

✂ teachers
learning
code



GUIDE POUR LES ENSEIGNANTS

Formez un club de code ou intégrez la programmation à votre classe.



Un programme éducatif de

CANADA EN PROGRAMMATION

TABLE DES MATIÈRES

Bienvenue	2
Qu'est-ce que Canada en programmation?	3
Teachers Learning Code	4
Pourquoi enseigner la programmation?	5

Objectifs d'apprentissage	7
Comment organiser un cours	9
Créer un climat favorable	11

Concepts de programmation clés	13
Pour bien commencer avec Scratch	14
Introduction à Scratch	15
Notions de base de l'éditeur Scratch	16
Concepts de base suite	17
Débogage dans Scratch	18
Gestion des imprévus	19

Autres ressources	21
À imprimer	22

BIENVENUE!

Nous souhaitons inspirer les Canadiens en leur enseignant la programmation afin de leur permettre de devenir des citoyens numériques. Ainsi, ils auront les outils pour comprendre notre nation et le monde entier et pourront se servir de ceux-ci pour participer à la société et l'influencer en tant que créateurs et innovateurs de la technologie.

Le programme Teachers Learning Code est conçu pour les enseignants du primaire et du secondaire ayant peu ou aucune expérience en programmation et souhaitant initier leurs élèves à la pensée informatique.

Grâce à nos nombreuses années d'expérience, nous avons appris quelques trucs pour enseigner la programmation. Dans ce guide pratique, nous partagerons avec vous quelques-uns de nos trucs pour commencer, des ressources pour vous initier à la programmation ainsi que des activités tant faciles à comprendre qu'à organiser pour enseigner la programmation à la prochaine génération de technologues du Canada.

Ce guide est votre point de départ : il comprend des ressources, des leçons, des exemples et des liens qui vous permettront de bien participer. Le contenu de votre cours est à votre discrétion : vous avez carte blanche!

Nous vous encourageons à utiliser le contenu de ce guide à votre guise en modifiant les leçons ou en créant de nouvelles que vous pourriez partager avec notre communauté.

La technologie se prête bien à la création. Amusez-vous!

QU'EST-CE QUE CANADA EN PROGRAMMATION?

Canada en Programmation est une initiative sans but lucratif qui se consacre au développement de la littératie numérique de tous les Canadiens.

Notre mission est de concevoir et d'organiser des programmes d'enseignement de la technologie et de faire des partenariats avec des agents du milieu afin que nos citoyens soient préparés à vivre pleinement dans une société numérique.

L'événement Canada en Programmation est une initiative de l'équipe derrière l'organisme Ladies Learning Code, qui donne des ateliers de programmation partout au pays depuis 2011.



✧ girls
learning
code

✧ kids
learning
code

✧ ladies
learning
code

✧ teens
learning
code

✧ teachers
learning
code

canadalearningcode.ca

TEACHERS LEARNING CODE

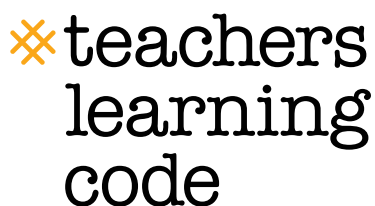
UTILISER LA TECHNOLOGIE POUR CHANGER LE MONDE GRÂCE AU TRAVAIL D'ÉQUIPE, À LA CRÉATIVITÉ, ET, BIEN SÛR, AU CODE.

Il s'agit d'un programme parrainé par la Banque Scotia visant à inspirer les enfants à ne pas seulement consommer la technologie, mais aussi à la bâtir en faisant des activités et des exercices de programmation.

Teachers Learning Code enseigne une méthode d'enseignement de la programmation dont le contenu peut être simplifié ou enrichi selon les besoins des apprenants. Tous les organisateurs de programmes pour les jeunes ainsi que les enseignants, qu'ils aient une formation technique ou non, sont en mesure de donner nos cours dans les écoles et les groupes communautaires.

