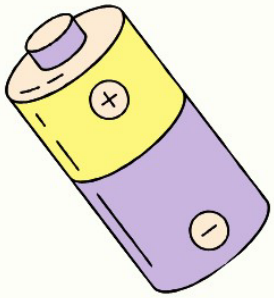
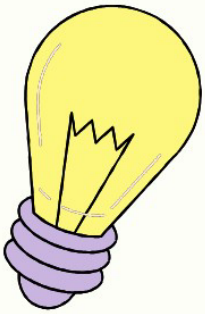
MANUEL D'UTILISATION DE

CONSTRUCT 3





CRÉATION D'UN JEU VIDEO SUR L'ÉLECTRICITÉ

Tables des matières

Préambule **3**
Qu'est-ce que Construct 3 et pourquoi l'utiliser ? 3
Exemplification d'un projet de jeu vidéo sur Construct 3 4
Accès au jeu 4
Descriptif 4
Cheminement du jeu 5

Découverte de l'interface de Construct 3 ..... **6**

Créer un nouveau projet **7**
Comprendre l'espace du jeu 7
Tester le jeu vidéo 8
Créer un *sprite* 9
Les différents plans sur Construct 3 11
Les boîtes de collision 12
Programmation avancée : la feuille d'évènements 13
Importer une image dans Construct 3 16

Programmation avancée : les mécaniques de jeu en détail **17**
Changer la taille de son niveau (Difficulté : *) 17
Écrire du texte dans le jeu (Difficulté : **) 18
Mettre en mouvement un ennemi ou une plateforme (Difficulté : *) 18
Lancer des projectiles pour combattre des ennemis (Difficulté : ***) 19
Mettre une limite de temps (Difficulté ****) 21
Allumer/éteindre la lumière (Difficulté : **) 24
Faire se retourner le personnage lorsqu'il change de direction (Difficulté : 25
**) 25
Créer une animation de saut (Difficulté : ***) 26
Changer l'apparence et les propriétés du personnage jouable (Difficulté : 28
*****[3]) 28
Les questions fréquemment posées en animation 32
Ressources utiles 33

Préambule

Qu'est-ce que Construct 3 et pourquoi l'utiliser ?

Pour réaliser un jeu ou une application, il faut écrire un **programme**, un ensemble d'instructions que l'ordinateur va suivre l'une après l'autre afin de savoir quoi faire et comment réagir (à un clic de souris, à une touche, ...).

Mais un programme, cela ne s'écrit pas en français (ni en anglais). Cela s'écrit dans un langage que l'ordinateur comprend. On appelle ça un **langage de programmation**.

Il existe plusieurs langages différents et il faut apprendre le vocabulaire, la grammaire et la façon dont on construit les phrases pour chacun de ces langages. Mais même si l'orthographe et la grammaire diffèrent, il existe des façons de réfléchir (ou « logiques ») qui sont communes à tous les langages.

Construct 3 est un site qui permet de créer des jeux vidéo en représentant de manière visuelle cette logique de programmation.

Si nous avons choisi de nous tourner vers Construct plutôt que vers un autre logiciel, c'est pour différentes raisons :

1. Construct offre des possibilités quasi infinies tout en étant relativement simple d'exécution
2. Le logiciel est en français et ne demande aucune connaissance préalable en programmation pour pouvoir créer son propre jeu vidéo
3. Le programme permet d'exporter facilement des jeux sur différents supports (site internet, consoles de jeu, mobile). De plus, Construct s'utilise via un site internet et ne nécessite aucune installation.

Avec Construct, il est donc possible de faire facilement des jeux en 2D comme :

- Des jeux de plateforme (comme Mario) ;
- Des RPGs (comme Zelda) ;
- Des Tetris, Pacman, casse-briques, etc.

Exemplification d’un projet de jeu vidéo sur Construct 3

Ce jeu a été conçu à titre d'exemple, servant de modèle autour duquel les enfants peuvent créer leur propre jeu vidéo. Il intègre une synthèse des leçons sur l'électricité, appliquées de manière ludique. Ainsi, le joueur résoudra des énigmes portant sur les conducteurs et les isolants, ainsi que sur les différents types de circuits électriques

:

Accès au jeu

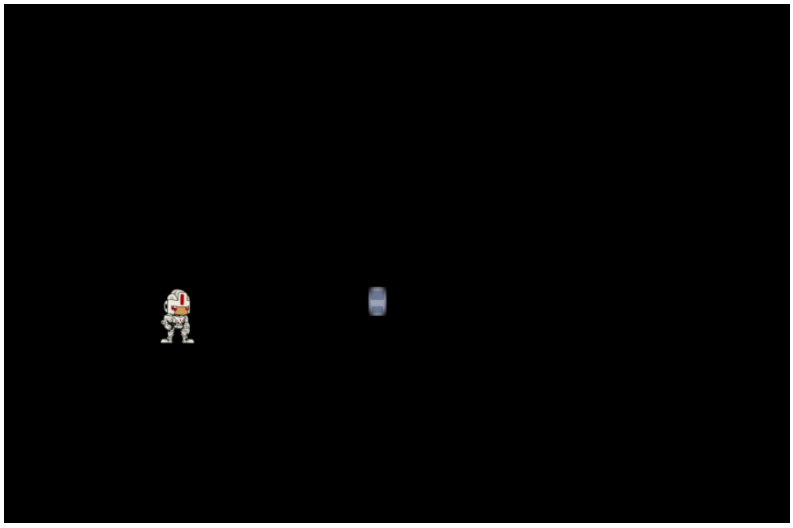
https://kodowallonie.itch.io/exemple-electricit

Descriptif

Dans ce jeu, vous devrez résoudre une série de petites énigmes comme allumer des interrupteurs, rendre le héros insensible à l'électricité et aux robot ennemis grâce à des matériaux conducteur et isolant, résoudre des énigmes grâce à des circuits électrique, réactiver le courant mais attentions toutes ces opérations devront être réalisés avant que la batterie du robot ne tombe à plat.

Cheminement du jeu

Scène 1



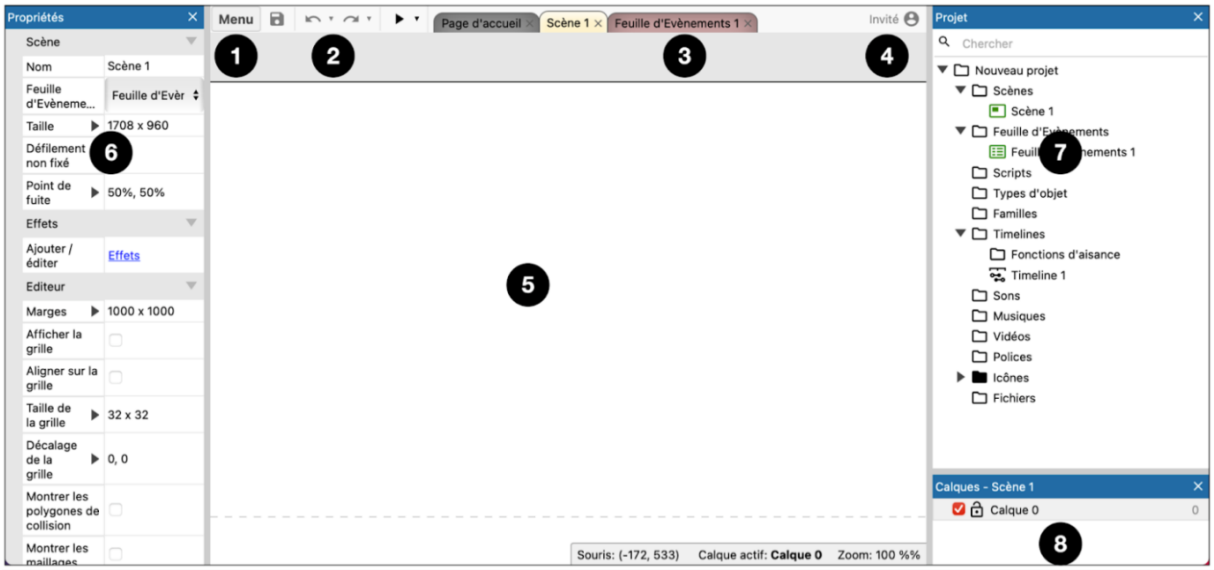
- Pour allumer le courant, le joueur doit se placer sur l'interrupteur et appuyer sur la touche E.
- Une fois le courant allumé, le joueur pourra passer à l'écran suivant.

