LYNX™

LCSI

SIMULATION DE COURSE
GUIDE DES ENSEIGNANTS

Simulation de course
Guide des enseignants

Aperçu du projet

Ce projet est constitué de six leçons de trente minutes. Il consolide certains concepts de base du codage et présente d'autres concepts plus avancés, comme la détection de collision, les processus parallèles et le contrôle de multiples tortues.

Les étudiants utiliseront des primitives (vocabulaire intégré de Lynx), des procédures (vocabulaire ajouté par le codeur) ainsi que des objets de contrôle interactifs (comme des boutons), des éléments graphiques et du son pour créer une simulation de course.

En créant cette simulation, les étudiants exploreront des concepts mathématiques comme le hasard et son lien avec les probabilités.

Compétences requises

Les étudiants doivent avoir déjà complété le projet : *Récit interactif*.

Liens avec le programme scolaire

Dans ce projet, les étudiants acquerront les compétences suivantes :

- Décrire le sujet, l'objectif et l'auditoire pour un document multimédia qu'ils vont créer.
- Construire un document électronique significatif en combinant plusieurs médias - images, sons, éléments graphiques et texte.
- Planifier et rédiger de simples programmes informatiques, appliquer des concepts fondamentaux de programmation incluant des commentaires internes révélateurs.
- Développer des solutions créatives pour différents types de problèmes qui impliquent l'élaboration d'un discours « mathématique ».
- Faire des liens entre certains concepts mathématiques et des procédures, et entre les mathématiques et des événements de la vie de tous les jours comme les activités sportives.
- Communiquer des concepts mathématiques verbalement et visuellement.

Simulation de course
Guide des enseignants

Avant de commencer

1. Avant d'entreprendre ce projet, vos étudiants devraient :
   - Avoir créé leur compte individuel permanent sur lynxcoding.club.
   - Avoir répondu à l'invitation à se joindre à un groupe (école - club).
   - Avoir complété les activités du projet de niveau 1 : *Récit interactif*.
2. Regardez le projet donné en exemple *À vos marques!* à l'adresse https://lynxcoding.club/ (cliquez sur Tous les projets et ouvrez le dossier Mode débutant) avec vos étudiants pour qu'ils comprennent comment il fonctionne et comment il a été créé. Passez en revue les techniques suivantes :
   - Comment utiliser des cliparts pour créer un arrière-plan.
   - Comment ajouter des tortues (sprites).
   - Comment utiliser des cliparts pour donner des formes aux tortues.
   - Comment déplacer la tortue à l'aide de commandes.

Ensuite, discutez des caractéristiques du projet. Incitez les étudiants à réfléchir aux points suivants :

- Comment faire pour ramener toutes les tortues à un point de départ prédéterminé?
- Comment déclencher la course?
- Comment faire en sorte qu'une instruction unique pour toutes les tortues donne des résultats variables?
- Comment un coureur sait-il qu'il a franchi le fil d'arrivée?
- Que se passe-t-il lorsqu'un coureur franchit la ligne d'arrivée? Quelles conséquences ou réactions peuvent être programmées?

Discutez avec vos étudiants de l'importance de la planification. Suggérez aux étudiants de mettre sur papier une planification sous forme de diagramme ou de liste d'étapes.

DONNEZ UN **NOUVEAU NOM** AU PROJET ET **ENREGISTREZ**-LE : Dès les premières minutes d'un projet, assurez-vous que les étudiants donnent un nouveau nom au fichier modèle. Il suffit de cliquer sur le nom actuel, tout en haut du panneau de procédures. Tapez un nouveau nom, puis cliquez sur l'icône d'enregistrement (une flèche vers le nuage).

Rappelez aux étudiants que Lynx n'enregistre PAS automatiquement le travail; ils doivent enregistrer leur projet souvent. L'enregistrement est facile et rapide, il suffit d'exécuter la commande **Enregistrer** dans le menu **Fichier** (disquette).

Simulation de course
Guide des enseignants

Cibles d'apprentissage et liste de vérification

Les projets devraient illustrer la maîtrise, par les étudiants, des compétences suivantes. Chaque compétence devrait contribuer, d'une certaine façon, à la conception de cette simulation de course.

