Les Cahiers d'Exercices en Programmation : Le langage Python

Apprendre à coder en t'amusant

Apprenez et entraînez vos acquis

- De très nombreux exercices à réaliser par vous-même

AVANT-PROPOS

Ce livre est un cahier d'exercices : il vous propose des énoncés d'exercices et leurs corrigés. Vous allez apprendre le logiciel en vous entrainant à travers des exercices regroupés par thème.

Chaque énoncé vous présente l'exercice à réaliser. Vous trouverez à la fin du cahier le corrigé de chaque exercice. Certaines explications peuvent-être présentes.

METHODOLOGIE

Lors de la réalisation des exercices, vous pourrez remédier à certain problème à l'aide des corrections à la fin du cahier.

Après avoir réalisé tous les exercices de chaque chapitre vous allez pouvoir vérifier les compétences acquises à l'aide du tableau des objectifs.

Celui-ci sert à la cotation du professeur (grille d'évaluation).

Des **légendes ou recommandations** peuvent être présentes dans certains exercices. Celles-ci vous aideront dans vos recherches. Elles ne doivent pas être reproduites dans votre travail.

Chaque point de matière acquis dans un exercice peut être utilisé dans des exercices suivants sans explication.

Table des matières

Chapitre 1 : Qu'est-ce que Kidscod.in ?

Kidscod.in est une méthode en ligne pour apprendre aux enfants de 7 à 77 ans le fonctionnement de la programmation.

- L'apprentissage s'y fait à l'aide d'un langage de programmation visuel, reposant sur des blocs logiques à imbriquer ;
- Chacun apprend à son rythme à l'aide d'exercices guidés et libres
- Kidscod est le tremplin pour s'orienter par après vers des langages de programmation moderne tels que Python ou Javascript.

Vous allez aborder le langage Python d'une façon novatrice : chaque brique de code déposée sur l'espace de travail génère le code Python correspondant, afin de favoriser la découverte et de la syntaxe et de la structure du code Python, de façon visuelle et ludique.

Chapitre 2 : Apprendre à faire bouger un tank

Objectif(s) des notions et des exercices de ce chapitre.

À la fin de ce chapitre, l'élève sera capable de :

Création d'un compte sur Kidscod.in
Déclencher une action suite à l'appui d'une touche
Tester la position du Sprite pour vérifier s'il a « le droit » de se déplacer ou pas
Effectuer un test conditionnel
Mettre en parallèle ces actions avec du code Python

Ce premier exercice va te permettre de ta familiariser avec ton environnement de travail en réalisant les premières briques de ton jeu vidéo.

1 : Création de ton compte

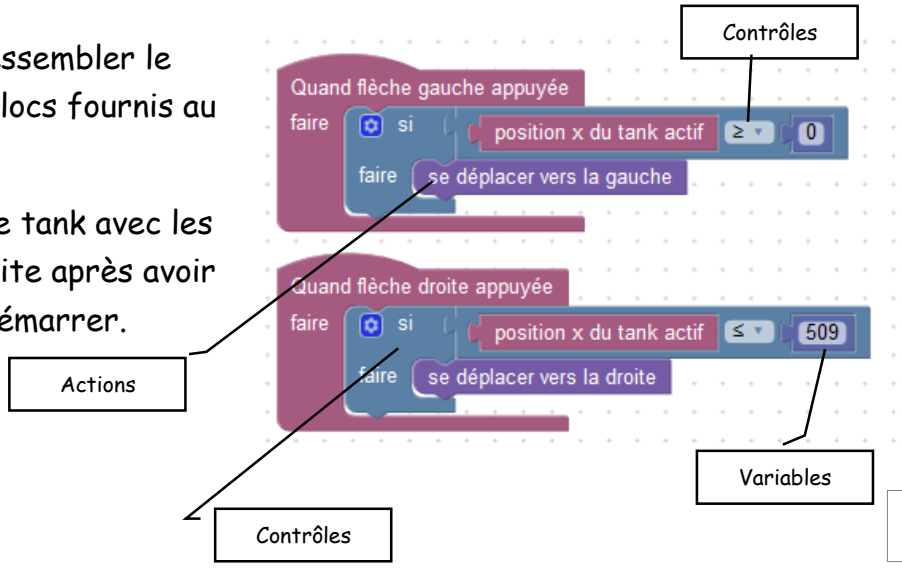
Adresse : http://www.kidscod.in/app/tank

Crée ton compte et valide-le en cliquant sur le mail que tu recevras.

2 : Donner un mouvement à un tank

Voici à quoi devrait ressembler le code en utilisant les blocs fournis au chapitre 1 du site.

Essayer de déplacer le tank avec les flèches gauche et droite après avoir cliqué sur le bouton Démarrer.



Il est temps de jeter un œil au code Python généré pour bien comprendre comment tout ce que tu as fait aurait pu être codé de façon textuelle :

```
for event in pygame.event.get():
  if event.type == K_LEFT:
    if getActiveTankX() >= 0:
      avancerGauche()

for event in pygame.event.get():
  if event.type == K_RIGHT:
    if getActiveTankX() <= 509:
      avancerDroite()
```

Ne t'inquiète pas, tu peux regarder ce code mais la programmation se réalisera à partir du niveau 2 de ce cahier d'apprentissage.

