Pygame

Introduction :

Pygame est un module python proposant un ensemble de fonctionnalités utiles pour la création de jeux vidéos comme :

- La création d'une fenêtre graphique
- La possibilité de dessiner sur cette fenêtre (ligne, cercle ou directement une image)
- La gestion de la souris et du clavier
- La gestion du son
- beaucoup d'autres fonctionnalités à découvrir par vous-même.

http://pygame.org

Le principe de fonctionnement d'un grand nombre de jeux vidéo avec animations graphiques est le suivant :

Une boucle infinie va en permanence :

1. Dessiner l'ensemble des éléments sur la fenêtre graphique (personnages, fond, score, …)
2. Réagir aux évènements "extérieurs" : bouton de la souris, touche du clavier …
3. Calculer la logique de notre jeu : déplacement, détection de collision, intelligence artificielle ...

Ce qui peut se symboliser en python par le code suivant :

```
while True:
    dessinerSurLaFenetre()
    gererClavierEtSouris()
    logiqueDuJeu()
```

Cette boucle, va ainsi dessiner le plus rapidement possible, en déplaçant à chaque fois les éléments, et le décor de votre jeu : à la manière des dessins animés, cela provoquera une "animation graphique" simulant un déplacement fluide des éléments.

Descriptions de différents objets :

Surface	Pygame introduit la notion de 'Surface' pour **représenter une zone de dessin**. Cette zone peut-être affichée à l'écran (la fenêtre d'une application) ou seulement définie "en mémoire". *Par exemple une image chargée depuis un fichier mais non encore affichée.* Il est donc possible de ● dessiner des formes simples sur une surface (trait, cercle, arc) ● de superposer une deuxième surface sur la première.
Rect	'Rect' représente un rectangle sous la forme d'un quadruplet de valeurs : les coordonnées x et y du coin supérieur gauche ainsi que la hauteur et la largeur. $(x_{coin.sup.gauche}, y_{coin.sup.gauche}, H, L)$ Beaucoup de fonctions de pygame utilisent ou renvoient des objets 'Rect' pour représenter une zone virtuelle sur une Surface.
Font	Représente une police de caractère

Sélection de fonctions proposées par “pygame” :

Documentation complète à l’adresse : http://www.pygame.org/docs/
Une introduction en français : http://fr.wikibooks.org/wiki/Pygame

Fonctions générales :

pygame.**init**()	Initialise la bibliothèque Pygame
pygame.**quit**()	Quitte la bibliothèque Pygame
pygame.display.**set_mode**(*(largeur,hauteur)*) → *Surface*	Initialise une fenêtre de taille (largeur, hauteur). Cette fonction renvoie la *‘Surface’* correspondant à la fenêtre elle-même. C’est sur ce résultat que votre application dessinera tous les éléments.
pygame.display.**flip**()	Rafraîchit la fenêtre graphique. A appeler après avoir dessiné sur la fenêtre.
pygame.display.**set_icon**(*surface1*)	Change l’icône de la fenêtre graphique. Le paramètre donné (l’icône ‘surface1’) est une ‘Surface’ soit dessinée par vos soins, soit une simple image chargée depuis un fichier.
pygame.display.**set_caption**(*titre*)	Change le titre de la fenêtre graphique. *titre* : une chaîne de caractères

Couleurs :

module ‘Color’.

pygame.**Color**(*r, v, b*) pygame.**Color**(*rvbvalue*)	Crée un objet de type Color représentant une couleur de composante r, v et b. *r,v,b* : un nombre entier dans l’intervalle [0,255] *rvbvalue* : une chaîne de caractères représentant la couleur : “#FF0000”
uneCouleur.**r** *uneCouleur*.**v** *uneCouleur*.**b**	Renvoie la composante rouge, verte ou bleu de la couleur ‘uneCouleur’.