Le contenu de ce guide pratique et les exercices de programmation sont conçus pour les élèves du primaire, mais les apprenants de tous les âges peuvent utiliser nos ressources. Elles contiennent toutes les caractéristiques qui sont à l'origine du succès de nos programmes.

teacherslearningcode.com



POURQUOI ENSEIGNER LA PROGRAMMATION?

La technologie est partout, et elle est là pour de bon. Les sciences, la technologie, le génie et les mathématiques, un ensemble de disciplines connues sous l'abréviation STEM en anglais, sont à la clé de notre avenir, particulièrement lorsqu'elles sont intégrées à d'autres disciplines. Nous voulons donner aux jeunes Canadiens les compétences essentielles dont ils ont besoin pour vivre dans notre société actuelle et s'épanouir dans celle du futur.

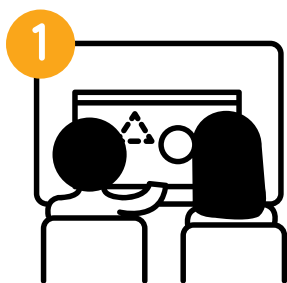
Mais pourquoi enseigner la programmation?

Selon le Conseil des technologies de l'information et des communications du Canada, il y aura une pénurie d'environ 200 000 travailleurs dans le domaine des TIC d'ici 2020. Pour nos jeunes, l'apprentissage de la programmation peut mener à une carrière gratifiante et payante. Toutefois, l'enseignement de la programmation aux enfants n'est pas qu'une façon de leur faire comprendre la technologie et d'obtenir un emploi stable. Fondamentalement, elle améliore la capacité à résoudre des problèmes et à penser et elle permet de comprendre le monde autour de soi.

Nous croyons qu'il est important d'apprendre à programmer aux jeunes pour les raisons suivantes :

- **La programmation est un super-pouvoir.** L'apprentissage de la programmation permet aux enfants de ne pas seulement consommer la technologie, mais aussi de la créer, que ce soit sous forme de jeux vidéo, de sites Web, de robots et d'autres supports.
- **La programmation permet aux enfants de penser de nouvelles manières.** Son apprentissage aide les enfants à développer des compétences essentielles applicables dans tous les domaines. Grâce à elle, ils peuvent résoudre des problèmes touchant à la programmation ou à d'autres sujets en innovant.
- **La programmation aide les enfants à comprendre le monde qui les entoure.** Si l'on enseigne la biologie et les mathématiques pour mieux comprendre le monde, il va donc de soi que l'on enseigne les rudiments de la communication avec les ordinateurs et l'interaction avec eux.
- **La programmation peut changer le monde.** En enseignant aux enfants comment utiliser la technologie comme instrument de création, nous leur donnons les outils pour créer des solutions à des problèmes de tous les jours.
- **La programmation est amusante!** Nous voulons que les enfants soient excités et satisfaits d'avoir créé quelque chose eux-mêmes.

CONSEILS IMPORTANTS POUR ENSEIGNER LA PROGRAMMATION



1 Assurez-vous d'être à l'aise avec l'outil, mais ne vous inquiétez pas si vous n'êtes pas un expert : laissez les apprenants s'entraider!

2

Soyez créatif, n'ayez pas peur de faire des erreurs, et surtout, amusez-vous!

Ayez une vision claire de ce que vous souhaitez accomplir et trouvez un collègue qui aimerait enseigner à vos côtés. C'est un apprentissage véritablement intégré à l'enseignement!

3

4

N'oubliez pas que l'apprentissage est progressif. Voyez les problèmes de technologie comme l'occasion d'apprendre. C'est un excellent exemple du concept populaire de conception que l'on appelle le « débogage »!



5 Invitez des experts, des conférenciers et des bénévoles de votre région qui donneront de petites leçons et offriront un soutien supplémentaire à l'apprentissage de la programmation.

6 Laissez la créativité des apprenants indiquer la direction que prendra votre cours. Que souhaitent-ils apprendre et approfondir?

