TUTORIEL - SCRATCH : CIRCUIT DE VOITURE

CRÉATION DE CONTENU > 3.4 PROGRAMMATION

CONVIENT POUR	AGE	NIVEAU DE COMPÉTENCE	FORMAT	DROITS D'AUTEUR	LANGUE(S)
Elèves (école primaire), Elèves (école secondaire), Jeunes en décrochage scolaire	Adolescents, Enfants	Niveau 2	Fiche d'activité	Creative Commons (BY-SA)	Français

Ce tutoriel explique comment réaliser un jeu de circuit de voiture avec le logiciel Scratch.

Objectif général	Compétences
Temps de préparation pour l'animateur	moins d'une 1 heure
Domaine de compétence	3 – Création de contenu
Temps requis pour compléter l'activité (pour l'apprenant)	0 – 1 heure
Nom de l'auteur	Pierre Huguet
Matériel supplémentaire	Un ordinateur connecté à internet pour accéder à Scratch (https://scratch.mit.edu/) ou avec le logiciel Scratch pré-installé
Ressource originellement créée	Français

DÉROULÉ

1 Réalisation

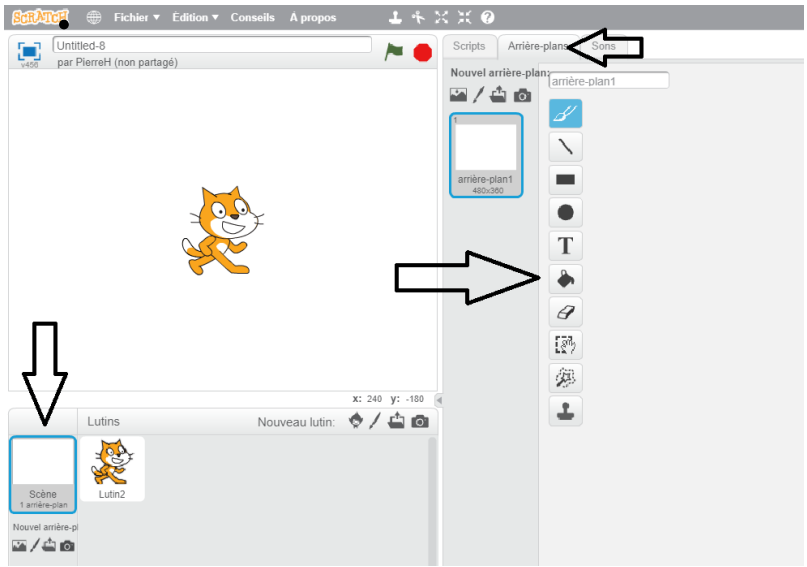
Le but de cette activité est de dessiner un circuit et y faire circuler une voiture en l'empêchant de sortir de la route. Lorsqu'elle touche la couleur de fond, elle revient au départ et lorsqu'elle touche la ligne d'arrivée, elle indique qu'elle a gagné.

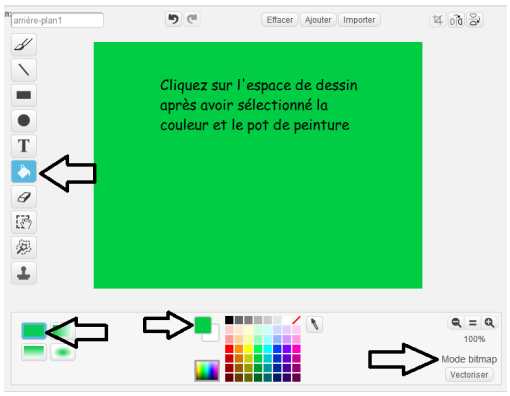
Vous pouvez suivre les indications de cette vidéo ou celles qui suivent.

Point de départ : Pour cette première version, vous pouvez créer un nouveau projet, ou démarrer à partir du canevas de projet https://scratch.mit.edu/projects/175146528/ qui comprend déjà un dessin de la piste et deux blocs de démarrage.