Chapitre 3 : Comment tirer un obus

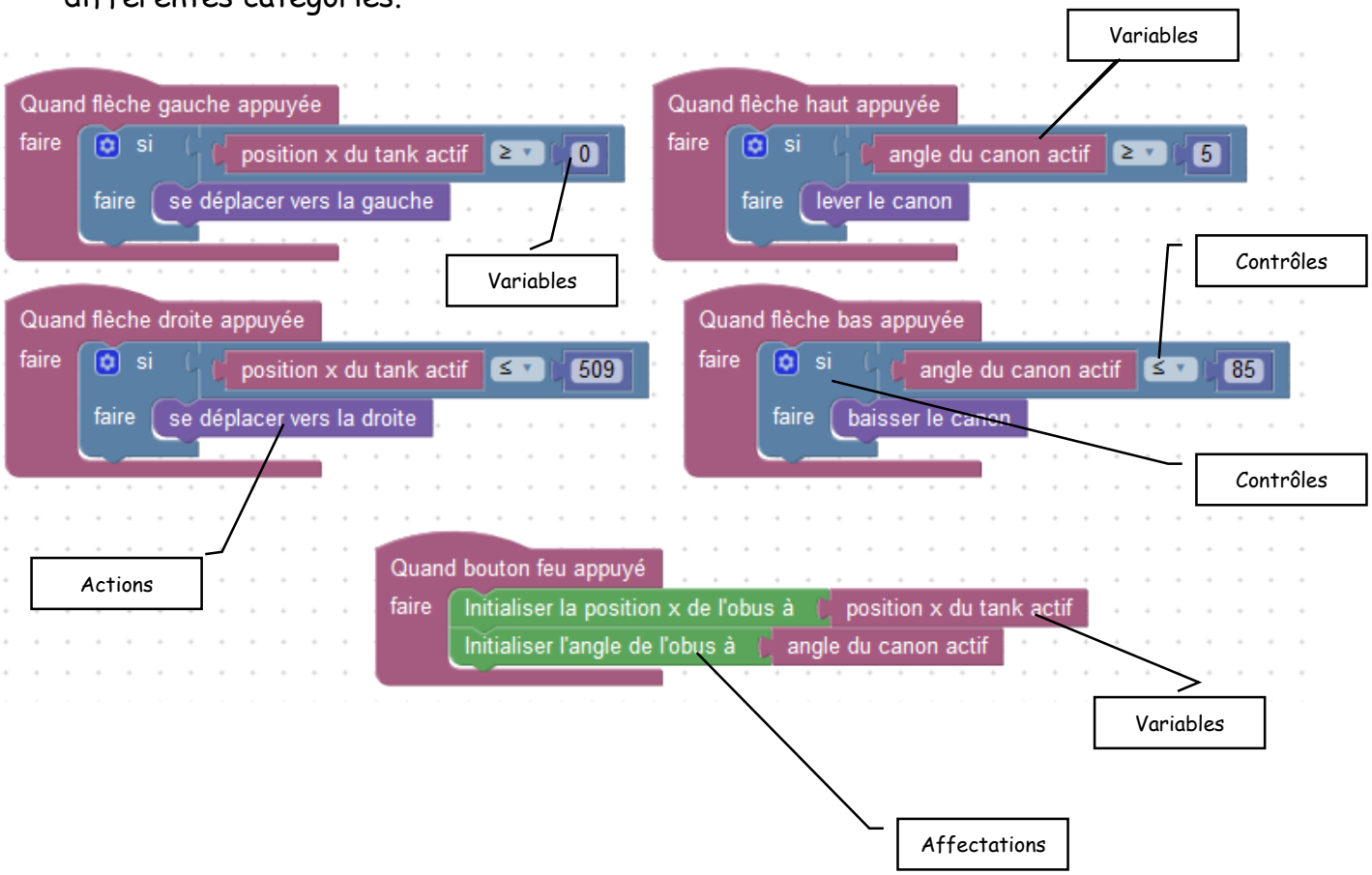
Objectif(s) des notions et des exercices de ce chapitre.

À la fin de ce chapitre, l'élève sera capable de :

Affectation de valeurs
Apparition et disparition de Sprites
Orienter les canons
Comprendre et utiliser la fonction Si … Faire

Tirer un obus et gestion du déplacement du tank

Voici à quoi devrait ressembler le code. Trouver donc les blocs dans les différentes catégories.

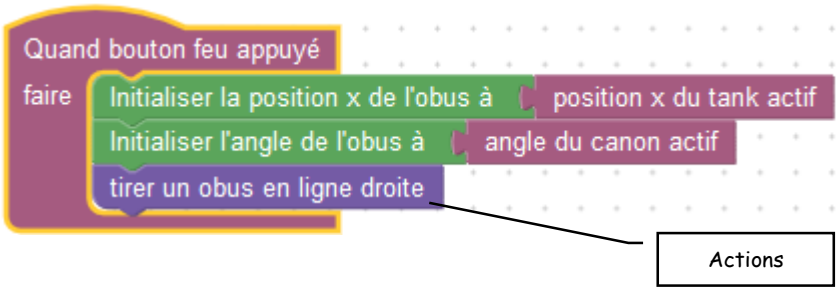


Il est temps de jeter un œil au code Python généré pour bien comprendre comment tout ce que tu as fait aurait pu être codé de façon textuelle :

```
for event in pygame.event.get():
  if event.type == K_LEFT:
    if getActiveTankX() >= 0:
      avancerGauche()

for event in pygame.event.get():
  if event.type == K_UP:
    if getActiveBarrelsRotation() >= 5:
      leverCanon()

for event in pygame.event.get():
  if event.type == K_RIGHT:
    if getActiveTankX() <= 509:
      avancerDroite()

for event in pygame.event.get():
  if event.type == K_DOWN:
    if getActiveBarrelsRotation() <= 85:
      baisserCanon()

for event in pygame.event.get():
  if event.type == K_SPACE:
    setWeapon1X((getActiveTankX()))
    setWeapon1Rotation((getActiveBarrelsRotation()))
```

Et l'obus dans tout ça ?