OBJECTIFS D'APPRENTISSAGE

Avant tout, il faut voir la programmation comme une manière de résoudre un problème plutôt que comme l'apprentissage d'une langue ou d'un outil en particulier, comme le Java ou Scratch. Les langages de programmation évoluent et changent constamment. Toutefois, l'essentiel de la méthode de résolution de problème à l'aide d'un ordinateur reste le même.

La pensée informatique, le processus de résolution de problème, peut être enseignée sans même toucher à un langage de programmation en particulier. Le code n'est que l'outil facilitant la résolution de problème d'une certaine manière. Un fait important dont il faut se souvenir en enseignant : il n'est pas nécessaire de connaître la syntaxe d'un langage par cœur pour l'enseigner. Il s'agit de comprendre la logique de la résolution de problème, une logique que vous avez déjà très probablement développée dans le cadre de votre parcours comme enseignant. Le temps est venu de faire un lien entre cette logique et l'enseignement de la programmation.

En enseignant la programmation, nous enseignons aussi:

La pensée informatique

- Le raisonnement logique
- La pensée critique
- La reconnaissance de formes
- La résolution de problèmes complexes en les divisant en petites parties
- Le débogage de problèmes
- Le développement d'idées, du concept initial au projet final

Des concepts reliés aux ordinateurs

- Les programmes informatiques sont créés par des humains et indiquent exactement quoi faire à l'ordinateur.
- Les ordinateurs ne sont pas si intelligents ou intuitifs, car ils ne comprennent pas les choses comme le font les humains. Avec eux, il faut être précis en leur donnant des instructions à suivre.
- Il ne faut pas être un expert dans le domaine pour programmer. Il s'agit de penser clairement et avec soin.



La créativité et la collaboration

- La démarche de conception
- L'innovation

La citoyenneté numérique

- Établir une attitude positive envers la création de la technologie, pas seulement envers la consommation
- Donner aux enfants le goût de « regarder sous le capot » et de poser des questions à propos de la technologie qu'ils consomment

Des concepts fondamentaux de programmation

Il y a des centaines de langages de programmation, et bien qu'elles ne se ressemblent pas du tout à l'œil nu, elles sont fondamentalement les mêmes. Les concepts et les façons d'interagir avec un ordinateur restent identiques. Nous utiliserons l'outil Scratch pour enseigner ces concepts aux enfants de manière amusante et significative.

L'histoire et la culture du Canada

Pour la semaine Canada en programmation, nous avons conçu des cours qui abordent différents aspects de notre histoire. Les participants apprendront à propos d'événements historiques et d'éléments culturels et découvriront des histoires de chez nous d'une manière tout à fait novatrice. Nous avons conçu ces cours afin qu'ils complètent le programme de formation et que les enseignants puissent satisfaire aux objectifs pédagogiques traditionnels en intégrant la programmation à leurs leçons et ainsi participer à la semaine.



COMMENT ORGANISER UN COURS

Vous aurez besoin des éléments suivants:

Un enseignant

Pour superviser les apprenants et présenter les exercices de programmation.

Des mentors bénévoles

Facultatif, mais très recommandé. Vous trouverez plus d'informations à propos du recrutement et de la formation des mentors dans la section « Enseignement en collaboration ».

Apprenants

Dans les pages suivantes, vous trouverez des ressources à imprimer qui vous permettront de faire la promotion de votre cours et de bien l'organiser.

Matériel informatique

Ordinateurs • Projecteur + Écran pour le projecteur (ou toute autre façon d'afficher l'écran pour les apprenants).

Plan de cours

Vous aurez accès à des plans de cours, à des ressources, à un forum et à de multiples conseils et outils de planification!

Enseignement en collaboration

Les mentors, ces personnes remarquables, sont l'une des raisons principales pour lesquelles nos programmes sont si populaires partout au pays.

