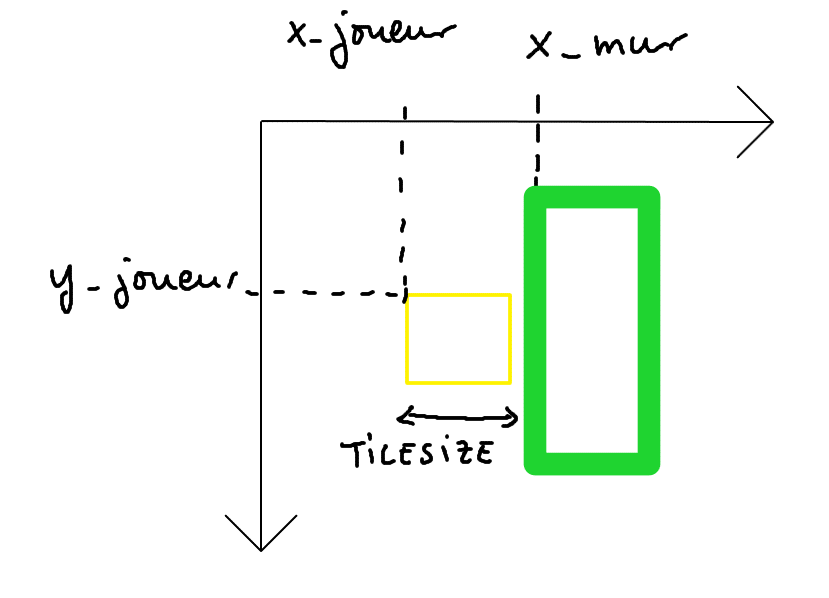
Notre objectif est de corriger le bug qui fait que le joueur ne peut pas toujours « coller » au mur : quand il se déplace, il reste parfois quelques pixels entre lui et le mur. Regardons pourquoi !

Le joueur est ici symbolisé par le carré jaune et la position de son coin haut-gauche est (x\_joueur, y\_joueur).

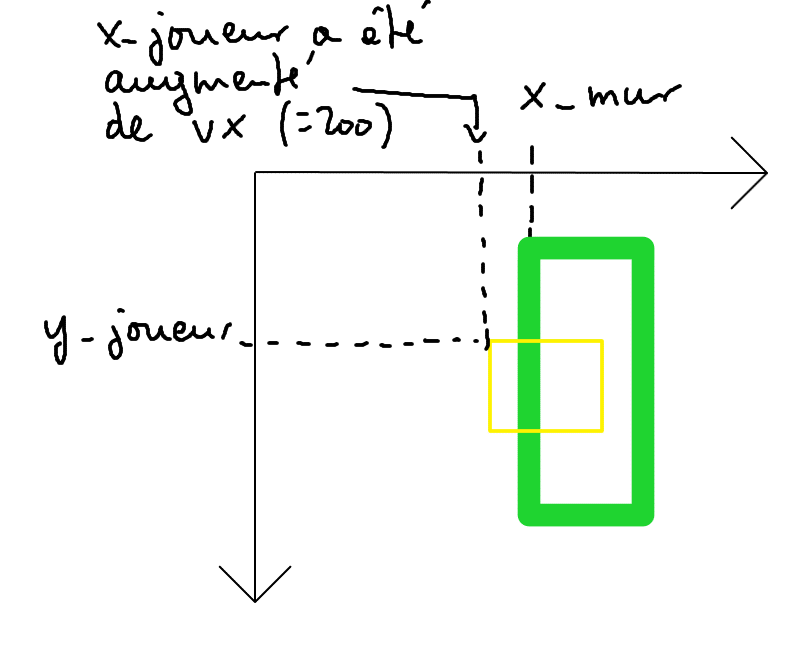
Le mur est symbolisé par le rectangle vert et son abscisse est x\_mur.



Le joueur veut se déplacer vers la droite et, comme il ne se déplace pas pixel par pixel (ce serait trop lent !), il possède une vitesse horizontale vx qui vaut dans notre code PLAYER\_SPEED = 200.

Si le joueur se déplace vers la droite, la fonction update() de la classe Player applique l’algorithme suivant :

1. Elle augmente la position x\_joueur de la valeur vx, soit PLAYER\_SPEED, ce qui donne le dessin suivant :



1. La fonction update() appelle la fonction pg.sprite.spritecollideany() pour tester si le joueur est rentré en collision avec un mur. Dans notre cas, c’est vrai ! Alors la fonction update() annule le déplacement du joueur en le remettant à sa position précédente ; dans notre cas, elle soustrait la valeur de vx à la position x\_joueur.

