Variations numériques sur un thème de Claude Monet

Comment l'artiste peut-il, à quelques centimètres de sa toile,
se rendre compte d'un effet à la fois précis et subtil
qu'on ne peut apprécier qu'avec un recul de plusieurs mètres ?
C'est le déconcertant mystère de son écran rétinien.
Georges Clémenceau

Notre intention avec cet article est d'apporter des éléments sur la programmation et de préciser en particulier les différents qualificatifs attribués au mot programmation. Il en est ainsi de la programmation que l'on dit être séquentielle par opposition à la programmation événementielle. Nous expliquerons aussi à la fin de cet article ce que l'on appelle la programmation orientée objet. Notre souhait – exprimé au début de cet article – est de rendre les choses les plus explicites possibles ; les multiples relectures que nous avons effectuées de ce texte ont eu pour unique objectif d'éviter tout langage « crypté » inaccessible aux personnes non initiées et au contraire d'avoir un langage « simple » et explicite dès la première lecture.

Ce magnifique tableau de Claude Monet est une œuvre créée pendant le courant impressionniste. Cette technique picturale est très liée à la conception d'une image numérique, elle est essentiellement basée sur le fait que la perception du monde qui nous entoure est parfois bien différente de la réalité.

Dame en blanc au jardin. Claude Monet
Musée de l'Hermitage, Saint-Pétersbourg
Source Wikipédia

Lorsque vous regardez un film, les images projetées successivement vous donnent l'impression d'une certaine fluidité ne vous permettant en rien de noter qu'il s'agit en vérité d'images projetées l'une après l'autre. Les explications de ce phénomène font débat : on a pensé jusqu'à présent qu'il s'agissait uniquement d'une persistance de l'information sur notre rétine mais aujourd'hui les neurosciences apportent des éléments tendant à prouver que c'est notre cerveau qui « re-fabrique » toutes les étapes intermédiaires entre deux images différentes permettant ainsi de lisser le passage de l'une à l'autre et de créer l'impression d'un mouvement parfaitement continu.

À l'identique de ce qu'il se passe pour le cinéma où nous récréons les chainons manquants entre deux images qui se succèdent, la perception d'une œuvre impressionniste est elle-aussi basée sur un phénomène qui – quant à lui - annihile les écarts. Pour comprendre ce phénomène, rappelons que la technique impressionniste consiste à représenter le monde réel à l'aide d'une juxtaposition de petites touches de peinture. Ces petites touches sont parfaitement décelables lorsque l'on se place suffisamment proche du tableau (on ne voit d'ailleurs – de près – que ces tâches plus ou moins

difformes) ; en revanche dès que l'on s'éloigne du tableau, l'image que l'on perçoit devient magiquement photographique.

Ce phénomène est expliqué par les scientifiques qui parlent de **pouvoir de résolution de l'œil** (pour faire simple) : si vous regardez **de près** ces deux points verts contigus, vous allez les voir nettement séparés ; en revanche, si vous les regardez **de loin**, il va vous sembler qu'ils se touchent.

Il en va de même d'une image numérique qui est à la base une juxtaposition de petites touches de couleurs. Tout est une question de densité de ces petites touches de couleurs sur une surface donnée : soit la densité est suffisante à ce que notre œil ne distingue pas la « supercherie » et dans ce cas cette image numérique va nous paraître identique à la réalité ; soit dans le cas contraire, nous allons en premier lieu voir ces petits carrés puis décrypter qu'il s'agit de la représentation d'un objet.

Pour que ces explications soient davantage explicites, prenons un exemple : voici à droite, la photographie numérique d'un fidèle voisin ; en vous plaçant devant cette photographie, vous ressentez une certaine réalité qui vous suffit à appréhender la scène. Or cette photographie est bien une image numérique - juxtaposition de petites touches de couleurs - preuve en est l'agrandissement de la tête de notre voisin que nous avons placé à gauche.