Scène 2

- Dans cette scène, le joueur devra esquiver le premier robot.
- Il devra ensuite atteindre la bouteille en plastique qui transformera la combinaison du héros, cette transformation en plastique permettra de devenir isolant afin de pouvoir traverser les pièges électriques
- La dernière étape du jeu demande d'utiliser l'interrupteur pour fermer le circuit électrique et activer l'ascenseur qui permet d'atteindre la batterie.

Découverte de l'interface de Construct 3

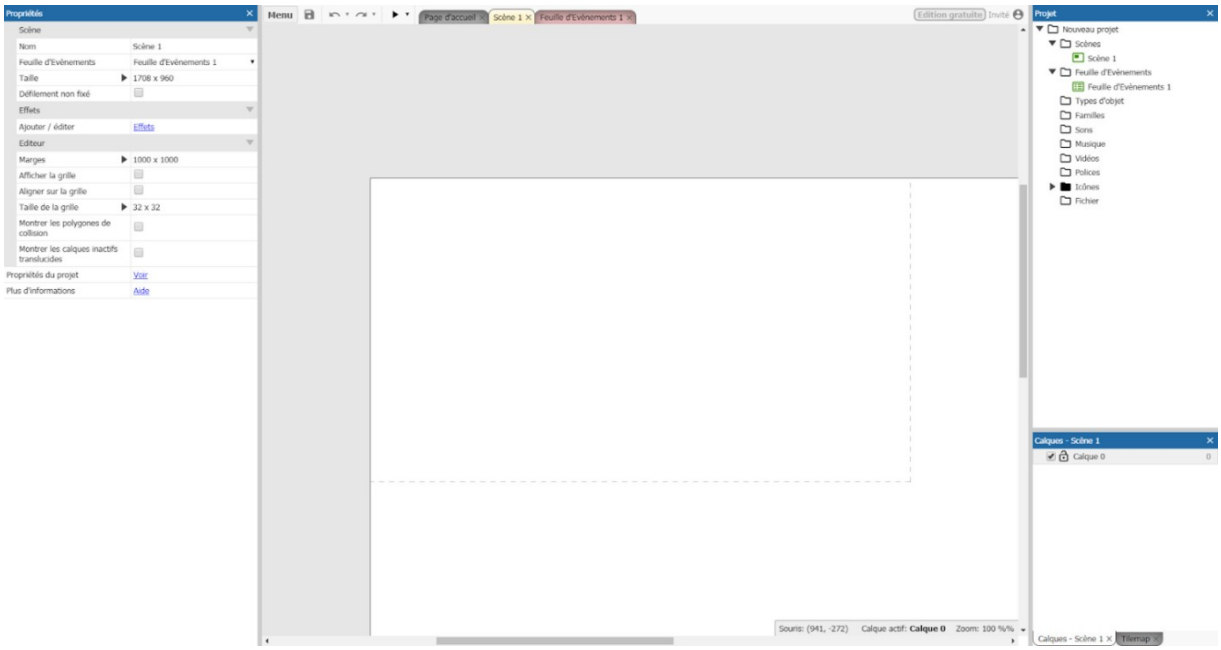
L'image suivante met en évidence les parties importantes de l'interface utilisateur (IU) de Construct 3.



1. Le menu principal : cliquez sur ce bouton pour ouvrir le menu principal. Celui-ci fournit des options pour les tâches de base comme l'ouverture et la fermeture de projets, l'exportation, la modification des paramètres, etc.
2. Barre d'outils principale : elle fournit des raccourcis vers les fonctions les plus utilisées : enregistrer, annuler, rétablir et jouer à son jeu.
3. Onglets d'affichage : ces onglets vous permettent de basculer entre vos différents niveaux et feuilles d'événements (endroits où vous programmez votre jeu).
4. Badge de compte : il indique l'état de votre compte.
5. Vue principale : c'est là qu'apparaît la vue le niveau ou la vue de feuille d'événements actuellement sélectionnée.
6. Barre de propriétés : elle répertorie toutes les propriétés de l'élément sélectionné, ce qui vous permet d'en modifier les paramètres.
7. Barre de projet : elle répertorie tous les éléments de votre projet (images, musiques...). Elle vous donne un aperçu de ce que vous avez ajouté et vous permet de naviguer dans le projet, par exemple en ouvrant des modèles de personnage ou des feuilles d'événement pour les visualiser.
8. Barre des calques : lorsqu'un élément est ouvert, elle affiche les calques de celui-ci

Créer un nouveau projet

Cliquez sur le bouton **Nouveau projet**. Une fenêtre de dialogue apparaît, vous demandant certains détails. Vous n'avez besoin de rien changer, mais vous pouvez ajouter un nom à votre projet si vous le souhaitez (pourquoi pas : ""Mon nom + mon prénom : jeu"). Cliquez sur **Créer** et vous devriez observer sur votre écran quelque chose qui ressemble àceci :



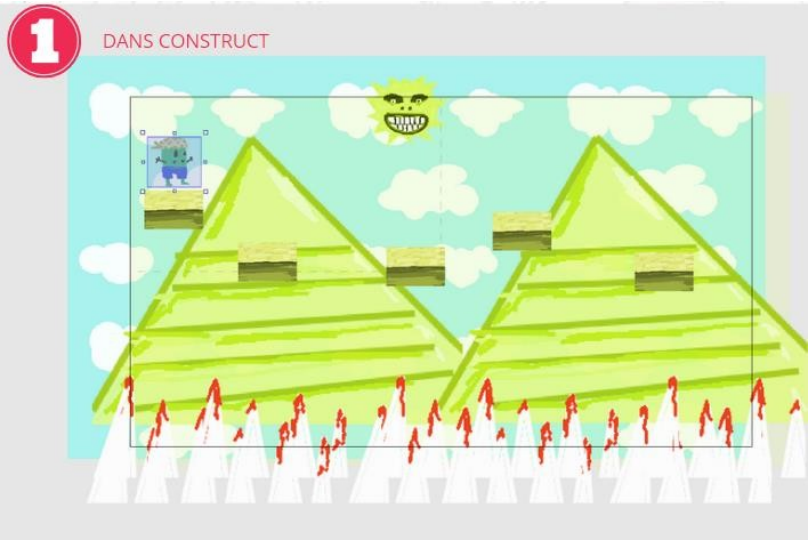
Remarque : Pour zoomer et dézoomer sur un ordinateur, il faut laisser son doigt appuyé sur la touche CTRL tout en bougeant la petite roulette de la souris.

Comprendre l'espace du jeu

En observant bien votre écran, on peut voir **deux rectangles.** Le premier, le plus grand est un grand **rectangle blanc** et le second, plus petit est le petit rectangle délimité par des **pointillés**.

Le grand rectangle blanc correspond au **niveau du jeu**. Ainsi, on voit tout le parcours que notre personnage va devoir accomplir pour arriver jusqu'à la fin du niveau[1] et aller au suivant.

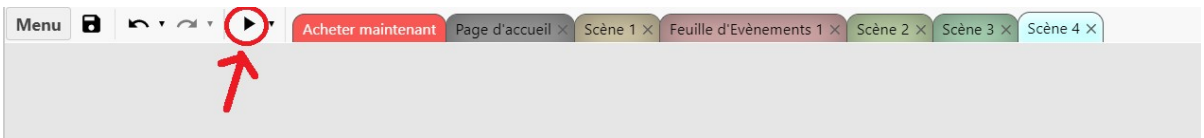
[1] Le niveau d'un jeu vidéo est le parcours que le personnage doit traverser pour continuer sa progression. Il constitue une étapeà franchir et est un élément structurant à la manière d'un chapitre de livre.



Le petit rectangle en pointillé correspond à la **caméra** : ce que le joueur voit du niveau. Pour faire bouger la caméra et voir la suite du niveau, rien de plus simple : il faut bouger le personnage.

Tester le jeu vidéo

Pour **lancer** son jeu, rendez-vous en haut de la page.