Dessin :

module ‘draw’.

pygame.draw.**line**(*surface1, couleur, (x1,y1), (x2,y2), width=1*)	Trace une ligne sur la Surface ‘surface1’ donnée en paramètre. *couleur* : un objet de type Color
pygame.draw.**rect**(*surface1, couleur, rect1, width=0*)	Trace un rectangle sur la Surface ‘surface1’ donné en paramètre. *couleur* : un objet de type Color *rect1* : un Rect
pygame.draw.**circle**(*surface1, couleur, (x,y), rayon, width=0*)	Trace un cercle sur la Surface ‘surface1’ donné en paramètre. *couleur* : un objet de type Color *rayon* : un nombre entier positif
pygame.draw.**ellipse**(*surface1, couleur, rect1, width=0*)	Trace une ellipse sur la Surface ‘surface1’ donnée en paramètre. L’ellipse sera inscrite dans la zone rect1 (un Rect) donné en paramètre. *couleur* : un objet de type Color *rect1* : un object de type Rect
pygame.draw.**arc**(*surface1, couleur, rect1, start_angle, stop_angle, width=1*)	Trace un arc sur la Surface ‘surface1’ donné en paramètre. L’arc sera inscrit dans la zone rect1 (un Rect) donné en paramètre. *couleur* : un objet de type Color *rect1* : un object de type Rect *start_angle,stop_angle* : un nombre à virgule
pygame.draw.**polygon**(*surface1, couleur, listeDePoints, width=0*)	Trace un polygone sur la Surface ‘surface1’ donné en paramètre. *couleur* : un objet de type Color
pygame.draw.**lines**(*surface1, couleur, ferme, listeDePoints, width=1*)	Trace une suite de lignes sur la Surface ‘surface1’ donnée en paramètre.
pygame.draw.**aaline**(*surface1, couleur, (x1,y1), endpos, blend=1*)	Identique à la fonction line(..) mais applique un traitement de lissage. Note : le préfix ‘aa’ signifie Anti-Aliasing, ou anticrénelage en français (voir Wikipédia).
pygame.draw.**aalines**(*surface1, couleur, closed, listeDePoints, blend=1*)	Identique à la fonction lines(..) mais applique un traitement de lissage. Note : le préfixe ‘aa’ signifie Anti-Aliasing, ou anticrénelage en français (voir Wikipédia).

Image :

module ‘image’.

pygame.image.**load**(*nomDeFichier*) → Surface	Charge en mémoire une image depuis un fichier. Renvoie la ‘Surface’ (en mémoire) correspondante.
pygame.image.**save**(*surface, nomDeFichier*)	Enregistre la Surface ‘surface’ dans le fichier ‘nomDeFichier’. Le format d’enregistrement dépend de l’extension du nom de fichier donné : nomDeFicher.jpg → Format JPEG

	nomDeFicher.png → Format PNG

Évènements souris et clavier :

Clavier :

pygame.key.**get_pressed**() → *bools*	Renvoie un tableau de booléens (valeurs vraie/faux) représentant l'état des touches du clavier (appuyée ou non).
	http://www.pygame.org/docs/ref/key.html#pygame.key.get_mods
	Exemple:
	`touches = pygame.key.get_pressed()`
	`if touches[pygame.K_SPACE] == True:`
	`  print("touche espace enfoncé")`
	`if touches[pygame.K_RIGHT] == True:`
	`  print("touche droite enfoncé")`

Souris :

pygame.mouse.**get_pressed**() → *(button1, button2, button3)*	Renvoie une liste des états des 3 boutons de la souris.
pygame.mouse.**get_pos**() → *(x, y)*	Renvoie la position de la souris
pygame.mouse.**get_rel**() → *(x, y)*	Renvoie le déplacement de la souris : le nombre de pixels x et y dont la souris s'est déplacée depuis le précédent appel à cette fonction.
pygame.mouse.**set_pos**(*[x, y]*)	Positionne le curseur de la souris en (x;y)
pygame.mouse.**set_visible**(*bool*)	Cache ou rend visible le curseur de la souris. *bool* : True ou False

Rectangle :

module 'Rect'.

pygame.**Rect**(*x, y, largeur, hauteur*) → *Rect* pygame.**Rect**(*(x, y), (largeur, hauteur)*) → *Rect*	Crée un nouveau rectangle.
unRectangle.**contains**(*rect2*) → *bool*	Teste si le rectangle 'rect2' donné en paramètre est à l'intérieur du Rect 'unRectangle'. *rect2* : un objet de type Rect
unRectangle.**collidepoint**(*x, y*) → *bool* *unRectangle*.**collidepoint**(*(x,y)*) → *bool*	Teste si le point de coordonnées *(x,y)* données en paramètre est à l'intérieur du Rect 'unRectangle'.
unRectangle.**colliderect**(*rect2*) → *bool*	Teste si le rectangle 'rect2' donné en paramètre superpose même partiellement le Rect 'unRectangle'. *rect2* : un objet de type Rect

Surface :

module ‘Surface’.