Modifier le bloc **Quand bouton feu appuyé**.



Code Python correspondant :

```
for event in pygame.event.get():
  if event.type == K_SPACE:
    setWeapon1X((getActiveTankX()))
    setWeapon1Rotation((getActiveBarrelsRotation()))
    fireWeapon()
```

Chapitre 4 : Simuler les lois de la physique

Objectif(s) des notions et des exercices de ce chapitre.

À la fin de ce chapitre, l'élève sera capable de :

Création d'un moteur physique
Influence de la gravité et des forces externes (comme le vent) dans un moteur physique
Jouer avec la gravité et le vent sur le déplacement d'un objet
Initialiser des variables jouant sur la gravité et le vent

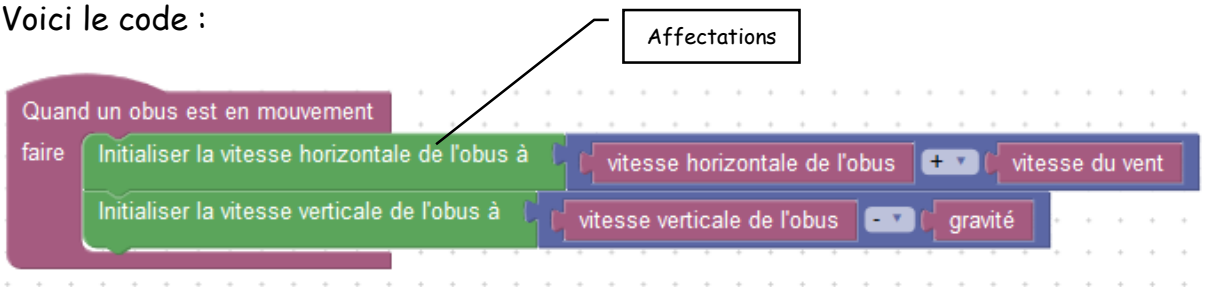
Les lois de la physique

Ajout par rapport au code précédent : le moteur physique (la gravité).

Ordre de grandeur des gravités de plusieurs astres rapportées à celle de la Terre

Planète	Ordre de grandeur de la gravité
Lune	égale à 0,1 fois celle de la Terre
Mercure	égale à 0,4 fois celle de la Terre
Vénus	égale à 0,9 fois celle de la Terre
Jupiter	égale à 2,5 fois celle de la Terre

Voici le code :



Code Python correspondant aux lois de la physique :

```
for event in pygame.event.get():
  if event.type == "ObusMoving":
    setWeapon1VX((getWeapon1VX() + getWindVX()))
    setWeapon1VY((getWeapon1VY() - getGravity()))
```

Chapitre 5 : Collisions et explosions

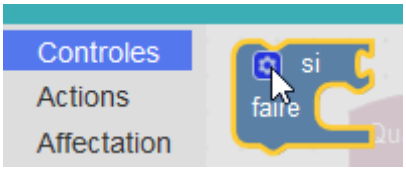
Objectif(s) des notions et des exercices de ce chapitre.

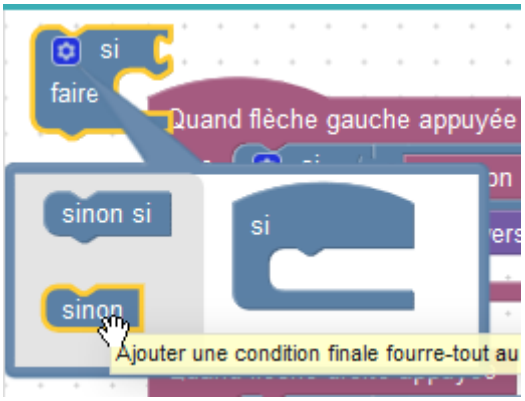
À la fin de ce chapitre, l'élève sera capable de :

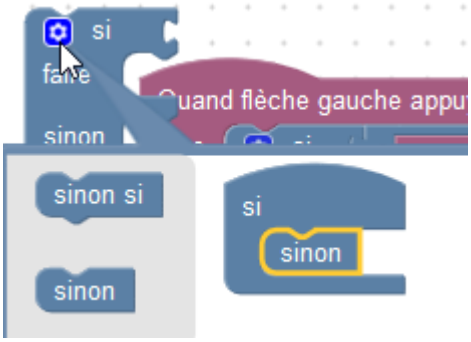
Création d'un moteur physique
Influence de la gravité et des forces externes (comme le vent) dans un moteur physique
Jouer avec la gravité et le vent sur le déplacement d'un objet
Créer et initialiser des variables jouant sur la gravité et le vent
Comprendre et utiliser la fonction Si … Retour
Comprendre et utiliser la fonction Si …. Faire ….. Sinon

Collision quand un obus touche un tank

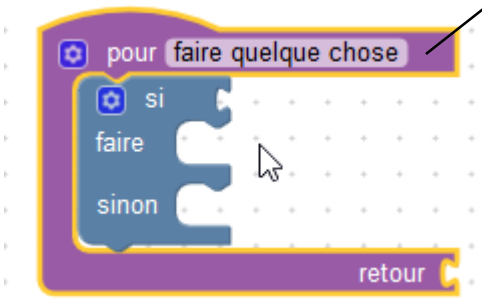
Création d'une fonction :