Revenons à notre tableau de Monet. Il serait réducteur de laisser penser qu'une telle œuvre engendre uniquement une perception visuelle, cette œuvre est là-aussi pour créer de l'émotion pour nous faire partager l'apaisement, la quiétude de ce beau jardin printanier fixée sous le pinceau de cet artiste qui réalise ici l'archétype de ce que l'homme peut faire de beau. Ce n'est plus une œuvre que l'on voit, une œuvre que l'on observe mais bientôt une œuvre que l'on regarde puis que l'on contemple, que l'on admire pour en faire un objet que l'on mémorise afin qu'il fasse partie de notre culture.

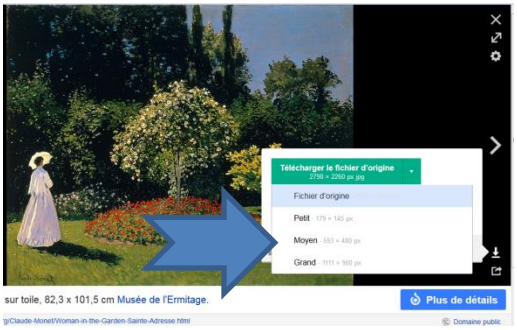
Comment faire pour que nos élèves arrivent à suivre ce cheminement intellectuel propice à les construire culturellement mais aussi à leur faire découvrir ces champs d'épanouissement ? En posant, la question nous pensons à ces élèves qui vivent dans un environnement rude, bien loin de la sérénité du musée de l'Hermitage.

Loin de prétendre apporter ici une solution prétentieuse, nous allons envisager un moyen de créer du lien entre un élève et une œuvre en utilisant un outil de programmation hautement performant : le logiciel Scratch. Il s'agit bien de faire en sorte que nos élèves arrivent à travailler sur **un support qui va leur être donné comme étant - au départ - un support comme un autre, puis – par imprégnation - à les conduire vers une perception de la joliesse de ce tableau**.

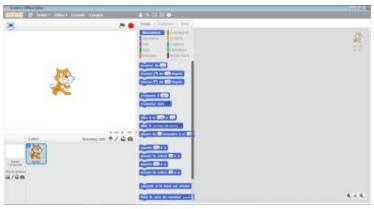
Téléchargement d'une image numérique du tableau de Monet

Nous nous excusons ici de retomber dans des descriptions techniques mais il nous faut sauvegarder notre image pour la faire réapparaître par la suite dans notre environnement Scratch. En tapant « Dame en blanc au jardin » sur un moteur de recherche, nous accédons à la page *Wikipédia* dédiée à notre tableau.

En cliquant sur l'image du tableau, celle-ci apparaît en plein écran avec au-dessous une indication comme quoi elle fait bien partie du domaine public (symbole ci-dessus à gauche). Nous optons pour le téléchargement du fichier nommé *Moyen* qui est une image de **définition** 593x480, nous enregistrons cette image dans le dossier des éléments téléchargés. Ce qui veut dire que cette image comporte au total 284 640 pixels répartis dans un tableau - on parle aussi de matrice - en 480 colonnes de 593 pixels.



Nous allons travailler à présent avec le logiciel Scratch. Le téléchargement de ce logiciel libre peut se faire à partir du lien suivant : https://scratch.mit.edu/scratch_1.4/ .
Une fois le logiciel lancé, il apparaît l'écran ci-contre :



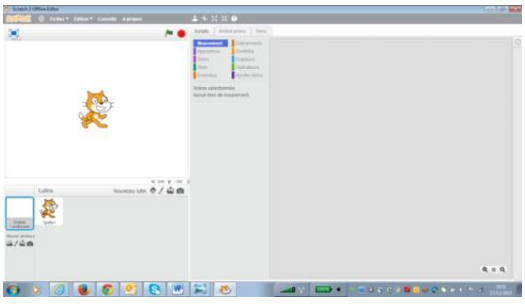
Programme n°1 : déplacement aléatoire d'un papillon dans le jardin de Claude Monet.