En conclusion : avec de la malchance, le joueur ne peut pas s’approcher parfaitement des murs ;-)

Nous allons corriger ce problème…

Tout d’abord, remarquez que le code de la fonction move() dans la classe Player ne nous sert plus à rien : vous pouvez l’effacer…

Le problème vient de l’appel à la fonction pg.sprite.spritecollideany()(dans la fonction update()) qui n’est pas assez précis. Nous allons donc remplacer cette fonction par une autre fonction que nous nommerons collide\_with\_walls(). Cette fonction prendra le joueur en paramètre ainsi que le sens du déplacement.

Le code de la fonction update() devient :

def update(self):

self.get\_keys()

self.x += self.vx \* self.game.dt

self.y += self.vy \* self.game.dt

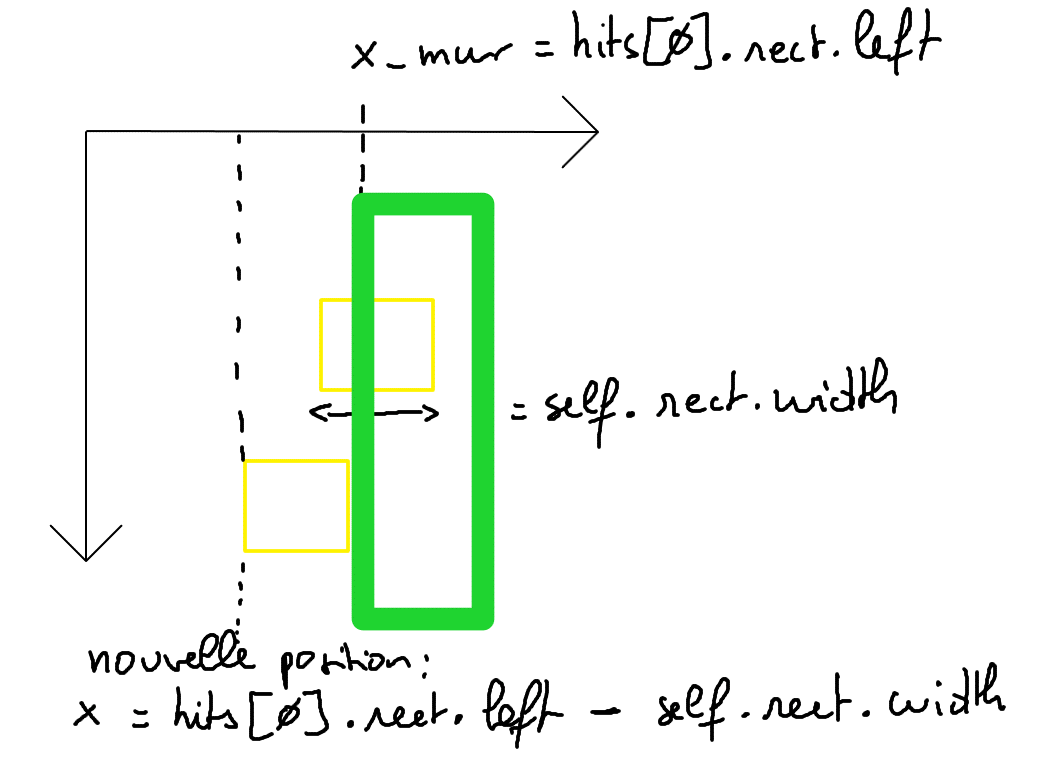
self.rect.x = self.x

self.collide\_with\_walls('x') # vérifie les collisions Horiz

self.rect.y = self.y

self.collide\_with\_walls('y') # vérifie les collisions Vert

La fonction collide\_with\_walls()cherche si le joueur a provoqué une collision horizontalement et si c’est le cas, on recule le joueur non pas à sa position d’origine mais à la position qui le fera “coller” au mur. Si le joueur se déplaçait vers la droite, la nouvelle position est la valeur de la position du mur moins la largeur du joueur, comme le montre ce dessin :



Si le joueur se déplace à gauche, sa nouvelle position serait la valeur du bord « droit » du mur, soit hits[0].rect.right !

Le code de la fonction collide\_with\_walls() est le suivant :

def collide\_with\_walls(self, dir):

if dir == 'x':

hits = pg.sprite.spritecollide(self, self.game.walls, False)

if hits:

if self.vx > 0: # on va vers la droite

self.x = hits[0].rect.left - self.rect.width

if self.vx < 0: # on va vers la gauche

self.x = hits[0].rect.right

self.vx = 0

self.rect.x = self.x

if dir == 'y':

pass

Testez ce code et vérifiez que nous avons régler le problème des déplacements horizontaux. Mais, comme vous le voyez le code de la fonction est incomplet : je vous demande de le compléter afin de corriger aussi le problème des déplacements verticaux. Les positions des bords d’un mur s’appellent rect.top et rect.bottom et la hauteur du joueur est self.rect.height..