pygame.**Surface**(*(largeur, hauteur)*) → *Surface*	Crée une nouvelle *Surface* en mémoire de taille ‘*largeurXhauteur’*.
uneSurface.**blit**(*surface2, (px,py)*) → *Rect*	Dessine la Surface ‘surface2’ à la position (px,py) sur la Surface ‘uneSurface’.
uneSurface.**copy**() → *Surface*	Renvoie une copie de la Surface ‘uneSurface’.
uneSurface.**fill**(*c1, rect=None*) → *Rect*	Colorie la Surface ‘uneSurface’ avec la couleur c1 donnée en paramètre. On peut passer un objet de type Rect en deuxième paramètre pour délimiter la zone à colorier.
uneSurface.**scroll**(*dx=0, dy=0*)	Décale la Surface ‘uneSurface’ de ‘dx’ et ‘dy’ pixels vers la droite et vers le bas.
uneSurface.**set_alpha**(*valeur*) *uneSurface*.**set_alpha**(None)	Définit le degré de transparence pour la Surface ‘uneSurface’. (valeurs dans l’intervalle [0,255]) Note : le degré de transparence est nommé ‘alpha value’.
uneSurface.**get_at**(*(x, y)*) → Color	Retourne la couleur du pixel à la position (x,y).
uneSurface.**set_at**(*(x, y), couleur*)	Change la couleur du pixel à la position (x,y). *couleur* : un objet de type Color
uneSurface.**get_size**() → *(width, height)*	Renvoie la largeur et la hauteur de la Surface ‘uneSurface’ sous la forme d’un couple (l,h)
uneSurface.**get_width**() → *width*	Renvoie la largeur de la Surface ‘uneSurface’.
uneSurface.**get_height**() → *height*	Renvoie la hauteur de la Surface ‘uneSurface’.

Transformation :

module ‘transform’.

pygame.transform.**flip**(*surface1, xbool, ybool*) → *Surface*	Renvoie une nouvelle Surface correspondant à la Surface ‘surface1’ passée en paramètre retournée suivant l’axe des x et/ou des y. *xbool, ybool* : True ou False
pygame.transform.**scale**(*surface1, (largeur, hauteur)*) → *Surface*	Renvoie une nouvelle Surface correspondant à la Surface ‘surface1’ passée en paramètre agrandie ou rétréci à la taille (largeur, hauteur) passée en paramètre.
pygame.transform.**rotate**(*surface1, angle*) → *Surface*	Renvoie une nouvelle Surface correspondant à la Surface ‘surface1’ passée en paramètre tournée du paramètre ‘angle’. *angle* : un nombre à virgule
pygame.transform.**rotozoom**(*surface1, angle, facteur*) → *Surface*	Renvoie une nouvelle Surface correspondant à la Surface ‘surface1’ passée en paramètre tournée de ‘angle’ et agrandie par ‘facteur’. *facteur* : un nombre à virgule *angle* : un nombre à virgule
pygame.transform.**smoothscale**(*surface1, (largeur, hauteur)*) → *Surface*	Identique à la fonction scale(..), mais applique un traitement pour “lisser” le résultat.

Texte :

module ‘font’.

pygame.font.**SysFont**(*nom, taille, gras=False, italic=False*) → Font	Renvoie une nouvelle police de caractère. *nom* : une chaîne de caractères *taille :* un nombre entier *gras, italic* : True ou False
unePoliceDeCaractere.**render**(*texte, antialias, couleur, background=None*) → Surface	Renvoie une nouvelle Surface ou le paramètre ‘texte’ est inscrit en utilisant la police ‘*unePoliceDeCaractere*’. Note : le paramètre ‘antialias’ (True/False) signifie Anti-Aliasing, ou anticrénelage en français (voir Wikipédia). *texte* : une chaîne de caractères antialias : True ou False couleur : un objet de type Color

Exemple :

```
arial24 = pygame.font.SysFont("arial",24)
surfaceTexte = arial24.render("mon texte",True,pygame.Color(0,255,255))
```

Horloge :

module ‘time’.

pygame.time.**get_ticks**() → *milliseconds*	Renvoie le nombre de millisecondes depuis l’appel à pygame.init().
pygame.time.**wait**(*nMilliseconds*)	Interrompt le programme pendant ‘nMilliseconds’.
pygame.time.Clock() → Clock uneHorloge.**tick**(*framerate=0*)	Crée une nouvelle horloge. tick(..) permet de s’assurer d’attendre le temps nécessaire pour obtenir le rafraichissement par seconde souhaité.