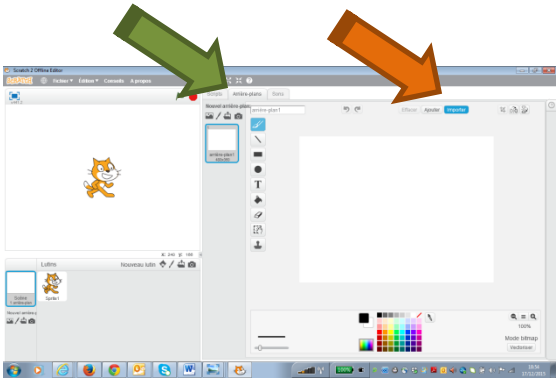
Un exemple de programme séquentiel.

On appelle **programme séquentiel** par opposition à programme événementiel un programme qui lorsqu'il est lancé s'exécute en entier. Autrement dit, ce programme fonctionne « tête baissée » et ne va en rien prendre en compte ce qu'il se passe que ce soit du côté de son déroulement propre ou du côté de l'utilisateur pour lequel toute intervention de sa part n'est pas prévue dans le programme.

Par souci d'efficacité, nous allons construire le programme en trois étapes.

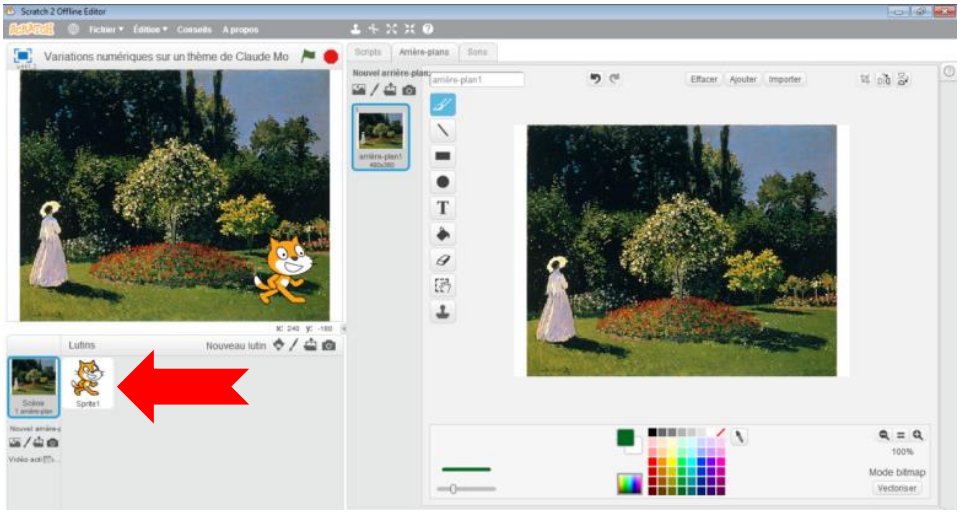
1ère étape : plaçons le tableau de Claude Monet en arrière-plan.
Nous accédons au registre des arrière-plans en cliquant sur la zone pointée par la flèche bleue ci-contre.





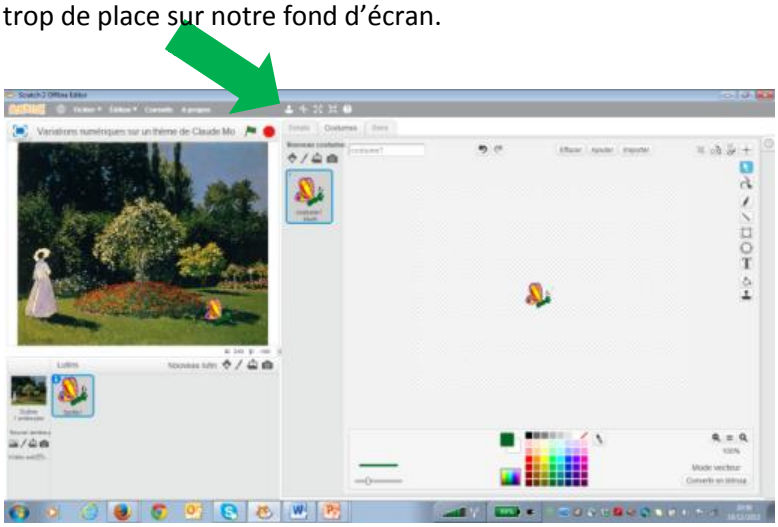
Nous rapatrions l'image du tableau de Monet que nous avons conservée dans le dossier des éléments téléchargés. Pour ce faire, on clique sur l'onglet arrière-plan (flèche verte), puis sur le bouton Importer (flèche orange) et l'on va tout simplement chercher le fichier à l'aide de l'explorateur qui s'ouvre.