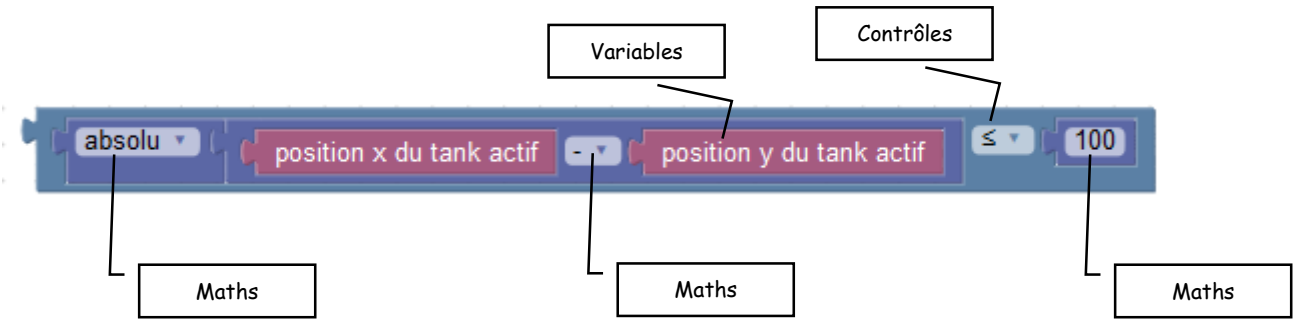




Fonctions

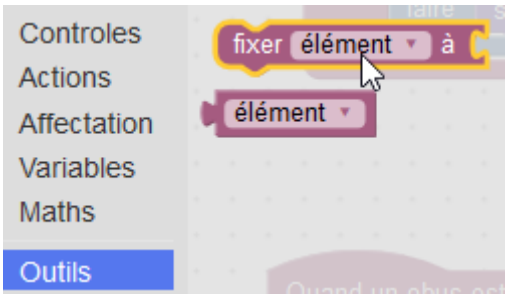


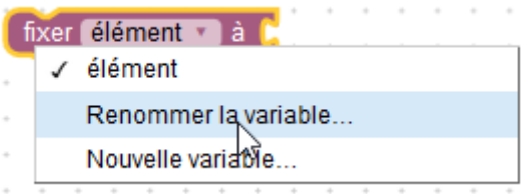
Création de la condition :



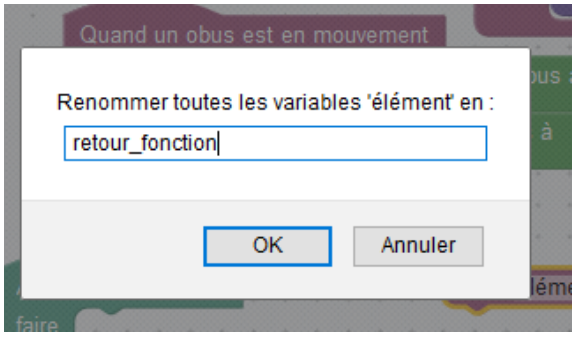
On insère la condition dans la fonction :

Création d'une nouvelle variable :

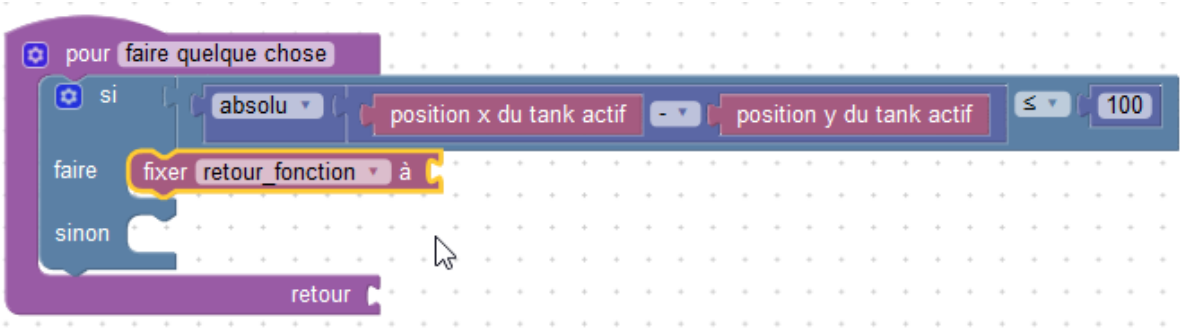




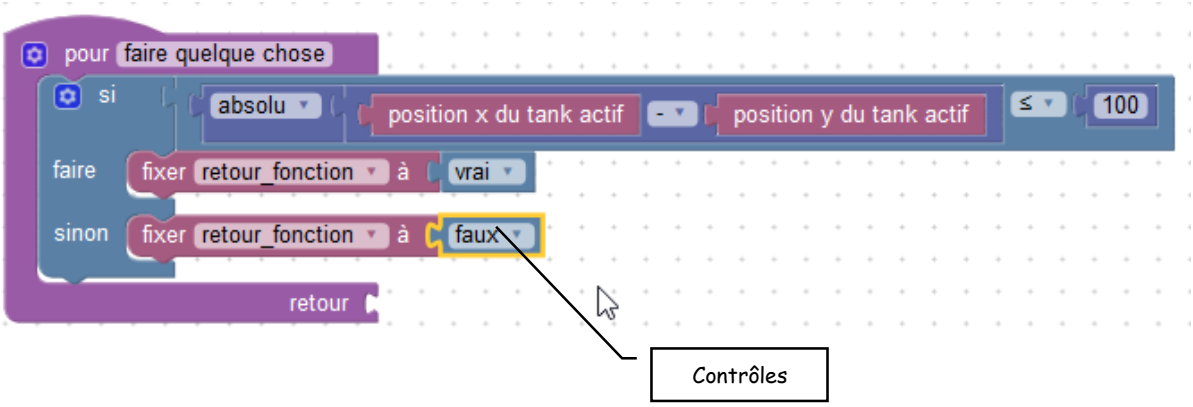
Nouveau nom de la variable : retour_fonction