√	COMPÉTENCE
	Utiliser des commandes de base pour déplacer la tortue et créer une animation simple.
	Utiliser des cliparts fournis et des cliparts importés pour créer un arrière-plan, estamper des images et changer la forme de la tortue, comprendre le concept de calques superposés.
	Créer, modifier et déboguer de simples procédures créées par l'utilisateur.
	Utiliser des commandes pour créer des animations plus complexes, explorer le concept de processus parallèles
	Ajouter des éléments interactifs sous la forme de boutons et de tortues programmées
	Ajouter des éléments multimédias comme des effets sonores ou de la musique.
	Utiliser la détection de collision pour déclencher un événement dans certaines conditions (par exemple, dans ce projet, lorsqu'un coureur franchit la ligne d'arrivée).
	Utiliser le type de donnée « mot » pour créer un message personnalisé à la fin de la course.
	Utiliser les coordonnées de la zone de travail (la page) pour placer des objets (tortues) à des endroits précis.

Nouveaux concepts et nouvelles commandes Lynx

1. Préparer l'arrière-plan
   ```
   fixecouleur, fcoul
   remplis
   ```
2. Travailler avec plusieurs tortues
   ```
   clone
   demande
   chacun
   ```
3. Hasard
   ```
   hasard
   ```
4. Processus parallèles
   ```
   toujours
   cliqueactive
   cliquedésactive
   arrêtetout
   ```
5. Détection de collision
   ```
   Événement « si touché »
   ```
6. Type de donnée : mots
   ```
   mot
   annonce
   dis
   ```
7. Procédure de remise à zéro, boutons, éléments de l'IU
8. Coordonnées cartésiennes
   ```
   fixex
   fixey
   pos
   ```

Erreurs fréquentes et débogage

Le débogage (correction des dysfonctionnements d'un logiciel) est une excellente activité pour développer les compétences de résolution de problèmes. Lynx propose plusieurs outils de débogage afin d'aider les étudiants qui apprennent à coder, incluant la saisie automatique, les infobulles, les messages d'erreur et, évidemment, un système d'aide en ligne.

Pour commencer, il serait opportun d'identifier les types de bogues les plus fréquents que les débutants doivent affronter. La plupart des erreurs, à ce niveau, seront du premier type.

1. Erreurs de frappe ou de syntaxe

- Oublier de mettre une espace entre une commande et sa donnée, par exemple `av30` au lieu de `av 30`.
- Erreurs de saisie au clavier, comme `ac 20` au lieu de `av 20`.
- Utiliser le mauvais type de donnée ou de ponctuation, par exemple `répète 4{av 1 attends 1}` au lieu de `répète 4 [av 1 attends 1]`.

  Lorsqu'une erreur survient, Lynx retourne un message qui décrit l'erreur. Il est fréquent que les étudiants négligent de lire de tels messages, mais comme le message décrit la nature de l'erreur et l'endroit où elle est survenue, il faut insister et les convaincre de considérer ces messages comme une assistance précieuse. Par exemple, si un étudiant exécute l'instruction `av30`, Lynx affichera le message : `Je ne sais pas comment av30`. Lynx interprète `av30` comme une commande qui ne fait pas partie du vocabulaire intégré de Lynx (les primitives), ni des procédures (définies par l'utilisateur). Contrairement aux humains, Lynx ne peut « interpréter » ni « essayer de comprendre » l'intention réelle de l'étudiant, il ne peut qu'exécuter les « ordres compatibles ».

- Oublier d'inscrire la ligne `fin` à la fin d'une procédure.

  Il arrive que les étudiants oublient de mettre le mot `fin` pour terminer une procédure. La procédure en question fonctionnera normalement, mais les procédures qui suivent ne seront pas reconnues. Si une procédure semble inconnue alors qu'elle est bien présente dans le panneau de procédures, si Lynx affiche le message « `je ne sais pas comment…` » ou si une procédure n'apparaît pas dans la boîte de dialogue d'un bouton, vérifiez la présence du mot `fin`, seul sur sa ligne, après chaque procédure.

2. Erreurs de logique

- Erreurs de séquence - Les commandes doivent être exécutées dans un ordre spécifique, selon le résultat désiré. Le concept de séquence est présent dans la vie de tous les jours, comme c'est le cas pour les chaussettes et les chaussures. La pensée algorithmique est une

compétence essentielle en programmation.