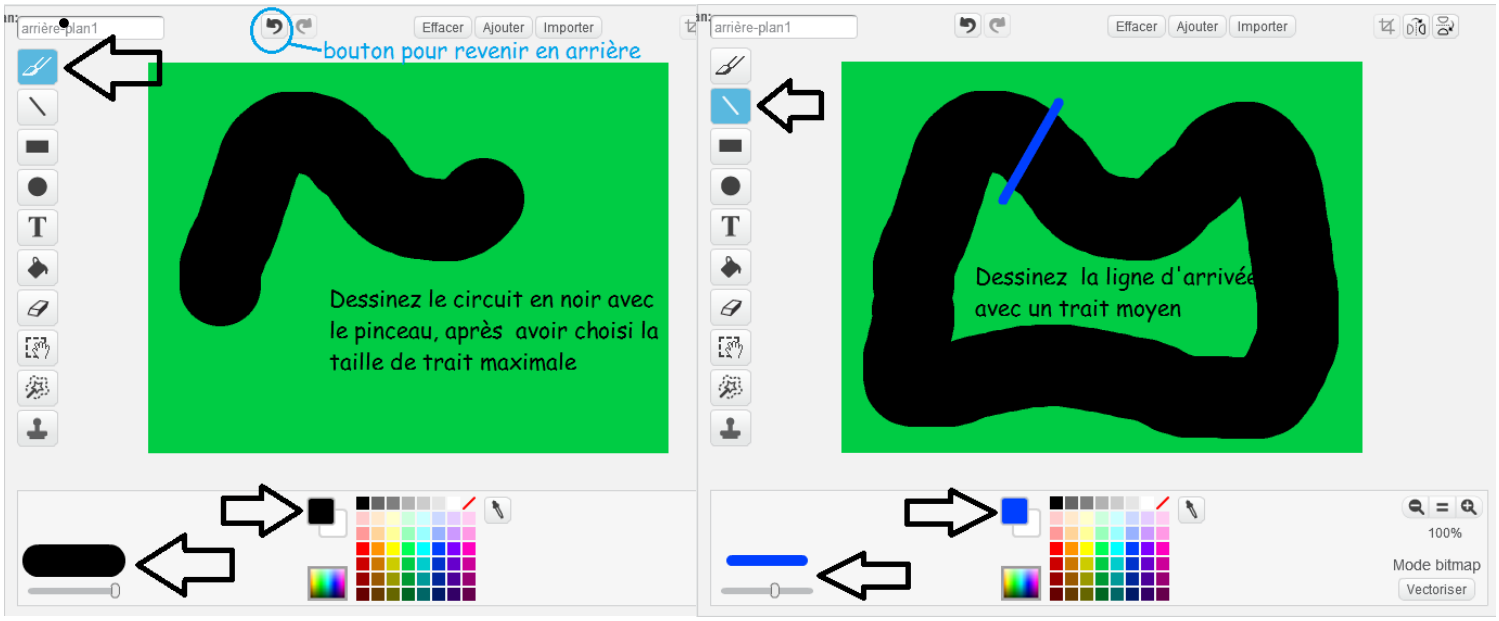
Dessin du circuit et création du lutin voiture

- Créez un nouveau projet Scratch et sélectionnez la scène (en bas à gauche) puis l'onglet Arrière-plan (en haut vers la droite).
- Vérifiez que vous êtes en mode bitmap avec le menu à gauche de la zone de dessin.
- Si le menu est à droite (mode vecteur), cliquez sur le bouton Bitmap en bas à droite.