Les mentors sont des bénévoles qui contribuent à nos programmes et favorisent un climat d'apprentissage social et collaboratif. Que vous ayez des compétences techniques ou non, nous vous recommandons fortement de choisir l'enseignement en collaboration pour initier votre classe à la programmation. Cette collaboration peut se faire avec un collègue qui enseignera à vos côtés pendant votre premier cours de programmation, avec un groupe d'apprenants plus âgés qui serviront de mentors ou avec des bénévoles externes. Les mentors comptent beaucoup dans l'enseignement, car ils apportent un soutien supplémentaire tant à vous qu'aux apprenants.

RECRUTEMENT DE MENTORS

Le rôle de mentor

Quand nos formateurs en chef (enseignants) enseignent à l'avant de la classe, les mentors sont dispersés dans le local et travaillent avec des petits groupes d'apprenants. Les mentors sont de vrais magiciens. Ils travaillent en collaboration avec les apprenants, ce qui permet d'instaurer un climat d'apprentissage social et gratifiant pour tous.

Pourquoi devenir mentor

- Vous avez soit des connaissances techniques ou êtes passionné de technologie et d'apprentissage
- Vous souhaitez apprendre aux côtés des enfants et les aider avec leurs projets
- Vous êtes préparé à aider les enfants à trouver les réponses à leurs questions
- Vous savez comment favoriser un environnement d'apprentissage ouvert et flexible
- Vous restez optimiste à propos de la technologie
- Vous encouragez les enfants à être des créateurs confiants et inspirés

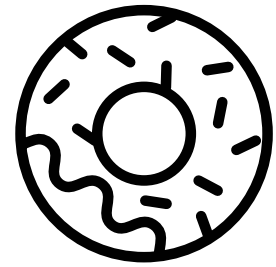
Avantages du mentorat

- L'expérience vous fera chaud au cœur!
- Vous ferez du réseautage avec d'autres mentors
- Vous apprendrez un nouveau langage ou à utiliser un nouvel outil qui ajoutera une corde à votre arc
- Vous développerez des aptitudes à diriger et à communiquer
- Vous gagnerez de l'expérience en enseignement
- Vous aiderez votre communauté

Voici nos recommandations pour trouver des mentors

- Approchez un enseignant d'expérience ou des apprenants d'un niveau plus élevé, que ce soit dans une école secondaire, un collège ou une université de votre région.
- Contactez les bénévoles actifs dans d'autres programmes et encouragez-les à consulter les ressources afin qu'ils puissent être des mentors.
- Approchez des entreprises de technologie dans votre région.
- Vous trouverez une affiche à imprimer ou à envoyer à des gens par courriel afin de recruter des bénévoles (dans la section « À imprimer »).

CRÉER UN CLIMAT FAVORABLE



Avant d'aborder le plan de cours, il est important d'instaurer un climat favorable en définissant des concepts et des valeurs clés pour orienter l'apprentissage des apprenants. Nous voulons encourager chez eux un état d'esprit axé sur la croissance, c'est à dire, une volonté d'aller au-delà de l'échec. Pour créer ce climat favorable à l'apprentissage, il est essentiel de préciser dès le début que la technologie échoue, mais que l'échec fait partie du processus d'apprentissage. Encouragez les apprenants à explorer, à essayer, à faire des erreurs et à résoudre des problèmes en collaborant. Avant tout, soyez patient. L'apprentissage de la programmation est comme celui d'une langue, car il demande beaucoup de temps, de patience et de répétition.

Voici quelques enjeux courants dans nos programmes que nous souhaitons aborder:

Problèmes avec la technologie

Si un ordinateur ne donne pas les résultats escomptés et qu'un apprenant ressent de la frustration, dites-lui qu'il s'agit d'une situation fréquente. Essayez de régler le problème ensemble. Demandez-lui ce qu'il ferait si ce problème survenait chez lui. Fermerait-il et relancerait-il le programme? Redémarrerait-il l'ordinateur? Si le problème n'est toujours pas réglé, cherchez la solution sur Google! Il est important de montrer à l'apprenant comment trouver des solutions lui-même pour régler des problèmes dans le futur.