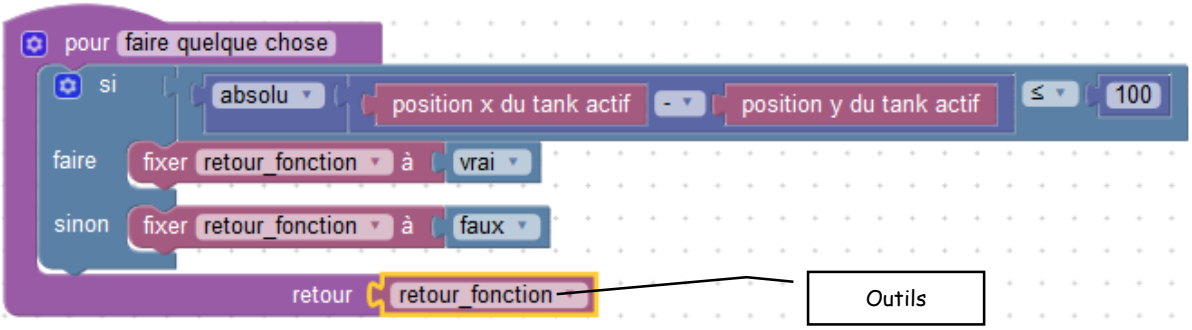
Fixer le bloc :



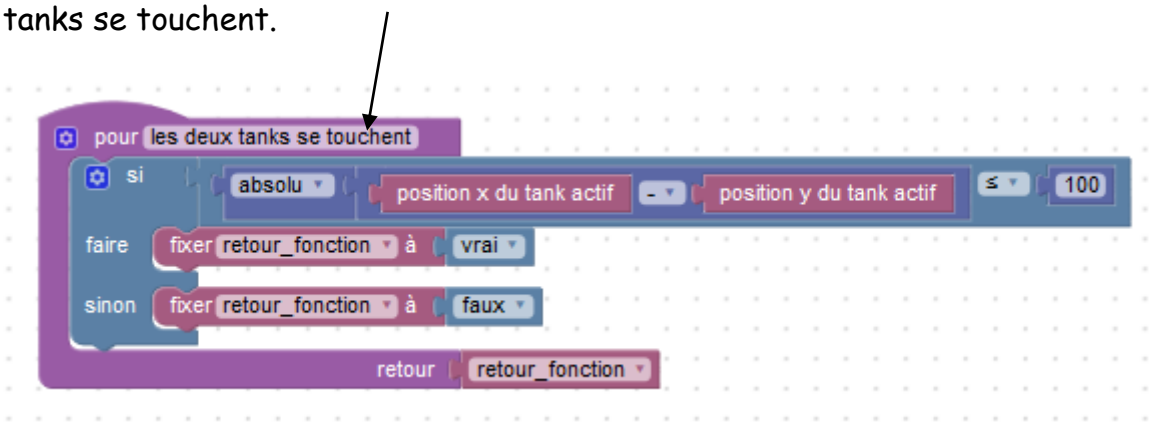
Initialiser une variable à Vrai ou à faux en cliquant sur le menu déroulant du même bloc



Retourner la variable retour_fonction. Tu as réalisé le code final de la détection entre tanks. BRAVO.



Et le pour faire quelque chose ne doit-il pas être défini ? C'est quand les deux tanks se touchent.

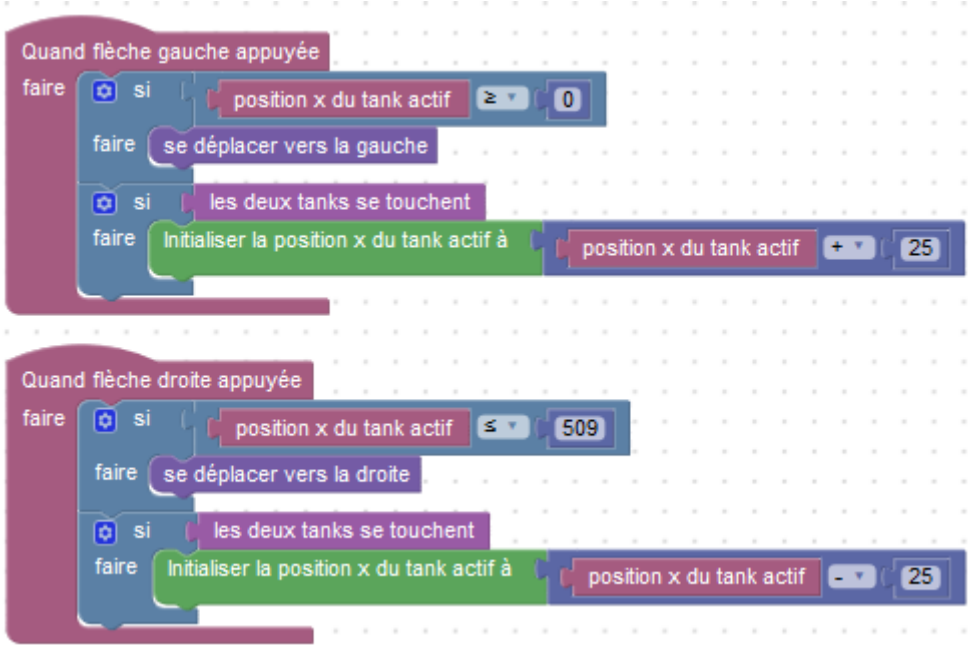


La fonction est maintenant ici définie et utilisable.