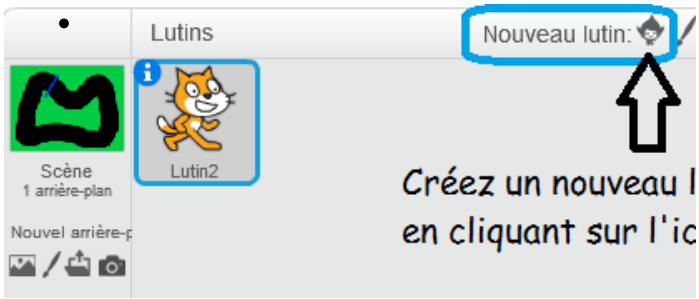


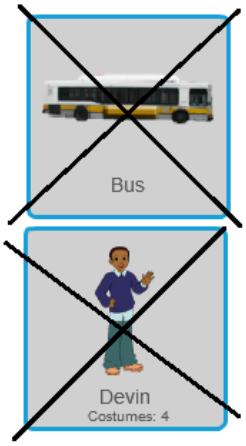


Remplir avec un fond vert, puis dessiner une route noire avec un pinceau de largeur maximale, et avec un trait plus fin, de couleur différente, une ligne droite qui correspond à la ligne d'arrivée.



Créez un nouveau lutin correspondant à la voiture, en choisissant une image dans la librairie. Choisissez plutôt une image compacte plutôt que pointue ou allongée, ce sera plus facile.

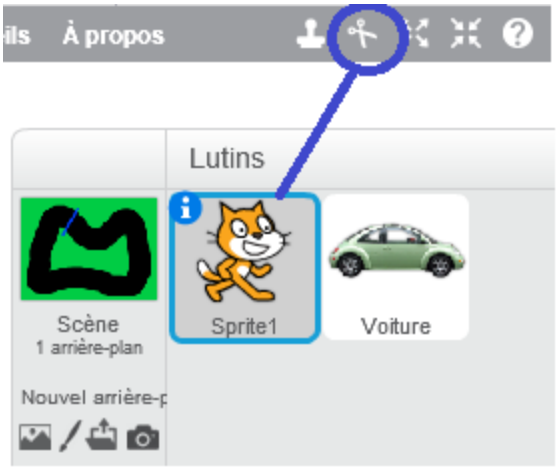




Renommez ce lutin « voiture » après avoir appuyé sur le i du lutin.

- Puis supprimez le lutin « Sprite1 » en sélectionnant les ciseaux (en haut) et en cliquant sur le lutin à supprimer.

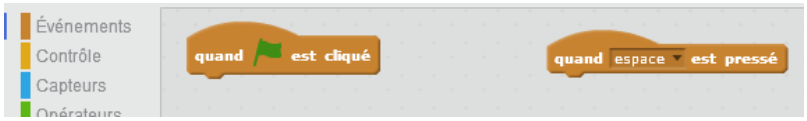
Nota bene : Vous pouvez aussi pointer sur le lutin à supprimer et cliquer avec le bouton droit pour faire apparaitre un menu contextuel (ou sur certains navigateurs en appuyant sur la touche majuscule/shft et en cliquant sur le bouton gauche). Vous aurez alors l'option « supprimer ».



Sélection des deux évènements (blocs de contrôle) auxquels le programme doit réagir.

Depuis l'onglet des blocs de contrôle dans le panier des blocs, glissez/déposez vers la zone de programmation

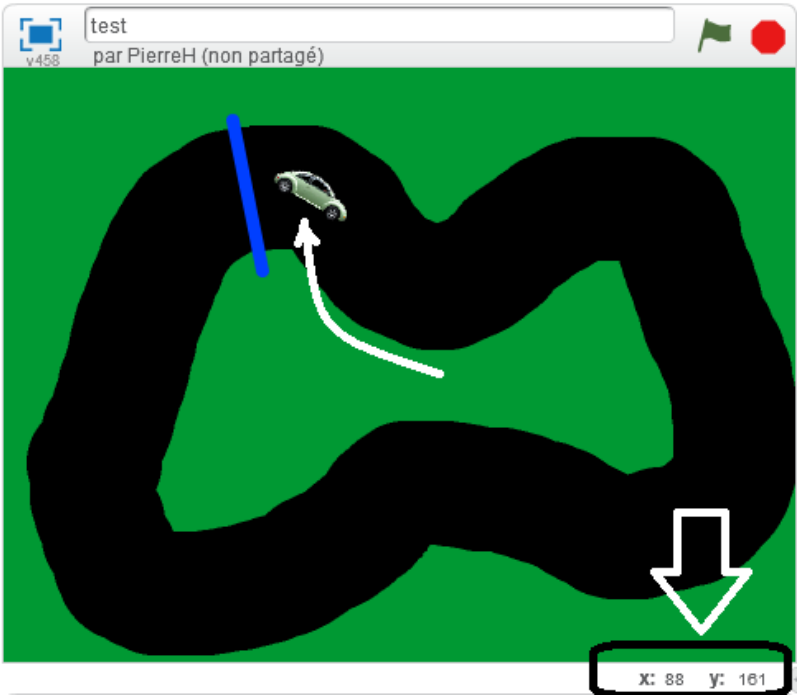
- un bloc « Quand on clique sur le drapeau vert « pour le démarrage du jeu
- et un bloc « Quand la barre d'espace est pressée » pour faire avancer la voiture