Résolution de problèmes par les apprenants

Encouragez les apprenants à s'entraider avant de venir vous voir. Les autres apprenants peuvent souvent régler un grand nombre des problèmes reliés à la technologie. C'est l'occasion parfaite pour développer leur capacité à montrer la voie aux autres. Vous pouvez également établir une liste de choses à vérifier pour résoudre des problèmes avec les apprenants. Finalement, que fait-on quand rien ne fonctionne? On cherche sur Google! La débrouillardise et la capacité à faire des recherches sur Internet sont des compétences extrêmement importantes que nous vous encourageons à développer.

Apprendre en posant des questions

Nous avons créé de nombreux exercices pour initier les apprenants aux concepts de programmation de base. Toutefois, nous vous rappelons qu'ils ne sont que des points de départ. Il est essentiel de permettre aux apprenants de poser des questions et de les laisser suivre leurs idées pour apprendre.

QUESTION DE DIVERSITÉ



Il est indéniable que l'industrie de la technologie a un problème de diversité. Nous avons donc conçu nos plans de cours en mettant l'accent sur celle-ci. Pour renverser la vapeur dans l'industrie, il est important d'encourager l'hétérogénéité des groupes d'apprenants.

Conseils pour rendre le cours intéressant pour tous:

Associez la programmation à des projets de création signifiants pour les élèves.

Choisissez des exercices de programmation reliés aux passions de vos apprenants ou modifiez-les en fonction de leurs préférences. Aiment-ils la musique? Les arts? Les animaux? La philanthropie?

Faites des liens entre la programmation et la technologie et des carrières intéressantes et créatives

Les emplois en technologie ne sont pas tous les mêmes : il y en a des milliers de différents. Les concepteurs travaillent sur des jeux, des films, des appareils médicaux et plus encore. En leur faisant découvrir ces différents emplois de création, vous contribuez à approfondir leurs connaissances à propos de l'industrie.

Favorisez un climat d'apprentissage social et collaboratif

Encouragez le travail en équipe, le mentorat par les pairs et mettez en valeur les réussites et le travail des apprenants. Organisez des activités pour briser la glace et des jeux pour créer des liens entre les apprenants.

Favorisez la participation de mentors de tous les horizons

Vous pouvez recruter des bénévoles ou inviter des femmes travaillant dans l'industrie à venir parler aux apprenants ou à servir de mentores (des mentors de sexe féminin).

Insistez sur le pourquoi, pas seulement sur le comment

Rappelez-vous les raisons pour lesquelles la programmation est importante : ce n'est pas que du code. Les jeunes filles ont besoin de savoir que leur travail a des répercussions et peut changer le monde.

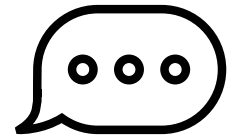
Soyez conscient des préjugés inconscients

Il est essentiel de rester vigilant à propos des préjugés. Inconsciemment, nous orientons les garçons vers des « choses de garçon » et les filles, vers des « choses de fille ». Il y a des tendances subtiles dans la société qui ont une influence sur les apprenants, par exemple, penser que « les filles ne sont pas bonnes en maths ».

Soyez conscient de la présence du syndrome de l'imposteur

Le syndrome de l'imposteur est très présent chez les femmes et est souvent associé aux apprenants les plus performants. Le syndrome de l'imposteur est le sentiment de ne pas être intelligent, de ne pas avoir du succès ou de ne pas pouvoir apprendre quelque chose. Par conséquent, l'apprenant croit qu'il est un « imposteur ». Il faut redoubler d'efforts pour renforcer les compétences des filles, particulièrement dans les disciplines des mathématiques, des sciences et des technologies.

CONCEPTS DE PROGRAMMATION CLÉS



Voici quelques concepts de programmation clés pour vous permettre de comprendre Scratch et la programmation davantage. Vous remarquerez que plusieurs de ces concepts existent dans d'autres domaines et la vie quotidienne. Ils ne sont donc pas aussi étranges ou effrayants que vous le croyez!

