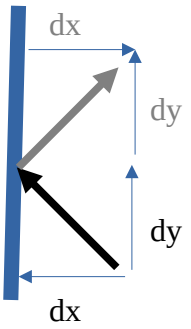
Casse-brique 2ème partie

On suppose la 1ère partie de casse-brique terminée et validée.

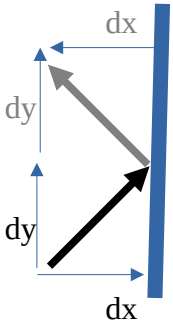
- la balle a un rayon de 8 pixels ;
- les noms des variables, objets et fonctions ont été francisés ;
- et bien sur et surtout le programme casse-brique_0005.py est bien compris

1. Gérer les rebonds de la balle : casse-brique_0006.py

La vitesse de déplacement de la balle possède une composante horizontale dx et une composante verticale dy. Lors des rebonds, la direction dans laquelle se déplace la balle est modifiée.
Lorsque la balle heurte le mur gauche la composante horizontale de la vitesse est inversée.
Lorsque la balle heurte le mur droit la composante horizontale de la vitesse est inversée.
Lorsque la balle heurte le mur en haut la composante verticale de la vitesse est inversée.

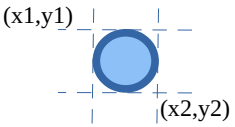


Lors d'un rebond à droite ou à gauche :
- la composante horizontale de la vitesse change de sens ;
dx = - dx # nouvelle valeur de dx = - dx
- la composante verticale de la vitesse n'est pas modifiée ;
dy = dy



Il faut appliquer ces modifications au vecteur vitesse si la balle heurte le mur gauche (ou le mur droit). Pour récupérer d la position de la balle nous pourrons utiliser la méthode coords qui s'applique aux objets de la classe tk.Canvas ; donc à l'objet balle. La documentation de la méthode nous indique que l'on peut récupérer les coordonnées de deux coins opposés du rectangle englobant l'objet.

```
x1, y1, x2, y2 = zone_de_jeu.coords(balle)
```



Il ne nous reste plus qu'à traduire en langage python les conditions de modification du vecteur vitesse :

```
# si la balle a touché le mur gauche ou si la balle a touché le mur droit
if     ……  < 0  or  ……. > LARGEUR :
      dx = -dx
# si la balle a touché le mur haut ou si la balle a touché le mur bas
if     ……  < 0  or  ……. > HAUTEUR :
      dy = -dy
```

Où faut-il ajouter ces lignes pour que le programme simule des rebonds sur les murs ?