Cela peut sembler simple à première vue, mais il est plus difficile de suivre la séquence des événements lorsque les programmes deviennent plus complexes. Dans ce projet, la logique linéaire est assez simple, mais il faut en profiter pour conscientiser les étudiants au fait qu'ils doivent être capables de « verbaliser » une séquence d'événements avant de la programmer.

- Le flux d'information entre les différentes commandes incluant celles qui renvoient (ou retournent) de l'information qui doit être utilisée par une autre commande. Ce type d'erreur peut survenir lorsque les étudiants créent des instructions plus complexes comportant plusieurs primitives.

Notions à discuter

Prenez le temps de discuter des points suivants :

- Pouvez-vous utiliser la même procédure de course pour tous les coureurs ou est-ce que cela fera en sorte que le résultat de la course sera prévisible? Est-ce que les valeurs générées au hasard finiront pas s'équilibrer après une longue période?
- Pourquoi est-il important de donner aux tortues des noms significatifs? Différents événements peuvent être programmés lorsqu'un coureur franchit la ligne d'arrivée; par exemple, la course prend fin et le gagnant est annoncé, ou la course continue jusqu'à ce que tous les coureurs aient franchi la ligne d'arrivée. Si on désire annoncer le nom du gagnant, l'annonce doit être unique, car le simple fait d'annoncer « J'ai gagné » ne sera pas suffisant si deux coureurs arrivent presque simultanément. Le nom de la tortue, dans ce cas, est très utile.
- La procédure `fincourse` utilise les commandes `annonce` et `dis`.

`Annonce` et `dis` utilisent une donnée qui, dans ce projet, est une séquence de caractères ou un mot. C'est ainsi que l'on nomme une séquence de caractères dans Lynx. Il existe plusieurs façons de créer ou de désigner un mot :

- Le guillemet simple (`'`) placé aux deux extrémités d'une séquence de caractères: `'Bonjour à tous'` est une séquence de caractères (ou, pour Lynx, un mot) comportant 14 éléments (dans cet exemple, le 8e et le 10e élément sont des espaces). Par exemple, tapez ceci dans le centre de commande :

```
montre 'Bonjour à tous'
Bonjour à tous
```

- Le guillemet double (") au début d'une séquence de caractères. On peut utiliser ce marqueur lorsque le texte ne comporte aucun symbole spécial (espace, crochet, point-vigule, etc). "Hélène est une chaîne de caractères (un mot) qui contient 6 lettres.

  ```
  montre "Hélène
  Hélène
  ```

  Les principales primitives qui traitent les mots sont mot, compte, item, mot? Dans le projet donné en exemple, la procédure terminé utilise la primitive mot qui crée un mot en combinant ses données. Ainsi :

  ```
  mot 'Le gagnant est ' "bibi
  ```

  … retourne le « long mot » Le gagnant est bibi à une commande (comme montre ou dis) qui utilise ce long mot comme donnée.

  Par exemple, tape ceci dans le centre de commande :

  ```
  montre mot 'Le gagnant est ' "bibi
  Le gagnant est bibi
  ```

- Incitez les étudiant à soumettre leurs idées sur la façon dont ils pourraient améliorer leurs projets.

Remarques supplémentaires

Chaque leçon devrait prendre environ trente minutes, mais certains étudiants auront un peu de temps pour explorer ce que l'environnement a à offrir.

Pour économiser temps et énergie dans la recherche de cliparts, ouvrez le projet Course Modèle qui contient quelques cliparts préfabriqués représentant des personnes et des animaux qui se déplacent de gauche à droite, ainsi qu'un fil d'arrivée. Les modèles de projets se trouvent dans la section Tous les projets de https://lynxcoding.club. Regardez dans le répertoire Modèles.

Insistez auprès de vos étudiants pour qu'ils prennent le temps de planifier leur projet. Ça les aidera à démarrer, et ils économiseront du temps et des efforts en cours de développement.

En même temps, dites aux étudiants qu'il est tout à fait normal de modifier un plan alors que le projet est en cours de développement. Les plans ne sont pas coulés dans le béton. Il est opportun de réévaluer un plan à des moments stratégiques pour déterminer s'il doit être ajusté en fonction de nouvelles idées ou de compétences nouvellement acquises.