Variable

enregistre une information p. ex. : le score d'un jeu qui augmente d'un point par but

Tableau

permet d'enregistrer plus d'une information

Fonction

un type de procédure ou de routine qui exécute une opération particulière. Il y a souvent des fonctions « enchaînées » composées de fonctions qui existent déjà, comme le bloc « sauter ».

Séquence

déterminer une série d'étapes pour une tâche. Les ordinateurs et Scratch lisent et exécutent des commandes dans un ordre précis, de haut en bas.

Événements

une action qui entraîne une autre action p. ex. : le bloc « quand _ est cliqué »

Instructions conditionnelles

prendre des décisions selon des conditions p. ex. : si une condition est satisfaite, une action est accomplie, sinon, rien ne se passe ou une autre action est accomplie

Boucles

exécuter la même séquence de nombreuses fois p. ex. blocs « répéter » et « répéter indéfiniment »

Logique booléenne

les opérateurs « et » (and) « or » (ou) et « pas » (not) sont des exemples de logique booléenne. Ce sont des valeurs qui peuvent être vraies ou fausses.

Parallélisme

faire en sorte que des actions se déroulent en même temps

Opérateurs

expressions mathématiques et logiques p. ex. : bloc X+X

Débogage

trouver des problèmes dans du code et les résoudre

Remixer

prendre un projet ou une idée déjà créée et la changer ou y ajouter des idées

Modularisation

explorer les liens entre les parties et le tout

Syntaxe

la grammaire et l'orthographe d'un langage de programmation. La structure en blocs élimine le besoin de suivre la syntaxe du langage.

Algorithme

un ensemble d'opérations par étape à suivre pour résoudre un problème

États

un état en programmation est la même chose qu'un état dans un autre contexte. p. ex. : la télé est allumée ou éteinte. Les variables ont des états, mais les valeurs n'en ont pas. Par exemple, 42 est 42, et on ne peut rien y changer.



POUR BIEN COMMENCER AVEC SCRATCH



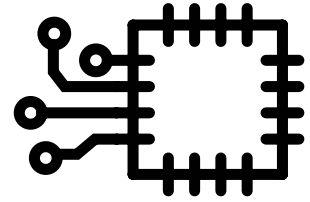
Nos cours de programmation sont conçus pour apprendre le langage de programmation graphique de l'outil Scratch, mais il y en a de nombreux autres, y compris Pencil Code, Touch Develop et Blockly, qui permettent aux utilisateurs de créer des programmes. Ce que ces langages de programmation graphiques ont en commun est la capacité de déplacer des blocs de directives pour créer des programmes. L'outil Scratch élimine le besoin d'écrire du code ou de s'occuper des points complexes de la syntaxe comme les deux-points, les virgules et les parenthèses, qui sont une distraction à l'apprentissage des concepts clés pour les débutants. C'est la raison pour laquelle Scratch est un excellent outil pour les débutants et les jeunes qui apprennent la programmation.

Nous avons décidé d'utiliser Scratch, car nous avons eu beaucoup de succès avec l'outil dans

le cadre de nos programmes. De plus, il y a une très grande communauté d'enseignants qui l'utilisent partout dans le monde. Nous l'aimons aussi pour sa facilité d'utilisation, que ce soit dans un navigateur avec la version Web ou hors ligne sur l'ordinateur avec la version téléchargée. C'est un énorme avantage de pouvoir l'utiliser sans Internet!

Au bout du compte, Scratch est conçu pour l'apprentissage et l'enseignement de la programmation. C'est un programme collaboratif fantastique pour dessiner, jouer de la musique et créer des jeux. C'est aussi l'endroit parfait pour expérimenter, en mettant à l'essai des fonctions mathématiques, des notions de géométrie, des graphiques, des pages Web, des simulations et des algorithmes.