La fonction ne sert pour l'instant à rien, car elle n'est jamais utilisée.

Le test de collision sera ajouté aux blocs : quand flèche gauche appuyée et quand flèche droite appuyée.

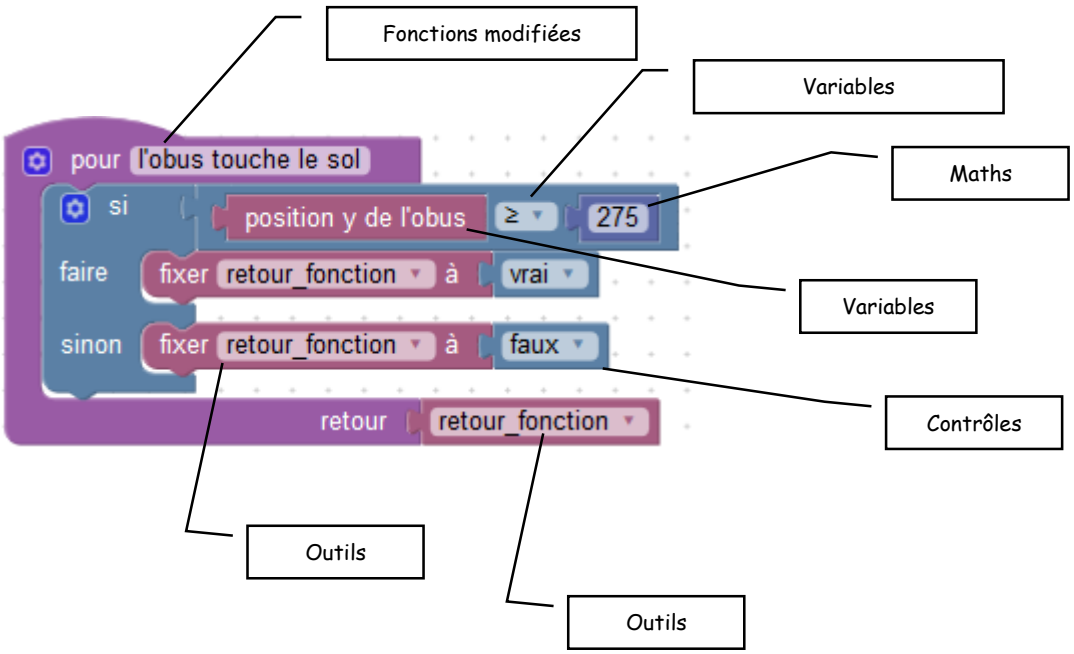
Quand une collision se réalise le tank doit rebondir brutalement en arrière (modification de la position x).



Et le code python de tout ce travail :

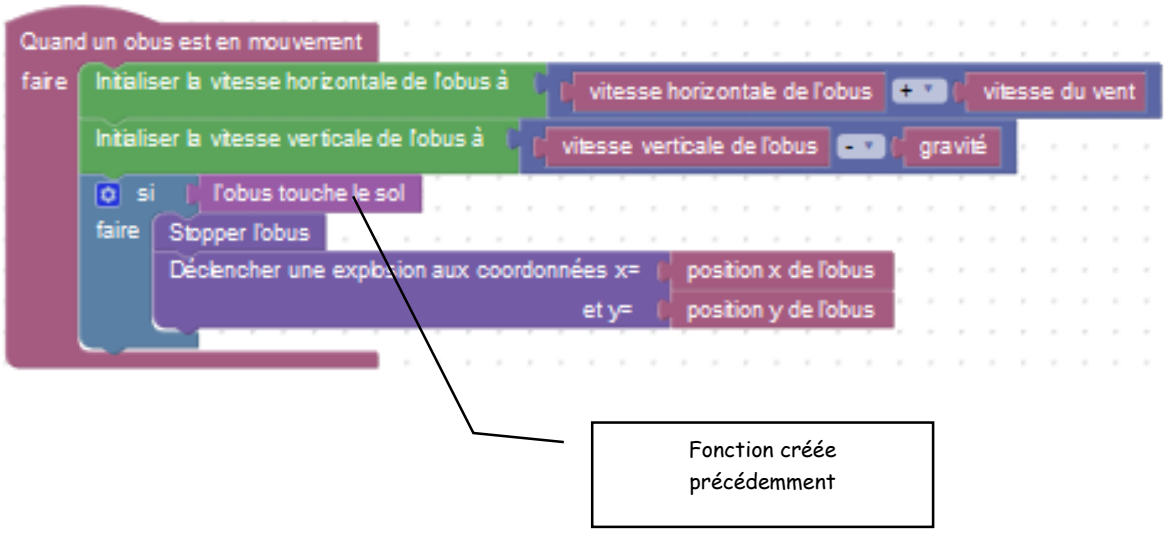
```
import math

retour_fonction = None

def les_deux_tanks_se_touchent():
  global retour_fonction
  if math.fabs(getActiveTankX() - getActiveTankY()) <= 100:
    retour_fonction = True
  else:
    retour_fonction = False
  return retour_fonction


for event in pygame.event.get():
  if event.type == K_LEFT:
    if getActiveTankX() >= 0:
      avancerGauche()
    if les_deux_tanks_se_touchent():
      setActiveTankX((getActiveTankX() + 25))

for event in pygame.event.get():
  if event.type == K_DOWN:
    if getActiveBarrelsRotation() <= 85:
      baisserCanon()

for event in pygame.event.get():
  if event.type == K_RIGHT:
    if getActiveTankX() <= 509:
      avancerDroite()
    if les_deux_tanks_se_touchent():
      setActiveTankX((getActiveTankX() - 25))

for event in pygame.event.get():
  if event.type == K_SPACE:
    setWeapon1X((getActiveTankX()))
    setWeapon1Rotation((getActiveBarrelsRotation()))
    fireWeapon()

for event in pygame.event.get():
  if event.type == K_UP:
    if getActiveBarrelsRotation() >= 5:
      leverCanon()

for event in pygame.event.get():
  if event.type == "Start":

for event in pygame.event.get():
  if event.type == "ObusMoving":
    setWeapon1VX((getWeapon1VX() + getWindVX()))
    setWeapon1VY((getWeapon1VY() - getGravity()))

for event in pygame.event.get():
  if event.type == "CompteurVautZero":

for event in pygame.event.get():
  if event.type == K_ENTER:
```