Nous arrivons ainsi à obtenir notre magnifique fond d'écran.



2ième étape : choisissons parmi les personnages le papillon.

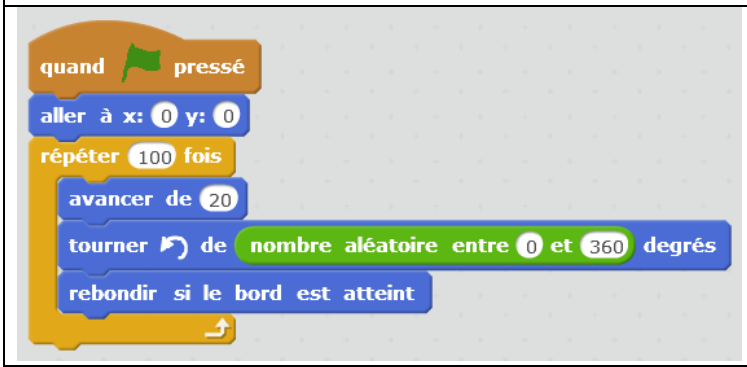
Nous cliquons sur le chat (flèche rouge **de l'écran ci-dessus**) dans la zone réservée aux lutins. Nous activons la fenêtre des costumes en cliquant sur l'onglet puis effaçons le chat pour le remplacer le papillon numéro 3 (flèche orange ci-contre).Nous réduisons les dimensions de ce papillon afin qu'il n'occupe pas trop de place sur notre fond d'écran.



À ce stade des opérations, nous obtenons l'écran ci-contre avec le papillon qui est pour l'instant immobile sur notre tableau.

3ième étape : écrivons à présent le programme proprement dit (que l'on appelle un script).

Pour ce faire, nous faisons afficher la fenêtre des scripts du papillon en cliquant sur l'onglet correspondant (flèche verte de l'écran de la deuxième étape). Le script que nous allons écrire va permettre à notre papillon de décrire un mouvement pseudo-aléatoire ; pour ce faire, nous construisons brique par brique le programme suivant :

Script	Explications
	Presser sur le drapeau vert permet de lancer le programme. Le papillon est placé au départ au centre de l'écran. Puis l'on répète 100 fois la boucle qui consiste « à faire avancer le papillon de 20 pas puis à le faire tourner de façon aléatoire avec un rebond s'il atteint éventuellement le bord de l'écran ».

Le mode plein écran donne toute son envergure au déroulement de la scène sur notre beau tableau.

Programme n°2 : « Essaie de cliquer sur le papillon ! »

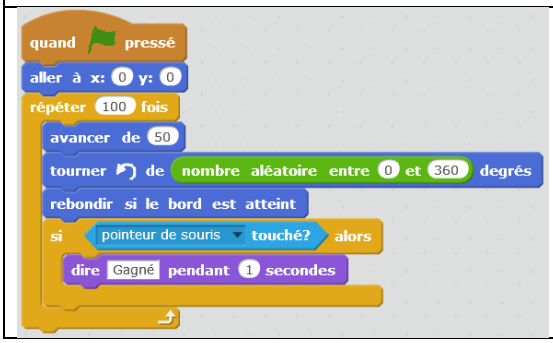
D'un programme séquentiel vers un programme événementiel.

Le programme n°1 est un programme séquentiel autrement dit : rien ne vient modifier son déroulement. Nous allons apporter un nouvel élément à ce programme afin qu'il prenne en compte l'intervention de l'utilisateur. Nous obtiendrons ainsi **un programme événementiel**.