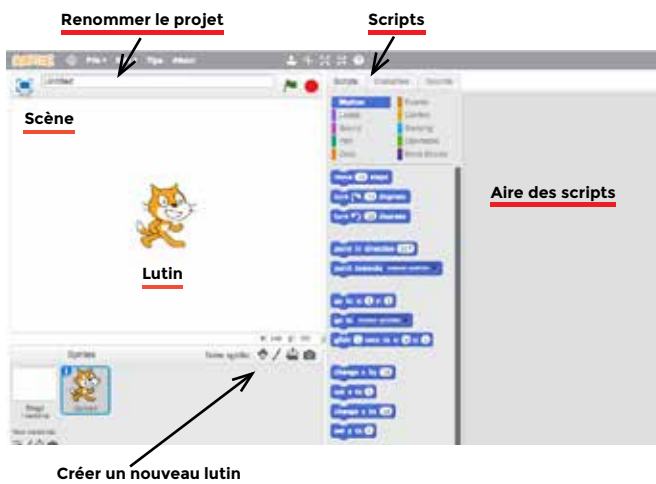
INTRODUCTION À SCRATCH



La leçon

Dans le cadre de l'exercice, nous vous ferons découvrir l'outil Scratch. Nous vous montrerons comment créer un nouveau projet, et quelles sont les fonctions principales.

1. Visitez le scratch.mit.edu.
2. Sur la page d'accueil, cliquez sur « Rejoindre Scratch » (si vous avez déjà un compte, cliquez sur « Se connecter ». Naviguez jusqu'en bas de la page et choisissez la langue de votre choix dans le menu déroulant.
3. Quand vous êtes connecté, cliquez sur « Créer » pour créer un nouveau projet.
4. C'est le temps d'explorer! Faites glisser des blocs (scripts) dans l'aire des scripts et regardez ce qui se passe.



5. N'oubliez pas d'enregistrer votre travail. Vous trouverez aussi des conseils pour utiliser Scratch dans la section « Conseils ».

L'objectifs d'apprentissage

- Familiarisez-vous avec la manière d'accéder à Scratch et apprenez comment créer un nouveau projet et utiliser l'éditeur

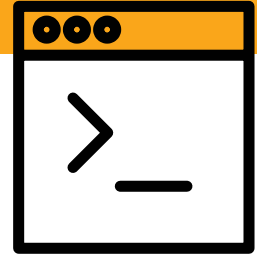
À préparer

- Les comptes Scratch en ligne nécessitent une adresse courriel. Planifiez si les apprenants créeront leur propre compte ou si vous allez créer un compte à utiliser par toute la classe.
- Pensez à imprimer ou à publier les identifiants. Consultez la section « À imprimer » pour voir un gabarit.

Prolongements et modifications

- Pensez à faire ce processus en grand groupe en suivant les étapes.
- Si les apprenants comprennent le tout rapidement, donnez-leur 10 minutes pour explorer Scratch et faire bouger le chat Scratch.
- Encouragez les apprenants à travailler en équipe, à s'entraider et à partager leurs découvertes.

NOTIONS DE BASE DE L'ÉDITEUR SCRATCH



Éléments clés de l'éditeur Scratch:

Lutins

Tous les objets dans Scratch sont des « lutins ». Vous pouvez ajouter des lutins en choisissant des éléments dans la bibliothèque, en dessinant vos propres lutins, en téléversant une image ou en prenant une photo avec votre webcam.

Scène

l'endroit où vous pouvez voir un aperçu du résultat de votre code. Vous pouvez ajouter des arrière-plans à votre scène. Vous pouvez « Démarrer » ou « Arrêter » l'aperçu en tout temps en cliquant sur le drapeau vert ou le panneau d'arrêt rouge en haut de la scène.

Scripts

Les scripts sont des commandes. C'est là que tout commence. Les apprenants utiliseront des scripts pour indiquer à leur lutin les actions à faire et comment les faire. Par exemple, ils peuvent faire avancer leur lutin et même lui faire dire « Bonjour ». Les scripts sont organisés en une séquence logique pour « programmer » les actions du lutin. Nous aborderons ces éléments en détail au fil de votre progression.