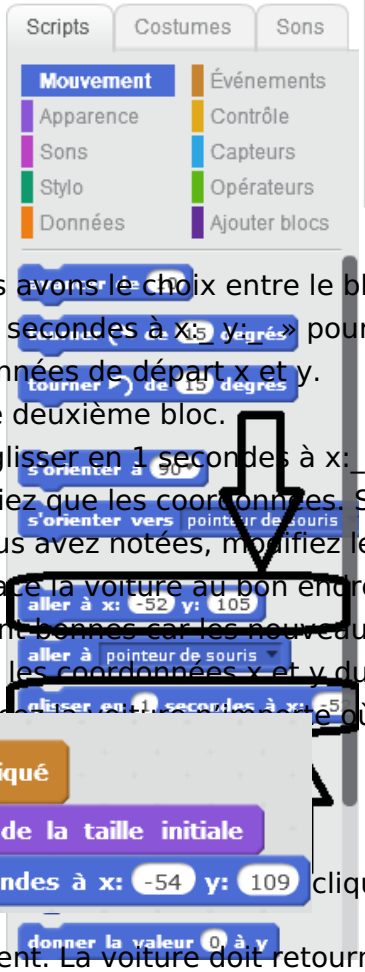
Programme ou script correspondant au clic sur Drapeau vert : démarrage du jeu

- Ajoutez un bloc de mise à l'échelle,
  et réglez la valeur à l'intérieur du bloc, pour que la taille de la voiture soit à peu près un tiers de la largeur du circuit;
  A chaque fois que l'on clique sur le bloc, il s'exécute et change la taille de la voiture pour la valeur donnée dans le bloc.

- Glissez / Déplacez la voiture avec la souris pour la mettre devant la ligne d'arrivée, sans qu'elle touche cette ligne, ni les bords de la route. Les coordonnées du curseur sont alors affichées en bas à droite.
  Vous pouvez donc connaitre les coordonnées de l'endroit où est la voiture en mettant le pointeur de la souris sur la voiture. Notez ces coordonnées x et y (par exemple x = -55 et y = 109).



- Nous allons maintenant ajouter un bloc qui va faire glisser la voiture à cet endroit au début du jeu, quand on clique sur le drapeau vert.



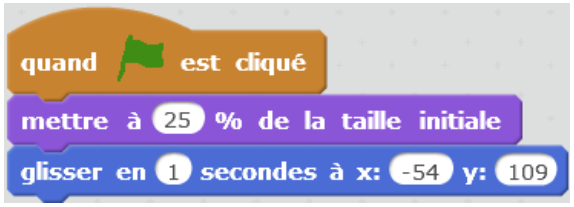
- Dans la catégorie (ou l'onglet) des mouvements nous avons le choix entre le bloc « aller à x:_ y:_ » ou le bloc « glisser en _ secondes à x:_ y:_ » pour déplacer ou faire glisser la voiture au coordonnées de départ x et y.
  Nous prendrons le deuxième bloc.
- Ajoutez le bloc « glisser en 1 secondes à x:_ y:_ » après le bloc de mise à la taille, puis vérifiez que les coordonnées. Si ce ne sont pas les mêmes que celles que vous avez notées, modifiez les avec les valeurs notées.
  Si vous avez déplacé la voiture au bon endroit juste avant, les coordonnées seront bonnes car les nouveaux blocs de déplacement ont automatiquement les coordonnées x et y du dernier déplacement.
- Pour tester, déplacez la voiture où avec la souris et