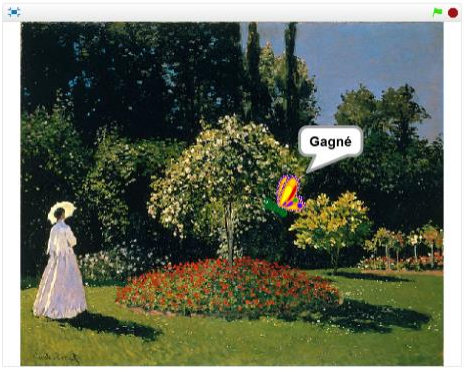
Avec le programme n°1, le papillon décrit un mouvement aléatoire de 100 déplacemens effectués sur le tableau de Claude Monet. Nous activons à présent ce que l'on appelle **un capteur**, ce capteur va

être positionné sur le pointeur de la souris et va donc permettre de relever (avec précision) la position de ce pointeur que l'utilisateur peut faire se déplacer sur l'écran.

Le programme n°2 consiste d'une part à faire effectuer à notre papillon un mouvement alétoire analogue à celui du programme n°1 et d'autre part à écrire le message « Gagné » à chaque fois que les positions du pointeur et du papillon seront identiques.

Script	Explications
	Nous reprenons le programme n°1 que nous allons simplement modifier sur les deux points suivants : 1) Nous réglons à 50 le nombre de pas réalisés par le papillon à chacun de ses déplacements (cela rend le jeu un peu plus difficile) ; 2) L'instruction dire « *Gagné* » pendant une seconde est réalisée uniquement si le papillon touche le pointeur de la souris.

Le passage en mode plein écran permet de cacher le script du programme et de ne conserver que notre tableau interactif.

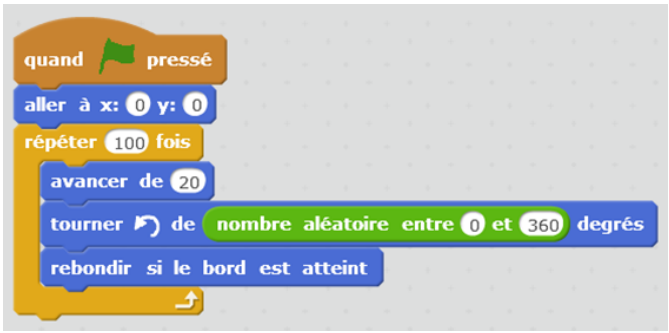


Programme n°3 : « Alerte dès que le cœur de la dame s'emballe ».

De l'objet à son script

La programmation orientée objet est le terme général qui caractèrise un certain type de programmation dont fait partie la programmation sous Scratch.

Pour bien comprendre, ce qu'est une programmation orientée, reprenons le programme n°1 (ci-contre) qui consiste rappelons-le à faire se déplacer aléatoirement notre papillon. Ce programme a été crée une fois que nous avions défini notre papillon (l'objet) et ce corps d'instructions (le programme) est interdépendant de l'objet papillon.



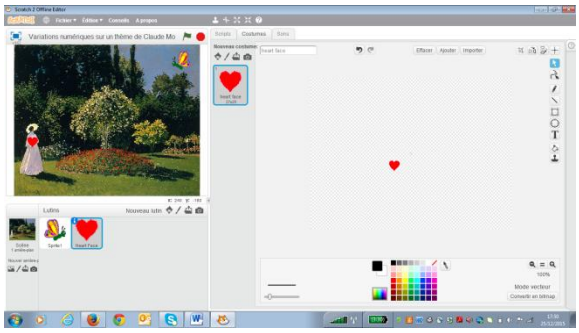
L'avantage de ce type de programmation par rapport à d'autre système est :

- d'une part qu'il simplifie la programmation (on fait la liste des objets dont on a besoin puis on programme une suite d'instructions propre à chacun d'entre eux) ;
- d'autre part que cela va **permettre aux objets de communiquer et d'interagir entre eux**. Afin d'illustrer cette possibilité, nous allons créer un nouveau programme pour le moins romantique qui consiste à faire intervenir le papillon dès que lui parvient un signal émis par le cœur de la dame.