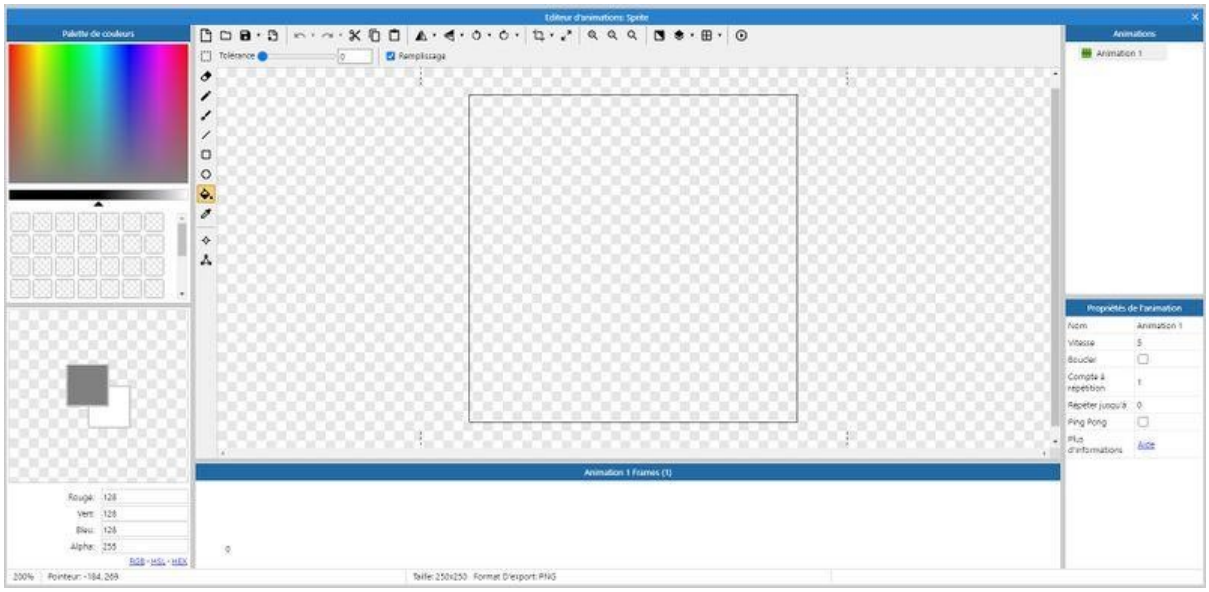


En appuyant sur cette petite flèche qui signifie "**play**", une nouvelle fenêtre s'ouvre. Il s'agit du jeu vidéo.

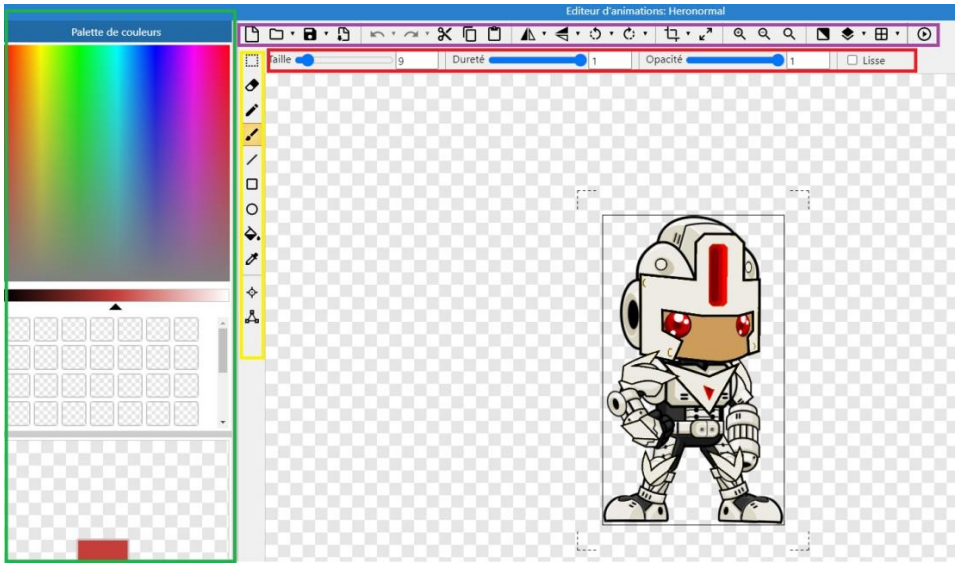
Créer un *sprite*

Au sein d'un jeu de plateforme, les premières choses à ajouter sont : un personnage et des plateformes. Dans Construct, tous les éléments graphiques se nomment les ***sprites***.

Pour créer un nouveau *sprite* : clic droit > Insérer un nouvel objet > *Sprite*. Construct 3 embarque un **éditeur de *sprite***.



Dans l'éditeur de création de *sprite*, on trouve des pinceaux (traits ronds), des crayons (traits carrés), un outil pour créer des formes, des lignes ou encore pour coloriser rapidement le dessin.



Rectangle vert : outils pour changer de couleur

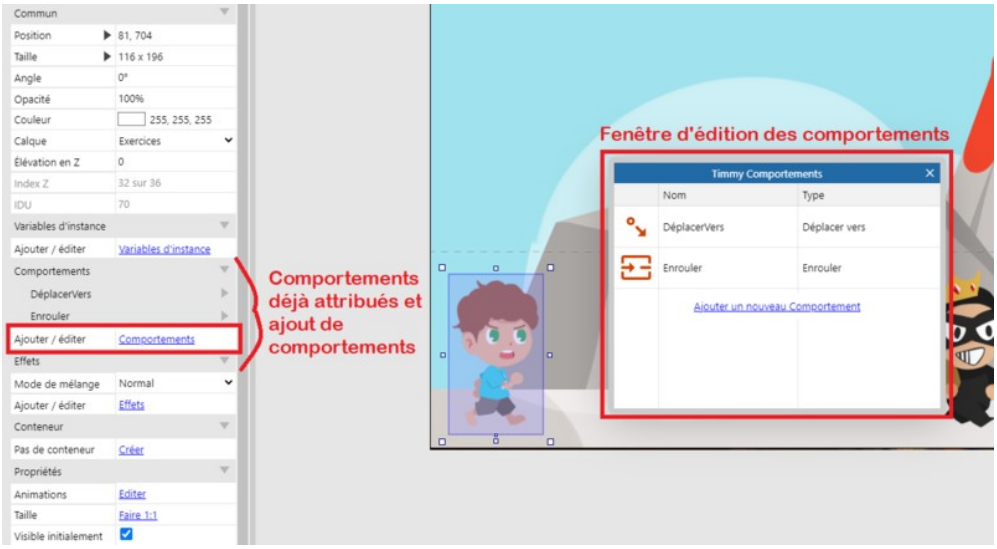
Rectangle jaune : outils pour dessiner

Rectangle rouge : outils pour changer la taille de son trait

Rectangle mauve : paramètres du dessin

Une fois le dessin terminé, il est nécessaire de **fermer cette fenêtre** en cliquant sur la croix sur le coin supérieur droit de l'éditeur de *sprite*.

Les *sprites* ne sont que des dessins numériques en 2D. De facto, ils sont immobiles. Pour commencer à les programmer, il va falloir leur associer des **comportements.**



Pour donner un comportement à un sprite, il faut cliquer sur ce dernier et se rendre dans le **menu propriété** qui vient de s'ouvrir à gauche de l'écran.

Dans le menu propriété, il faut appuyer sur le mot comportement souligné en bleu pour ajouter/modifier/enlever des comportements

Les trois comportements à associer impérativement sont

-Le comportement **PLATEFORME** à associer au **personnage** afin qu'il puisse se mouvoir dans un environnement en 2D.

-Le comportement **DEFILER VERS** à associer au personnage afin que la **caméra** suive la progression du joueur.

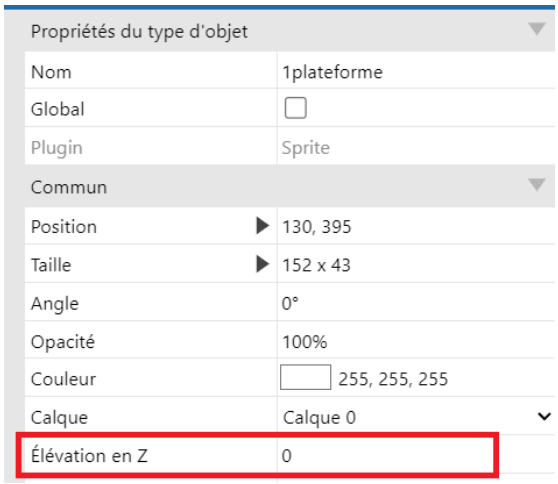
-Le comportement **SOLIDE** à associer à la **plateforme** pour que le personnage ne la traverse pas lorsqu'il saute dessus.