Les obus n'aiment pas tomber par terre sans exploser



Enfin une explosion

Essayer de trouver les différents blocs par vous-même.



Voici un petit récapitulons :

1. On utilise une bibliothèque externe, appelée math (pour le calcul de la valeur absolue) ;
2. Définition de la variable en tant que variable globale, c'est-à-dire un utilisable partout dans le code ;
3. Définition de notre fonction de détection de collision avec le sol ;
4. Utilisation de la variable global ;
5. Condition (position y de l'obus plus grande que la position y du sol, donc l'obus est en dessous du sol à cause de l'orientation de l'axe des y);
6. Initialisation de la variable retour à vrai ;
7. Initialisation de la variable retour à faux ;
8. Retour de la variable, fin de la fonction ;
9. Définition de notre fonction de détection entre tanks ;
10. Utilisation de la variable globale ;
11. Condition (la valeur absolue de la différence des positions x des deux tanks est inférieure à 100) ;
12. Initialisation de la variable retour à vrai ;
13. Initialisation de la variable retour à faux ;
14. Retour e la variable, fin de la fonction ;
15. Test de la détection e la collision avec le sol ;
16. Stopper la progression de l'obus ;
17. Remplacer le Sprite de l'obus par une animation d'explosion aux coordonnées x et y de l'obus au moment de l'impact.

Et maintenant ?

Le prochain chapitre va te permettre de finaliser ton jeu vidéo, en y apportant les derniers détails importants : mise en place d'un compteur de temps, changement automatique du tour de jeu, changement aléatoire du vent à chaque tour de jeu, et enfin, détection de la fin du jeu.

Voici le code :

```
import math

retour_fonction = None

def l_obus_touche_le_sol():
  global retour_fonction
  if getWeapon1Y() >= 275:
    retour_fonction = True
  else:
    retour_fonction = False
  return retour_fonction

def les_deux_tanks_se_touchent():
  global retour_fonction
  if math.fabs(getActiveTankX() - getActiveTankY()) <= 100:
    retour_fonction = True
  else:
    retour_fonction = False
  return retour_fonction


for event in pygame.event.get():
  if event.type == K_LEFT:
    if getActiveTankX() >= 0:
      avancerGauche()
    if les_deux_tanks_se_touchent():
      setActiveTankX((getActiveTankX() + 25))

for event in pygame.event.get():
  if event.type == K_UP:
    if getActiveBarrelsRotation() >= 5:
      leverCanon()

for event in pygame.event.get():
  if event.type == K_DOWN:
    if getActiveBarrelsRotation() <= 85:
      baisserCanon()

for event in pygame.event.get():
  if event.type == K_RIGHT:
    if getActiveTankX() <= 509:
      avancerDroite()
    if les_deux_tanks_se_touchent():
      setActiveTankX((getActiveTankX() - 25))

for event in pygame.event.get():
  if event.type == K_SPACE:
    setWeapon1X((getActiveTankX()))
    setWeapon1Rotation((getActiveBarrelsRotation()))
    fireWeapon()

for event in pygame.event.get():
  if event.type == "ObusMoving":
    setWeapon1VX((getWeapon1VX() + getWindVX()))
    setWeapon1VY((getWeapon1VY() - getGravity()))
    if l_obus_touche_le_sol():
      stopperObus()
      doExplode((getWeapon1X()),(getWeapon1Y()))
```

Chapitre 6 : Finalisation du jeu

Objectif(s) des notions et des exercices de ce chapitre.

À la fin de ce chapitre, l'élève sera capable de :

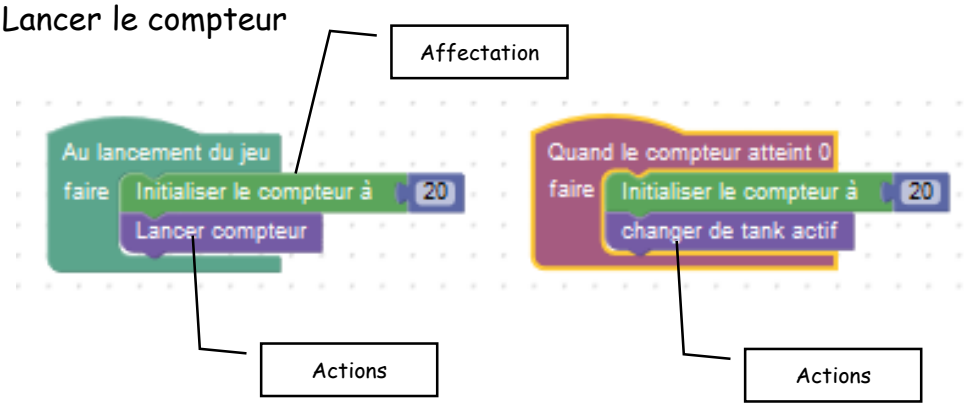
Comprendre la notion de boucles
Initialiser et gérer des nombres aléatoires
Gérer le décompte d'un nombre
Utiliser la fonction Si … sinon
Utiliser la fonction Si … faire … sinon

Apporter la touche finale

- Mettons en place des tours de jeu. Ils pourront donc s'affronter l'un à la suite de l'autre
- Et ne pas oublier de réinitialiser le compteur à 10 lorsqu'il atteint zéro
- Changement de la météo à chaque tour
- Détection de collision entre un obus et un tank
- Informer les joueurs quand la partie est terminée. Message qui s'affiche quand un tank a été détruit.