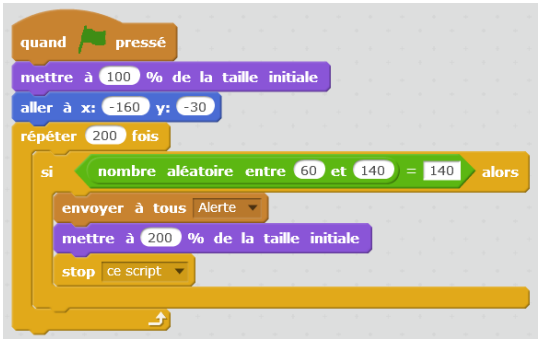
Par souci d'efficacité, nous partageons la construction de ce nouveau programme en trois étapes.

1ère étape : un simulateur de relevé de rythme cardiaque.

Nous partons du programme n°1 auquel nous adjoignons un nouveau lutin : un petit cœur que nous plaçons sur la dame en blanc.



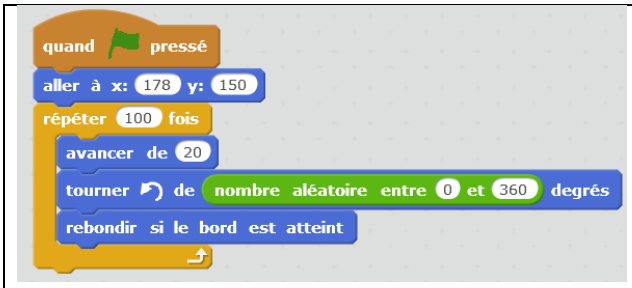
Pour ce nouveau lutin, nous construisons le script suivant :



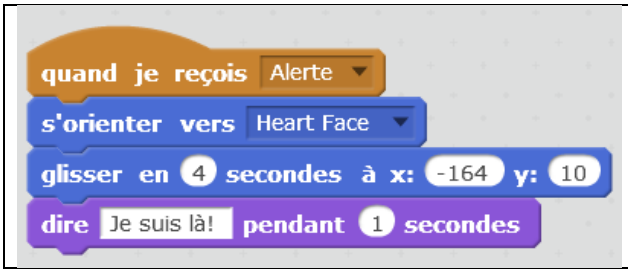
Le lancement du programme s'effectue en cliquant sur le drapeau vert.
Le cœur est remis à sa taille initiale puis il est placé sur la Dame.
On répète (au maximum) 200 fois la suite d'instructions :
« Le rythme cardiaque est simulé par un nombre aléatoire compris entre 60 et 140 pulsations à la minute. Si ce rythme cardiaque atteint la valeur 140, un alerte est émise et l'aire du petit cœur quadruple. On arrête alors le simulateur.

2ième étape : modification du script du petit papillon.

Plaçons-nous à présent sur le registre qui concerne le papillon. Le script va contenir deux parties. La première partie est identique à celle du programme n°1 et consiste à faire se déplacer le papillon de façon pseudo-aléatoire.



Quand on clique sur le drapeur vert, le papillon se déplace en haut et à droite de l'écran puis il parcourt 100 fois la boucle qui consiste à :
- avancer de 20 pas ;
- tourner d'un angle choisi aléatoirement ;
- rebondir éventuellement sur le bord.



Quand le papillon reçoit l'alerte, il s'oriente vers le cœur pluis glisse en 4 secondes pour s'arrêter au dessus de la damme.
Il apparaît ensuite le message « Je suis là » pendant une seconde.

Le lancement du programme peut se faire en plein écran, avec un peu de chance le mouvement - au départ aléatoire du papillon – s'oriente vers la dame dès le signal émis.

La programmation orientée objet est donc particulièrement propice à simplifier la programmation événementielle. Au délà de l'exemples ludique que nous venons de vous donner, nous imaginons les champs précieux d'interventions de ce type de programmation : chirurgie assistée par ordinateur, pilotage à distance d'un robot, dépannage en vol d'un avion, etc ; autant de domaines dans lesquels des objets indépendants physiquement opérent de façon dépendante informatiquement.

L'idée que ces élèves se souviennent de ce tableau en l'accrochant fermement dans leur environnement culturel reste bien entendu le souhait affirmé du modeste auteur de cet article.

Jean-Alain Roddier
IA-IPR de mathématiques
Académie de Clermont-Ferrand