'objet MoveComponent

Paramètres

— direction <list> : Direction du mouvement

— speed <int> (5) : Vitesse du mouvement

Note : direction est une liste où le premier nombre correspond au mouvement en x et le second en y. Cette liste ne doit contenir que des 1, -1 et 0. Exemple : (0, 1) créera un mouvement vers le bas

Voici ses méthodes :

18.2 get_speed

Description Récupère la vitesse

Retourne <int> : Vitesse du mouvement

Paramètre Rien

18.3 set_speed

Description Définit la vitesse

Retourne Rien

Paramètre speed <int> : Vitesse du mouvement

18.4 get_direction

Description Récupère la direction
Retourne <list> : Direction du mouvement
Paramètre Rien

18.5 set_direction

Description Définit la direction
Retourne Rien
Paramètre direction <list> : Direction du mouvement

CHAPITRE 19

ControlComponent

Cette classe permet de contrôler l'entité

19.1 Constructeur

Description Crée l'objet ControlComponent

Paramètres
- controltype <ControlType> : Type de contrôle
- speed <integer> (5) : Vitesse du mouvement

Voici ses méthodes :

19.2 get_control

Description Récupère un contrôle

Retourne <int> : Constante correspondant à la touche

Paramètre <Controls> : Contrôle à récupérer

19.3 set_control

Description Définit un contrôle

Retourne Rien

Paramètre
- name <Controls> : Contrôle à récupérer
- key <int> : Constante correspondant à la touche

19.4 get_speed

Description Récupère la vitesse
Retourne <int> : Vitesse du mouvement
Paramètre Rien

19.5 set_speed

Description Définit la vitesse
Retourne Rien
Paramètre speed <int> : Vitesse du mouvement

CHAPITRE 20

LifeBarComponent

Cette classe permet de définir une vie à l'entité et, optionnellement, une barre de vie.

20.1 Constructeur

Description Crée l'objet LifeBarComponent

Paramètres

- maxlife <int> (100) : Vie maximum
- sprites <list> (None) : Sprites de la barre de vie
- offset <list> (None) : Offset de la barre de vie

Voici ces méthodes :

20.2 create_life_sprites

Description Crée les entités de la barre de vie

Retourne Rien

Paramètre Rien

Avertissement : Utilisable seulement si sprites a été défini dans le Constructeur

CHAPITRE 21

Widget

Avertissement : Cette classe ne devrait pas être utilisée à l'état actuelle. Elle sert de base pour créer les autres widgets

Note : Les fonctions ajoutées par cette classe sont définies dans les autres pages de la documentation pour éviter les erreurs.

CHAPITRE 22

Label

Cette classe permet d'afficher un texte

22.1 Constructeur

Description Crée l'objet Label

Paramètres
- position <list> : Position du widget
- texte <string> : Texte à afficher
- color <list> ((255, 255, 255)) : Couleur du texte
- font <list> ([« arial », 15, False, False]) : Police du texte

Note : La police est composée comme ceci : [Nom, Taille, Gras, Italique]. Vous pouvez omettre des éléments mais seulement dans l'ordre.

Exemple : Vous pouvez écrire [Nom, Taille] mais pas [Nom, Gras, Italique]

22.2 set_color

Description Définit la couleur du texte

Retourne Rien

Paramètre color <list> : Couleur (RGB ou RGBA)

22.3 get_color

Description Récupère la couleur du texte

Retourne <list> : Couleur (RGB ou RGBA)
Paramètre Rien

22.4 set_text

Description Définit le texte
Retourne Rien
Paramètre text <string> : Texte

22.5 get_text

Description Récupère le texte
Retourne <string> : Texte
Paramètre Rien

22.6 set_font

Description Définit la police
Retourne Rien
Paramètre police <list> : Police

Note : La police est composée comme ceci : [Nom, Taille, Gras, Italique]. Vous pouvez omettre des éléments mais seulement dans l'ordre.

Exemple : Vous pouvez écrire [Nom, Taille] mais pas [Nom, Gras, Italique]

22.7 get_font

Description Récupère la police
Retourne <list> : Police
Paramètre Rien

Note : La police est composée comme ceci : [Nom, Taille, Gras, Italique]

22.8 get_id

Description Récupère l'id du widget
Retourne <int> : Id du widget
Paramètre Rien

Note : L'id vaut -1 si le widget n'a pas été ajouté à un System.