Remarque : Le comportement **PLATEFORME** ne doit pas être associés aux **plateformes** (objets sur lesquels le personnage va pouvoir se mouvoir).

Les différents plans sur Construct 3

Sur Construct 3, à chaque fois que l'on ajoute un nouvel élément, il se met devant le précédent. Il est donc compliqué de manipuler aisément les objets. Pour définir quel élément est devant ou derrière, il faut modifier ce que l'on nomme **l'élévation en Z**. Naturellement, l'élévation en Z est définie à 0.

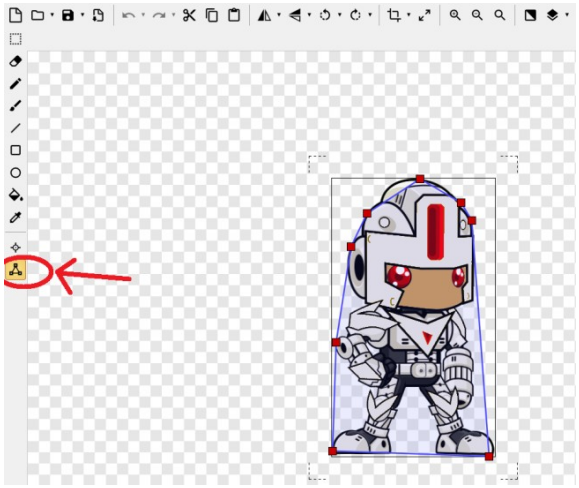
Les *sprites* avec une élévation en Z plus élevée se mettront devant et, au contraire, ceux avec une élévation en Z plus basses mettront à l'arrière-plan. L'arrière-plan se mettra en -1.



Les boîtes de collision

Par définition, un *sprite* comprend à la fois le dessin réalisé et toute la zone autour de celui-ci. Dès lors, en cas de collision avec d'autres objets, le personnage peut sembler léviter. Pour régler ce problème, il est nécessaire de régler la boîte de collision. Cette boîte de collision symbolise la forme réelle du dessin.

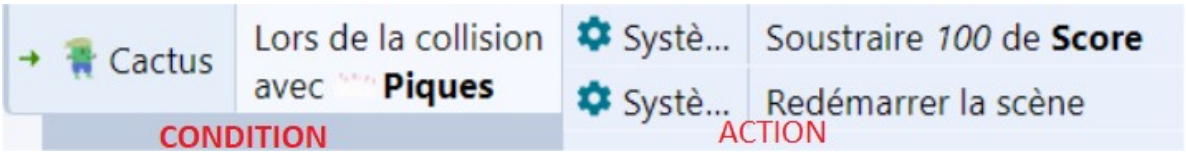
Pour sculpter les boites de collision, il faut cliquer sur l'objet pour retourner sur l'outil de **création de *sprite***. Clique ensuite sur l'icône en forme de triangle pour voir les points apparaître.



Remarque : cela ne sert à rien de prendre de longues minutes pour sculpter au centimètre près la silhouette du personnage. Une boîte de collision trop complexe risque de faire bugger le jeu. Il convient plutôt de réaliser une boîte de collision composée de 4 points en carré entourant le personnage.

Programmation avancée : la feuille d'évènements

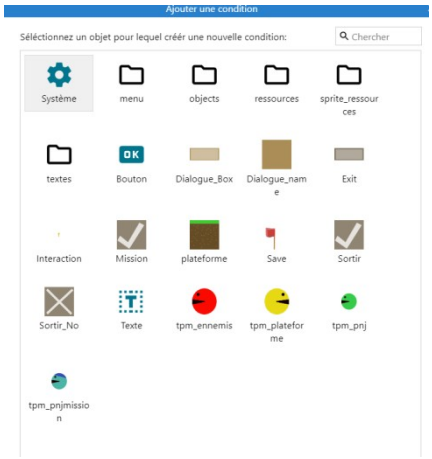
Les comportements ne sont pas suffisants pour programmer son jeu vidéo : ils sont un moyen rapide d'apposer des programmes automatisés aux ressources. Pour programmer son jeu en profondeur, la création **d'évènements** est indispensable. Dans Construct, un évènement est une **action** résultant d'une **condition.**



Dans ce cas-précis, le joueur perd du score et redémarre la partie.

Pour ajouter un évènement, il faut se rendre dans la **feuille d'événements**. Celle-ci se trouve dans les **onglets d'affichage** (voir schéma p.6).

Dans un premier temps, il faut programmer la **CONDITION :**

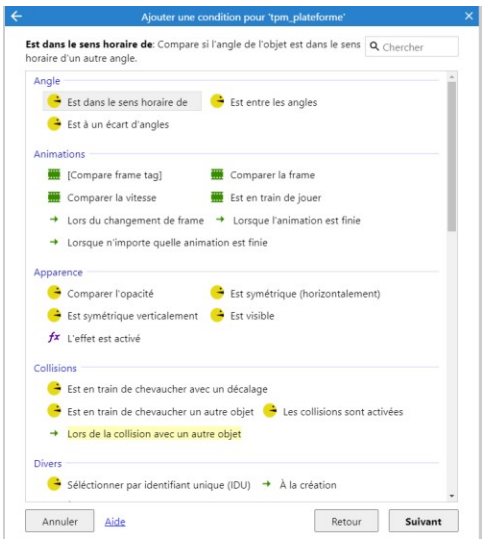


:

Pour ce faire, il faut sélectionner l'objet ou le personnage concerné par cette condition.

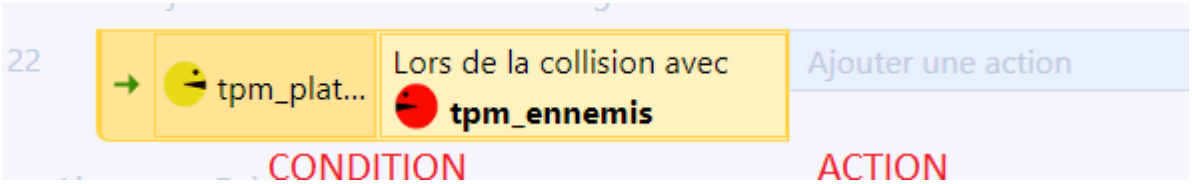
- Est-ce le *personnage* ? Dans ce cas, il faut cliquer sur le **personnage** et le définir comme *objet*
- Est-ce *tout le jeu* qui est concerné ? Alors, il faut cliquer sur **Système** pour le définir comme objet

Dans la phrase "*Quand mon personnage touche un ennemi, le niveau redémarre"* : c'est le **personnage** qui est concerné.

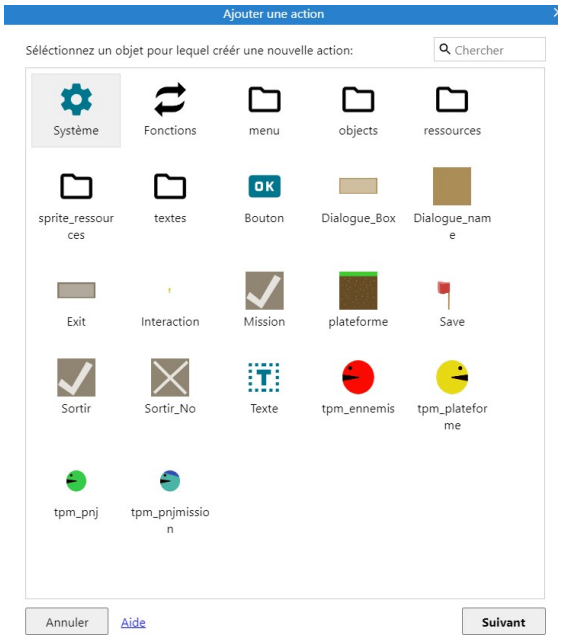


Il faut chercher désormais : « Lors de la **collision** avec un autre objet ». Une **collision** est le choc de deux objets qui se touchent.