cliquez sur le drapeau vert ou le bloc de déplacement. La voiture doit retourner au point de départ.Nota 1:
le fait que les coordonnées des blocs de déplacement soient mises à jour automatiquement après chaque déplacement, évite de passer du temps à expliquer les systèmes de coordonnées, mais il

faut le signaler aux participants pour qu'ils comprennent pourquoi le lutin va au bon endroit.Nota 2: vous pouvez également ajouter un bloc d'orientation de la voiture dans le sens de la piste (cela évite parfois des contacts avec les bords quand la voiture est mal orientée).

Programme (script) correspondant à « quand barre d'espace pressée » : faire avancer la voiture

- Ajoutez un bloc d'orientation vers le pointeur de la souris
- Ajoutez un bloc pour avancer (de 4 pixels)



Remarque :

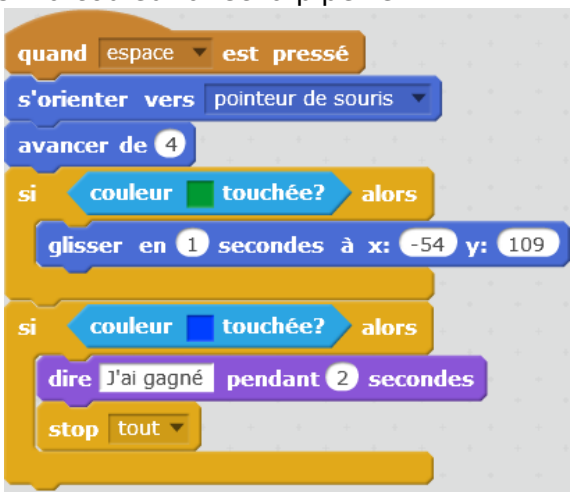
Si la barre d'espace reste pressée, après l'exécution de ces 2 blocs, ils sont de nouveau exécutés puisque la barre d'espace est toujours pressée. Donc la voiture avance tant que l'on appuie sur la barre d'espace. Nous modifierons ce comportement dans une version suivante.

Tester le fonctionnement du jeu en l'état, et discuter de ce qu'il faut faire :

- Retour au départ si la voiture touche le vert,
- Arrêt de la partie si la voiture touche le bleu (arrivée).

Compléter le programme ou le script qui fait avancer la voiture.

- Ajouter un bloc de contrôle de branchement conditionnel : si … alors
- Mettre dans la condition un test « couleur touchée » et choisir la couleur avec la pipette
- A l'intérieur du bloc de test, dupliquer le bloc qui renvoie au départ. Ce bloc ne sera exécuté que si la réponse du test est vraie (i.e. si la voiture touche le vert)
- Dupliquer le bloc de branchement conditionnel, mais avec la couleur de la ligne d'arrivée.
- A l'intérieur utilisez un bloc qui affiche « j'ai gagné » puis ajouter un bloc « stop tout ».

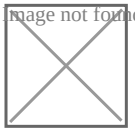


Tester le fonctionnement du jeu,
Debriefing sur les notions mises en œuvre et préparation des leçons suivantes