22.9 get_system

Description Récupère le système du widget
Retourne <UISystem|None> : Système du widget si il a été ajouté à un UISystem
Paramètre Rien

22.10 get_position

Description Récupère la position du widget
Retourne <list> : Position du widget
Paramètre Rien

22.11 set_position

Description Définit la position du widget
Retourne Rien
Paramètre <list> : Position du widget

CHAPITRE 23

Image

Cette classe permet d'afficher une image

23.1 Constructeur

Description Crée l'objet Image

Paramètres

- position <list> : Position du widget
- image <string> : Chemin vers l'image
- size <list> (None) : Taille de l'image

Note : Si size vaut None, l'image ne sera pas redimensionnée.

23.2 get_image

Description Récupère le chemin de l'image

Retourne <string> : Chemin de l'image

Paramètre Rien

23.3 set_image

Description Définit l'image

Retourne Rien

Paramètre <string> : Chemin de l'image

23.4 get_size

Description Récupère la taille de l'image
Retourne <list> : Taille de l'image
Paramètre Rien

23.5 set_size

Description Définit la taille de l'image
Retourne Rien
Paramètre <list> : Taille de l'image

23.6 get_id

Description Récupère l'id du widget
Retourne <int> : Id du widget
Paramètre Rien

Note : L'id vaut -1 si le widget n'a pas été ajouté à un System.

23.7 get_system

Description Récupère le système du widget
Retourne <UISystem|None> : Système du widget si il a été ajouté à un UISystem
Paramètre Rien

23.8 get_position

Description Récupère la position du widget
Retourne <list> : Position du widget
Paramètre Rien

23.9 set_position

Description Définit la position du widget
Retourne Rien
Paramètre <list> : Position du widget

CHAPITRE 24

Button

Cette classe permet de d'afficher un bouton

24.1 Constructeur

Description Crée l'objet Button

Paramètres

- position <list> : Position du widget
- text <string> : Texte sur le bouton
- command <Function> (None) : Fonction lancé à l'appui du bouton
- size <list> ([100, 40]) : Taille du bouton
- image <string> (None) : Image du bouton

Note : Si l'image n'est pas spécifié, le bouton aura un rectangle gris comme fond.

24.2 get_label

Description Récupère le label du bouton (le texte du bouton)

Retourne <Label> : Label du bouton

Paramètre Rien

Note : Cela permet de modifier le texte (police et couleur compris) du bouton

Avertissement : Certaines modification (notamment du contenu) peuvent entrainer des erreurs de placement. Pour les régler, il faut utiliser *Button.update()*

24.3 update_all

Description Met à jour les placements du boutons et de son label
Retourne Rien
Paramètre Rien

24.4 get_size

Description Récupère la taille du bouton
Retourne <list> : Taille du bouton
Paramètre Rien

24.5 set_size

Description Définit la taille du bouton
Retourne Rien
Paramètre <list> : Taille du bouton

24.6 get_command

Description Récupère la fonction du bouton
Retourne <Function> : Fonction du widget
Paramètres Rien

24.7 set_command

Description Définit la fonction du widget
Retourne Rien
Paramètre <Function> : fonction du widget

24.8 get_id

Description Récupère l'id du widget
Retourne <int> : Id du widget
Paramètre Rien

Note : L'id vaut -1 si le widget n'a pas été ajouté à un System.

24.9 get_system

Description Récupère le système du widget
Retourne <UISystem|None> : Système du widget si il a été ajouté à un UISystem
Paramètre Rien

24.10 get_position

Description Récupère la position du widget
Retourne <list> : Position du widget
Paramètre Rien

24.11 set_position

Description Définit la position du widget
Retourne Rien
Paramètre <list> : Position du widget

CHAPITRE 25

Entry

Cette classe permet de demander du texte à l'utilisateur

25.1 Constructeur

Description Crée l'objet Entry

Paramètres

— position <list> : Position du widget
— width <int> (200) : Largeur du widget

25.2 get_text

Description Récupère le texte du widget

Retourne <string> : Texte du widget

Paramètre Rien

25.3 set_text

Description Définit le texte du widget

Retourne Rien

Paramètre text <string> : Texte du widget

25.4 get_id

Description Récupère l'id du widget

Retourne <int> : Id du widget
Paramètre Rien

Note : L'id vaut -1 si le widget n'a pas été ajouté à un System.

25.5 get_system

Description Récupère le système du widget
Retourne <UISystem|None> : Système du widget si il a été ajouté à un UISystem
Paramètre Rien

25.6 get_position

Description Récupère la position du widget
Retourne <list> : Position du widget
Paramètre Rien

25.7 set_position

Description Définit la position du widget
Retourne Rien
Paramètre position <list> : Position du widget