Lors d'une collision, il faut sélectionner le **deuxième objet** avec lequel le premier vient d'entrer en collision, dans ce cas-ci : l'ennemi.



La première partie de la phrase est désormais réalisée. La condition est désormais programmée. Pour finir la ligne de programmation, il est nécessaire d'ajouter l'action. Pour ce faire, nous appuierons sur le bouton "Ajouter une action".



Désormais, il faut choisir **Système**. Car, ce n'est pas le personnage qui est détruit, mais plutôt tout le jeu qui va **redémarrer**

Cherchez "Redémarrer la scène" et confirmez la programmation.

Importer une image dans Construct 3

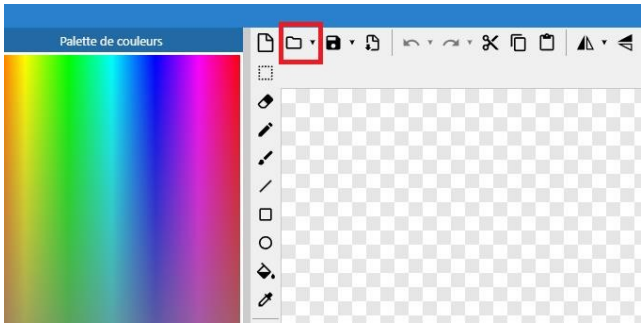
Si les ressources des élèves sont créées sur papier, il va être nécessaire de les **importer** dans Construct pour les manipuler.

Pour ce faire, les dessins devront être numérisés et devront être sauvegardés en **fichiers images** (jpeg., gif, png) et non pas en PDF. Le format PDF n'est pas lu par Construct.

Par facilité, chaque fichier image ne contiendra qu'**un dessin**.

Une fois cette étape réalisée, l'importation se déroulera comme suit :

Dans l'éditeur de *sprite* (voir p. 9), nous appuierons sur l'icône entourée ci-dessous :



Brève explication sur la transparence :

Une fois, le dessin importé cela ressemble à ceci. Comme nous sommes en "thème sombre", nous constatons que le fond derrière le personnage est blanc.

Pour l'importer parfaitement dans le jeu, il va être nécessaire de rendre ce **fond transparent**. Pour ce faire, nous allons utiliser la **couleur transparente**. La couleur transparente est représentée par les damiers blancs et gris situés en dessous de la roue des couleurs de l'éditeur de sprite. Pour enlever le fond avec précision, la j**auge de tolérance** située au-dessus de l'éditeur de sprite s'avère très utile.

Programmation avancée : les mécaniques de jeu en détail

Cette section couvre un bon aperçu des mécaniques de jeu et fonctionnalités programmables au sein du jeu vidéo. Pour chaque partie, un code allant de 1 à 5 étoiles a été apposé. Ce code correspond à la difficulté et au temps requis pour l'application de la mécanique de jeu à programmer. Ces étoiles peuvent ainsi servir de guide à l'enseignant pour choisir les mécaniques qu'il souhaite enseigner aux élèves en fonction du temps dont il dispose.

Remarque : Certaines mécaniques de jeu requiert des prérequis. Le cas échant, il en sera fait mention.

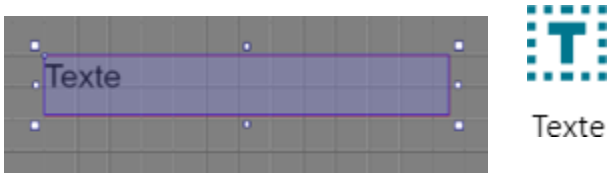
Changer la taille de son niveau (Difficulté : *)

1) Dans le menu à droite de l'écran (barre de projet), faire un clic droit sur Scène 1
2) Dans le menu "propriété", rechercher la ligne nommée **Taille** et mettre des nombres plus élevés pour agrandir le jeu.

Nombres plus élevés en x : agrandissent la scène horizontalement
Nombres plus élevés en y : agrandissent la scène verticalement

Écrire du texte dans le jeu (Difficulté : **)

Pour écrire du texte dans le jeu, il faut se rendre dans les menus de création d'un nouvel objet via le clic droit. Ensuite, sélectionner "Texte"



Dans le menu de gauche, celui des **propriétés,** il est possible de changer la taille, la couleur ainsi que la police d'écriture

Mettre en mouvement un ennemi ou une plateforme (Difficulté : *)

Pour qu'un objet bouge, il faut lui associer le comportement **sinus**

Sinus fait bouger les objets en leur faisant faire des aller/retour. Pour programmer ce comportement avec précision, il faut aller dans le menu "propriété" à gauche de l'écran

- En changeant l'option de **Mouvement,** nous influons sur la **nature** de l'aller-retour (de haut en bas, de gauche à droite, plus clair à plus sombre, plus grand à plus petit) ;
- En modifiant la **période,** nous influons sur la **vitesse** de l'aller/retour. Attention, plus la période est courte plus l'aller/retour est rapide et plus la période est grande plus l'aller/retour prend du temps ;
- En modifiant la **magnitude,** nous influons sur **distance** de l'aller/retour.

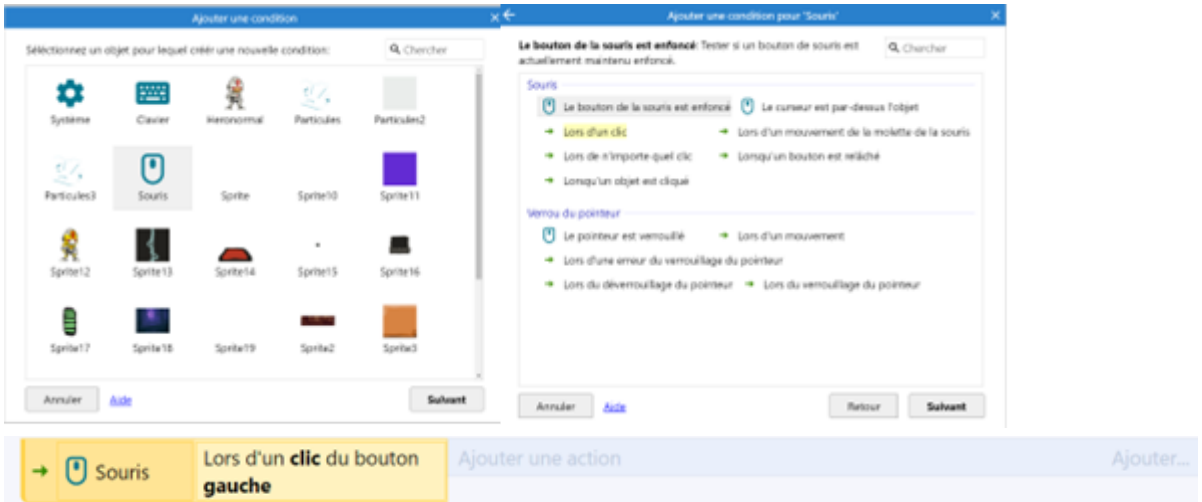
Lancer des projectiles pour combattre des ennemis (Difficulté : ***)

Pour que le personnage tire un projectile, il est indispensable de créer un nouveau *sprite* ressemblant au projectile que l'on veut lancer. Une fois le projectile dessiné, on va lui ajouter le comportement projectile.

Pour ajouter la souris aux contrôles acceptés par le jeu, il faut se rendre dans le menu de création d'un nouvel objet via le clic droit de la souris.

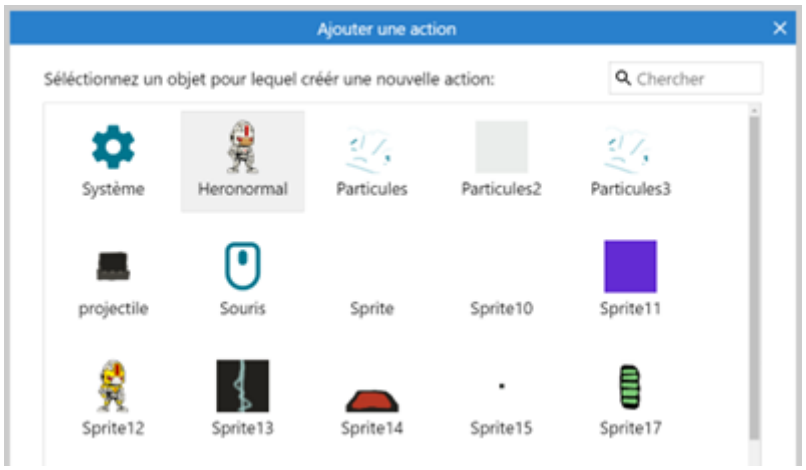
Pour programmer l'action de tirer, lorsque l'on clique sur le clic gauche de la souris, nous suivrons les indications ci-dessous.