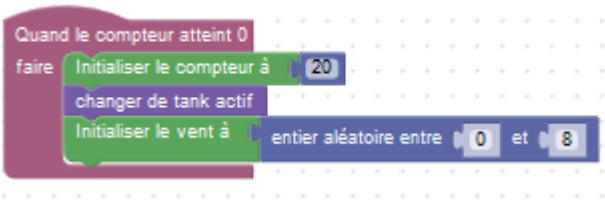
Mise en place du compteur et des tours de jeu

- Il faut choisir la valeur de départ du compteur. La valeur est en secondes. Par exemple si le choix est 20, chaque partie durera 20 secondes.
- Lancer le compteur



Attribuer une valeur au hasard pour le vent

Blocs situés dans la catégorie Maths

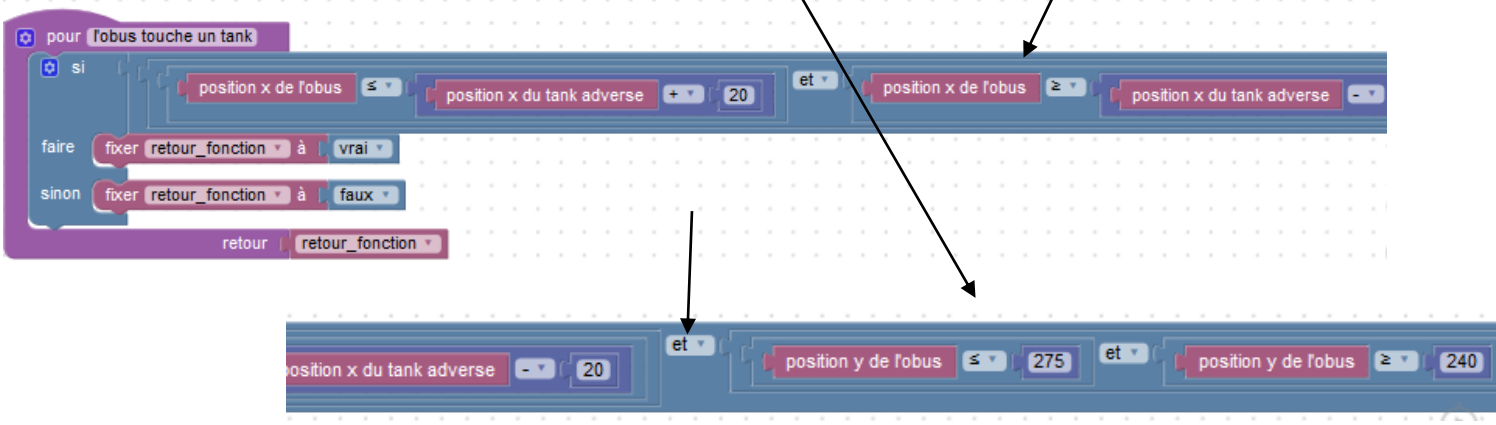


Quand un obus touche un tank

Voici la partie gauche de la condition (à gauche du ET).

Voici la partie droite de la condition (à droite du ET).

Et le reste de la fonction



Game Over

Il ne te reste plus qu'à coder l'écran informant les joueurs que la partie est finie et donnant le nom du vainqueur.

Dans le bloc détectant si un tank est touché par un obus, nous allons ajouter des instructions permettant d'afficher l'écran d'information de fin de partie.

Désolé, je ne dis rien de plus. Cherche donc les blocs dans les différentes catégories.

Code pour ce bloc :

```
for event in pygame.event.get():
  if event.type == "ObusMoving":
    setWeapon1VX((getWeapon1VX() + getWindVX()))
    setWeapon1VY((getWeapon1VY() - getGravity()))
    if l_obus_touche_le_sol():
      stopperObus()
      doExplode((getWeapon1X()),(getWeapon1Y()))
    if l_obus_touche_un_tank():
      stopperObus()
      doExplode((getOponantTankX()),(getOponantTankY()))
      if getOponantTank() == 2:
        showVictory(1)
      else:
        showVictory(2)
```

C'est fini ?

En ce qui concerne ce cahier d'activités, c'est terminé. J'espère qu'il t'aura donné envie de continuer la découverte de ce langage d'une autre manière.

Tu as acquis les bases qui te permettront d'aller plus loin dans l'apprentissage de la programmation et tu peux t'attaquer à la conception de programme plus évolué.

Le langage Python est un outil formidable pour continuer sur ta lancée : en t'aidant du module pygame.org et de ce que tu as appris, tu as en mains tout ce qu'il faut pour programmer des merveilles !

Passons maintenant à la programmation en Python dans le cahier d'activités suivant.

Bibliographie

- Python pour les Kids - Cahiers d'activités

- Site Internet : http://www.kidscod.in/app/tank/