2 Revue des notions mises en œuvre

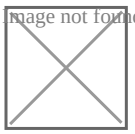
- **Programmer** : on programme pour réaliser quelque chose, comme un jeu ou une application utile à quelqu'un. Les jeux les plus réussis sont ceux où le.a créateur.rice a pris le temps de penser à la manière dont les utilisateur.rice.s vont jouer et à ce qui va les amuser.
- Un programme est construit avec des **séquences de blocs** qui commencent par un évènement. La séquence d'instructions qui correspond à chaque évènement s'appelle un **script**. Chaque script concerne le lutin dans lequel il est écrit (par exemple la voiture). Dans un programme il peut y avoir beaucoup de lutins et beaucoup de scripts. Cela devient parfois difficile de s'en rappeler.
- La **couleur des blocs** : La couleur des blocs correspond à leur **catégorie fonctionnelle**.
  - Marron foncé pour les évènements qui démarrent des séquences d'instructions, par exemple quand le joueur clique sur le drapeau vert ou appuie sur la barre d'espace.
  - Beige pur les blocs de contrôle, qui permettent de modifier le déroulement de la suite du programme.
    Nous avons utilisé un bloc si : les instructions à l'intérieur ne sont exécutées que si la condition est vérifiée ou vraie.
  - Bleu pour les blocs ou instructions de mouvement : déplacements et orientation,
  - Bleu/violet pour les blocs qui modifient l'apparence dont la taille ou les bulles de texte,
  - Violet pour les sons que nous utiliserons plus tard,
  - Bleu ciel pour les capteurs,comme le bloc qui capte si une couleur est touchée
  - Orange pour les données et vert pour les blocs de calcul.
- La **forme des blocs** correspond à leur contenu et à la manière dont on peut les assembler.
  - La plupart des blocs ont une encoche au-dessus et une en-dessous. On peut les empiler en séquence dans un script. Ils sont exécutés dans l'ordre, de haut en bas.
  - Certains blocs ont le dos rond, on ne peut rien mettre avant.
    Ils démarrent chaque séquence ou script et correspondent le plus souvent aux évènements extérieurs au programme, par exemple quand le joueur clicke sur le drapeau vert. (Nous verrons plus tard que le programme peut également générer des évènements, avec les messages).
  - Certains blocs ont le fond plat, on ne peut pas coller de bloc en dessous. C'est le cas du bloc stop tout
  - Il y a aussi des blocs qui n'ont pas d'encoche et qui sont arrondis ou pointus sur les côtés. Ces blocs contiennent une valeur, que l'on utilise à l'intérieur des autres blocs, comme les blocs de contrôle.
  - Les blocs pointus ne peuvent prendre que 2 valeurs, vrai ou faux. Ils correspondent à ce

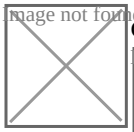
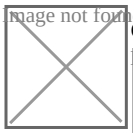
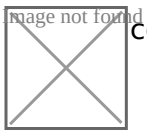
qu'on appelle des variables booléennes. On les utilise surtout dans les blocs de contrôle, par exemple pour savoir si les instructions à l'intérieur du bloc doivent être exécutées.



- Les blocs arrondis que nous verrons dans la leçon suivante, peuvent contenir des nombres ou des textes ou chaînes de caractères.

- La **valeur de ces blocs** peut être le résultat d'un calcul ou d'un test, par exemple



  -  contient la valeur Faux,
  -  contient la valeur 7,
  -  contient la valeur textuelle »Vous avez perdu »,

3 Préparer les étapes suivantes

Faites discuter les participant.e.s sur le jeu et la manière de l'améliorer.
Comme ielles en sont à leur 1° ou 2° leçon, vous ne pourrez pas passer tout de suite en création libre, mais vous pourrez situer les prochaines leçons comme des améliorations de la première version. Dans un premier temps, proposez de travailler sur les idées ci-dessous. Nous les avons classées dans l'ordre des exercices suivants :

- Ajouter des sons
- Compter les vies (fin du jeu quand la voiture n'a plus de vies)
- Changer la vitesse de la voiture, accélérer et ralentir
- Chronométrer les tours,
- Compter les tours,
- Choisir et changer sa voiture
- Mettre des obstacles,
- Attaquer la voiture, lancer des pierres,
- Passer par une pompe à essence, tomber en panne
- Faire plusieurs niveaux.

4 Pour aller plus loin

Conseil médiation

Pour aller plus plus loin sur le sujet, nous vous conseillons de vous référer à la fiche outil « Les erreurs fréquentes dans Scratch«