Ajouter un nouvel évènement > nouvelle condition > souris > lors d'un clic < lors d'un clic gauche.



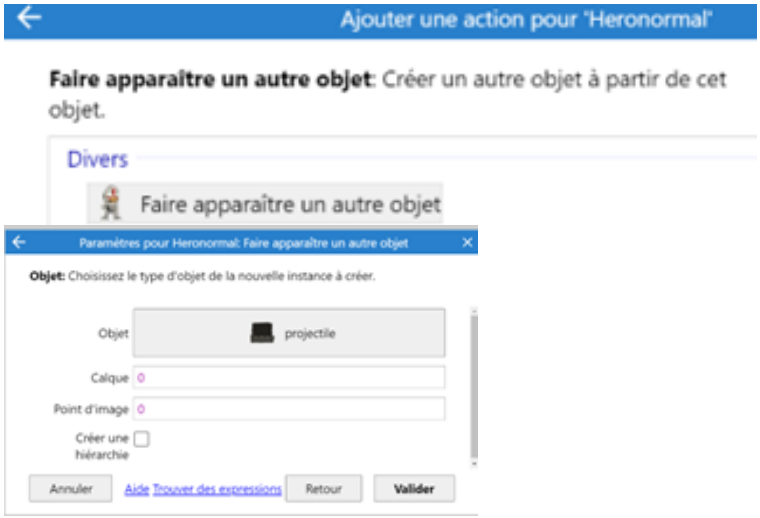
Pour ajouter l'action du projectile qui "part" du personnage, nous allons procéder comme suit : dans le menu ci-dessous, nous allons choisir notre personnage.

=> Le projectile doit sortir de lui !



Et nous allons sélectionner le code : faire apparaître un autre objet

=> Notre personnage va faire apparaître **le projectile.**



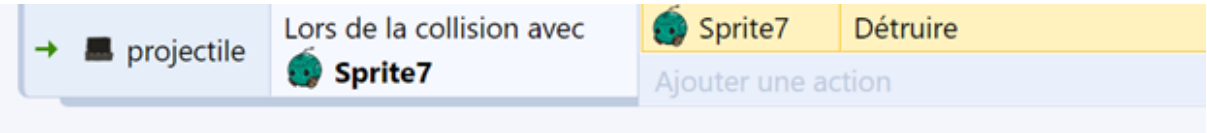
La ligne de code entière donne ceci :

Pour pouvoir viser à l'endroit exact où se situe la souris, il faut ajouter une nouvelle action en dessous de celle déjà crée et ensuite choisir : définir l'angle vers une position

La valeur en X doit être : souris.X
La valeur en Y doit être : souris.Y

Pour que les projectiles touchent et détruisent les ennemis, nous allons utiliser un type de programmation déjà mentionnée auparavant :

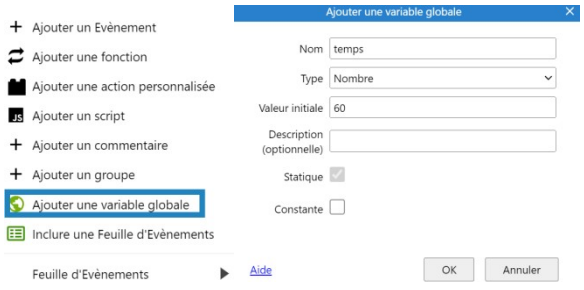
Quand le projectile touche l'ennemi => Alors l'ennemi est détruit. Dans Construct, cela donne :



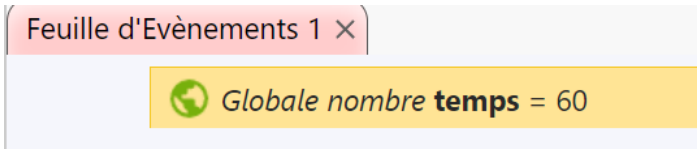
Mettre une limite de temps (Difficulté ****)

Pour mettre une limite de temps, il faut créer une **variable.** Une variable est un nombre qui change en fonction de divers éléments.

Dans la feuille d'événements, plaçons notre curseur sur la zone grise et appuyons sur le clic droit (comme si nous voulions ajouter un sprite). Remplissons ensuite la fenêtre qui vient de s'ouvrir comme ceci :



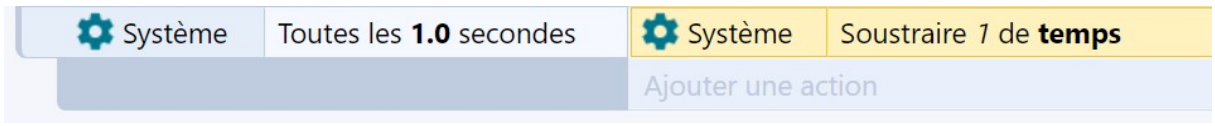
Lors de la création d'une nouvelle variable, une indication de ce type, reconnaissable par l'icône de planète, apparait :



Il faut désormais programmer le fait que toutes les secondes, la variable temps perd 1 de valeur.

Nous allons aller dans *Créer un nouvel événement > Système > toutes les X secondes > 1.0 > Système > Soustraire de (variable) > Valeur 1*

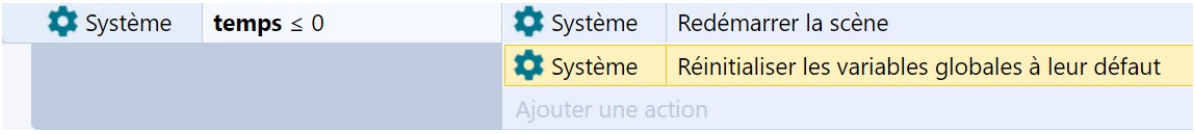
Cela donne ceci :



Désormais un compteur invisible est activé en permanence sur le jeu et toutes les secondes, la variable temps perd 1.

Nous voulons que lorsque la variable temps est à 0, nous n'ayons plus de temps et la partie redémarre. Nous allons aller dans *Créer un nouvel événement > Système > comparer la variable > Si temps est plus petit ou égal à 0 > Système > Redémarrer la scène*

La partie redémarre à chaque fois que le chrono est atteint. Mais ce que l'on vient de faire vient de créer un bug qui risque de rendre le jeu injouable. Quand nous mourrons, la variable temps continue de descendre sans interruption. Il faut réinitialiser le chrono pour que le jeu fonctionne. Pour remettre le chrono à son état initial, il faut ajouter une action en dessous de *Redémarrer la scène : Réinitialiser les variables globales à leur défaut*

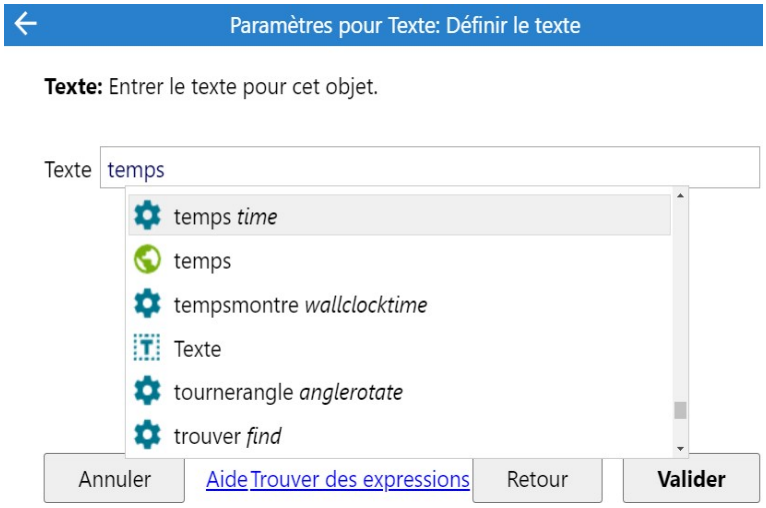


Désormais tout fonctionne parfaitement… Mais nous ne voyons pas le chrono affiché. Pour afficher le chrono, nous allons créer un texte que nous allons mettre sur notre écran (voir créer un texte).

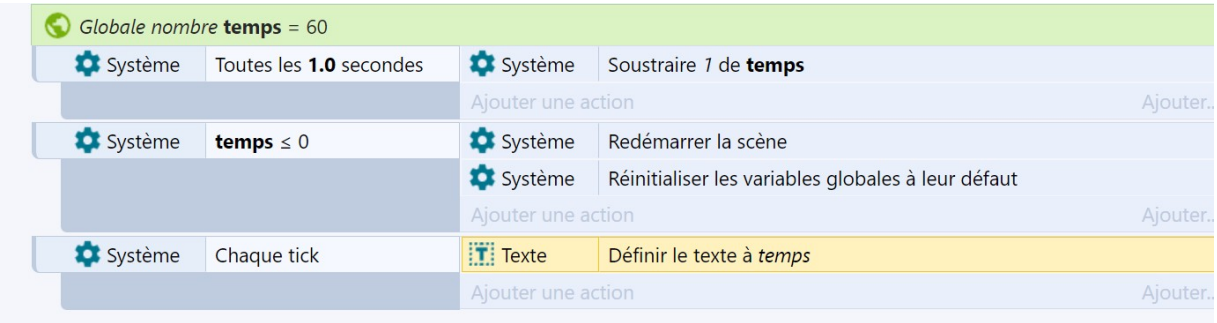
Nous allons ajouter un comportement au texte qui représente le score : grâce à **Ancre,** le texte va rester immobile sur notre écran.

La toute dernière étape désormais est de programmer le texte pour qu'il indique le chrono. Pour ce faire :

Créer un *nouvel événement > Système > chaque tick (à tout moment) > texte > définir le texte > temps*

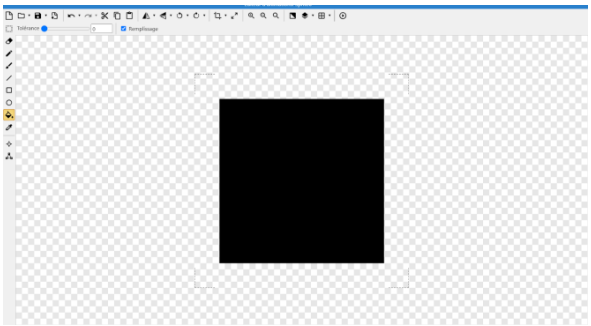


Pour terminer la programmation du **chrono**, il ne reste qu'appuyer sur temps et valider.



Allumer/éteindre la lumière (Difficulté : **)

Pour créer une mécanique de jeu sur la lumière, il est nécessaire créer un *sprite* complètement noir et de recouvrir tout le niveau de celui-ci.



Ensuite, il faut se rendre dans le menu "propriété" de l'objet et changer son opacité en 90%.

Le jeu devient bien plus sombre. Pour allumer la lumière, il faut créer la programmation suivante :

Ainsi, lorsque l'interrupteur est touché avec le personnage, le bloc noir disparait et toute la pièce s'éclaire.

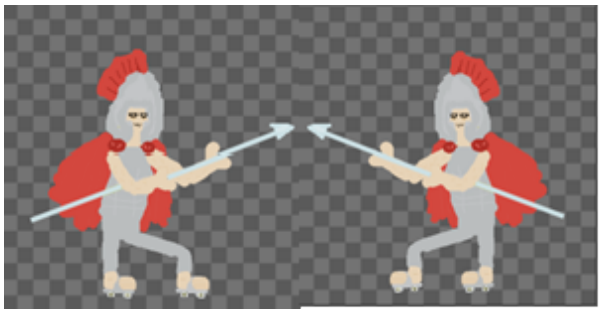
Il est également possible de créer un interrupteur on/off. Au lieu de détruire le bloc noir, il va disparaitre et apparaitre à chaque fois qu'il rentre en collision avec le personnage. Pour cela, il faut cliquer sur le bloc d'action "Détruire" et "Définir la visibilité" au sein de la feuille d'évènements. Ensuite, choisir "Basculer". Basculer signifie le changement d'à chaquecollision.

Faire se retourner le personnage lorsqu'il change de direction (Difficulté : **)

Actuellement le personnage se déplace toujours en regardant dans la même direction : la droite. Pour que le personnage se retourne à chaque fois qu'il change de direction, nous allons créer un nouveau code.

L'ajout du clavier est nécessaire pour cette partie. Pour ajouter le clavier au jeu, un simple clic droit (le même que pour ajouter un *sprite*) et chercher le **clavier** dans les objets.

Avec ce code, lorsque la touche gauche du clavier est appuyée, le personnage se retourne de manière symétrique. Avec la touche droite, le dessin reprend sa forme d'origine.

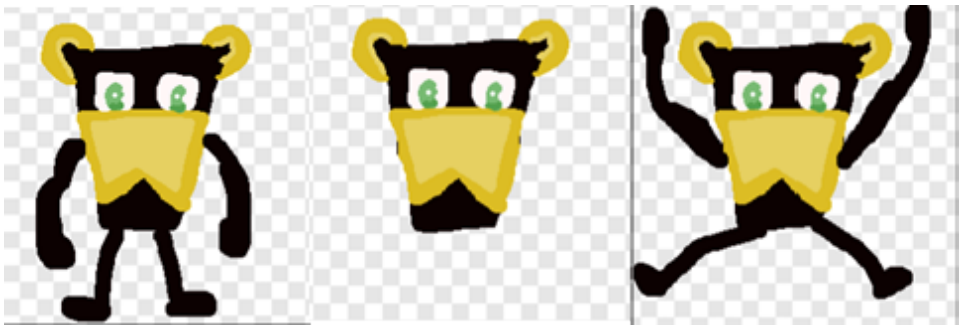


Créer une animation de saut (Difficulté : ***)

Pour créer une nouvelle animation[2], cela se passe dans l'éditeur de *sprite*. Dans l'éditeur de *sprite*, nous pouvons dupliquer notre personnage en appuyant sur ”Animation 1” avec un clic droit.

La nouvelle animation ressemble trait pour trait à la précédente. Elle a simplement été collée. Il va falloir modifier cette *animation 2*.

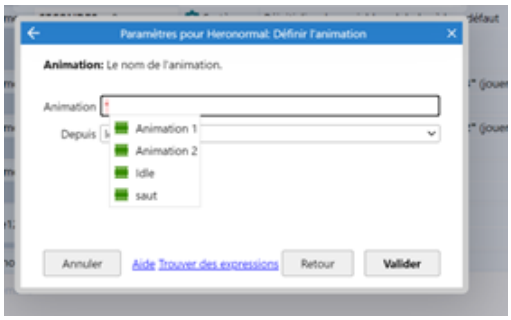
[2] Une animation dans un jeu vidéo est la retranscription d’un mouvement en dessin



Pour construire une animation de saut, il faut gommer les bras et les jambes et les redessiner ensuite en position de saut.

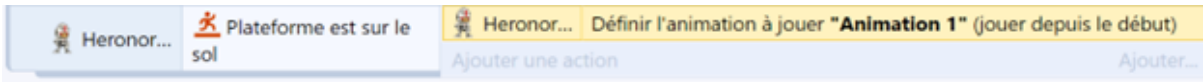
Après l'étape artistique vient l'étape technique : la programmation de l'animation. Dans la feuille d'événement, créez un nouvel événement : Quand mon personnage est en saut => Je veux que mon jeu affiche le dessin de mon personnage en saut. Dans Construct, cela donne

Pour définir l'animation, il faut sélectionner celle qu'on veut choisir et pour ça il faut écrire son **nom**. En cas d'oubli, il suffit d'enlever un des deux guillemets pour que Construct propose les animations déjà créées.



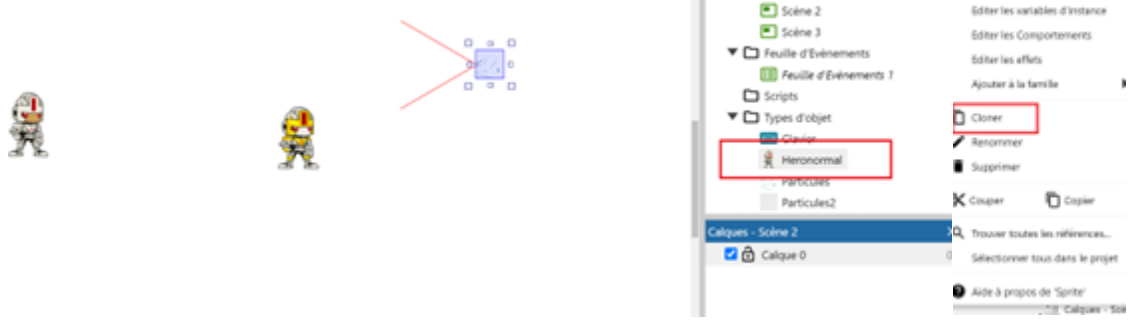
Actuellement, le personnage reste en position de saut en permanence. Pour le remettre en position de marche, il faut créer une nouvelle ligne de programmation suivant ce modèle.

Quand personnage sur le sol ... Alors personnage fait l'animation 1



Changer l'apparence et les propriétés du personnage jouable (Difficulté : *****[3])

Pour que le personnage puisse changer d'apparence, il est nécessaire d'en recréer un nouveau sur base de l'existant. Dans le menu Projet à droite de l'écran, il faut faire un clic droit sur le héros et choisir "cloner".



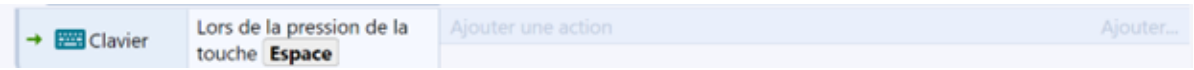
[3] Prérequis - animation de saut – personnage faire se retourner le personnage quand il change de direction)

Le personnage est désormais cloné. Dans l'éditeur *de sprite* de ce nouveau héros, on peut modifier son apparence. Dès lors, elle ne changera en rien l'apparence du premier personnage jouable créé.

Pour changer de costume, il faut créer un *sprite*/objet sur lequel le joueur va devoir appuyer pour sa transformation.

Une fois l'objet créé, il faut créer un nouvel événement : *quand notre personnage se trouve sur l'objet en question et que j'appuie sur la touche "espace" => notre personnage se transforme.*

La première étape, c'est "quand la touche espace est appuyée".



Attention à ne pas confondre "presser" = toucher une fois et "enfoncer" = laisser son doigt appuyé).

Pour faire en sorte que notre personnage se transforme, nous allons détruire notre personnage actuel (1) et faire apparaître le nouveau personnage (2).

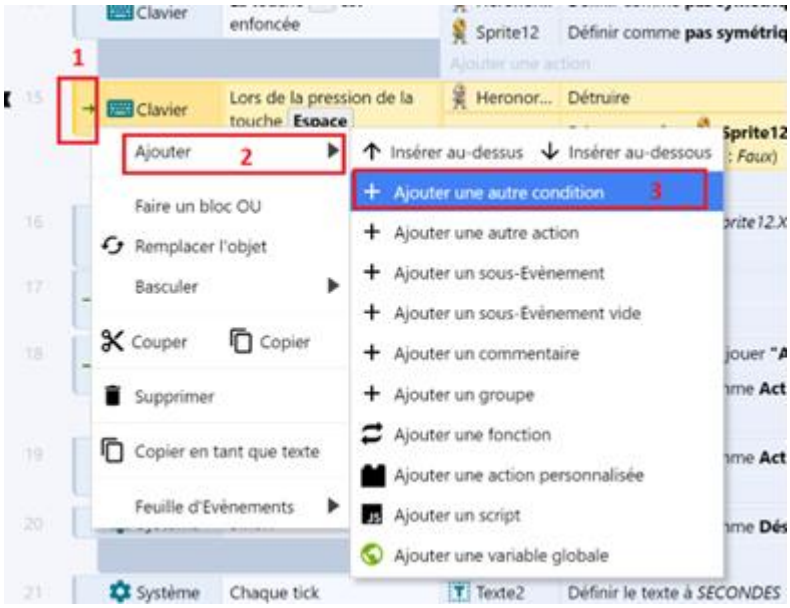
Pour faire en sorte que le personnage ne puisse uniquement se transformer qu'au contact d'un certain objet, nous allons de voir créer une double condition.

Pour se transformer, le personnage doit

- Se trouver sur l'objet qui permet la transformation
- Appuyer sur la touche "Espace"

Pour ajouter une double condition :

1) Faire un clic droit sur la **Flèche verte** pour ouvrir un nouveau menu
2) Dans ce menu, cliquer sur **Ajouter** pour ouvrir le dernier menu
3) Dans ce dernier menu, appuyer sur **Ajouter une autre condition**



La ligne de code à programmer dans cette double condition est :

La phrase "est en train de chevaucher" le *sprite*, signifie que le personnage **se trouve sur** un objet précis.

Le programme complet ressemble à ceci :

En forme **classique**, le personnage **meurt** quand il rentre en **collision** avec un ennemi.

En forme **dorée,** nous pouvons créer une **toute nouvelle ligne de code** pour dire que le personnage **vainc l'ennemi** s'il **rentre en collision** avec lui.

Les questions fréquemment posées en animation

Quand je lance mon jeu plus rien ne bouge / plus rien ne fonctionne.

- Option 1 : Lorsque l'on crée une nouvelle scène dans Construct, il est impératif de lier la feuille d'évènements 1, celle dans laquelle on a programmé toutes les précédentes actions. Si cette feuille d'évènement n'est pas sélectionnée, les éléments de notre jeu ne fonctionneront pas dans le nouveau niveau
- Option 2 : Le personnage est bloqué dans un objet solide et ne peut donc plus bouger. Il faut alors décaler notre personnage ou vérifier les boites de collisions

Mon personnage flotte ou mon personnage meurt alors que je ne touche pas l'ennemi.

Bien souvent, la boîte de collision a été mal réglée. Il s'agit d'un oubli récurrent.

Quand mon personnage se retourne, il se téléporte.

Il s'agit d'un problème lié au point d'image. Un point d'image est le point situé au centre absolu de notre élément. Il arrive qu'il soit mal réglé. Pour modifier le point d'image, il faut se rendre dans l'éditeur de *sprite*. Au-dessus du triangle de collision se trouve une icône en forme de cible. Une fois l'icône sélectionnée, il faut mettre son curseur au centre de son dessin et effectuer un clic gauche.

Mon jeu ne va pas au niveau suivant alors que la programmation a été effectuée.

Pour aller au niveau suivant, il faut impérativement qu'un deuxième niveau soit créé. Sans cela, le code ne renverra le joueur vers rien du tout.

Le fond de mon niveau a fait totalement disparaitre mes personnages.

Les personnages n'ont pas disparu. Le fond du niveau est simplement à l'avant-plan cachant tout le reste du niveau. Pour cela, il faut modifier l'élévation en z.

J'aimerais jouer à ce niveau avant celui-là.

Pour changer l'ordre des niveaux, il faut les mettre dans l'ordre désiré dans le menu projet.

Lorsque le chrono tombe à zéro, mon jeu ne fonctionne plus.

Il est impératif lorsque le chrono tombe à zéro de réinitialiser la variable temps. Cette variable étant à zéro va ordonner de redémarrer le jeu en boucle. Pour ce faire, il faut programmer l'action « réinitialiser les variables globales » après avoir redémarré sa partie.

Ressources utiles

Série de vidéos sur Construct 3 permettant un apprentissage complet et ludique (FR - 1h30)

https://www.youtube.com/playlist?list=PLDrnFpjF7H1VY5bY3bmp30t6LWNW5D_AQ

Guides et tutoriels sur le site officiel Construct 3 (FR et ANG)

https://www.construct.net/en/make-games/manuals/construct-3

https://www.construct.net/fr/tutorials?utm_campaign=C3Editor&utm_source=browser&utm_medium=r397&utm_term=StartPage2&utm_content=Tutorials&flang=2