Aire des scripts

Toute la programmation se fait ici. Vous pouvez déplacer des scripts dans cette aire. Pour effacer les scripts, déplacez-les à l'extérieur de l'aire.

Scène

Lutins

Scripts

Aire des scripts

Conseil!
Vous pouvez double-cliquer un bloc à tout moment pour exécuter le codet

CONCEPTS DE BASE SUITE



Il est essentiel que les apprenants remarquent le « chapeau » (c'est à dire, la surface ronde) en haut du bloc. Les autres blocs peuvent être connectés des deux côtés, mais ce bloc-ci doit absolument être placé en haut. Ce type de bloc permet à un ensemble d'actions d'être faites au début d'un événement en particulier.



Les blocs sont les outils de la programmation dans Scratch. Vous pouvez les déplacer de leur boîte à outils vers l'aire de scripts pour interagir avec eux et les emboîter. Pour changer la valeur à l'intérieur de certains blocs, cliquez le numéro et tapez-en un nouveau. Les blocs s'emboîtent seulement quand le lien entre eux est logique ou pragmatique.



Ce bloc semble inutile quand il est utilisé seul, mais d'autres blocs peuvent être insérés à l'intérieur. La petite flèche en bas du bloc donne un indice à propos de sa fonction. En effet, quand les blocs contenus à l'intérieur ont tous été exécutés l'un après l'autre et que nous en sommes à la fin du bloc, le programme retourne au début et recommence à exécuter les blocs.



Mettre des blocs dans d'autres blocs. Il est possible de mettre des blocs dans des blocs, qui sont eux aussi dans des blocs. Par exemple, vous pouvez placer des blocs « Opérateurs » à l'intérieur de blocs « Mouvement » pour créer des commandes avancées, comme pour faire marcher un lutin en lui faisant faire un nombre de pas sélectionné au hasard chaque fois.



Les blocs avec « Si » nous permettent de prendre des décisions. Plus les apprenants feront des exercices de programmation complexes et créeront des projets avancés, plus les blocs de conditions comme si <>alors gagneront en importance. Ce bloc dit au lutin de faire une action si une certaine condition est satisfaite. Si elle ne l'est pas, une autre action est enclenchée.

Remarque : Vous trouverez plus de détails à propos des blocs de script et des exemples de leur utilisation dans la section « Conseils » et la sous-section « Blocs » de l'éditeur Scratch.

DÉBOGAGE DANS SCRATCH

Le débogage de code est une partie importante du processus d'apprentissage de la programmation, tant pour les apprenants que pour les enseignants. Voici quelques problèmes fréquents que nous rencontrons dans Scratch et qui vous aideront à commencer.

Comment effacer des blocs?

Il suffit de déplacer des blocs à l'extérieur de l'aire des scripts. Si vous voulez effacer un lutin ou un arrière-plan, vous n'avez qu'à faire un clic droit sur l'objet que vous souhaitez effacer.

J'ai perdu tout mon code!

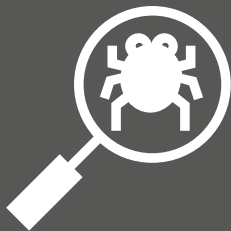
Les algorithmes sont un ensemble de directives indiquant à l'ordinateur quoi faire. Les algorithmes sont partout dans notre vie : un plan de cours est lui-même un algorithme pour une leçon, et une recette est un algorithme pour préparer notre plat préféré. En écrivant des directives par étape en français, avec des phrases, nous faisons ce que l'on appelle du « pseudo-code ».

Mon code ne fonctionne pas.

Le code est complexe et les problèmes le sont tout autant. Un élément essentiel de la pensée informatique et de la programmation est la division d'un problème en petites parties faciles à gérer, comme quand nous divisons un résumé de livre en sections différentes.

Mon lutin est parti à l'extérieur de l'écran et je ne le trouve pas. Aidez-moi!

Après avoir divisé les problèmes en petites parties, l'abstraction nous permet de déterminer ce qui est important et ce qui ne l'est pas. Grâce à elle, nous pouvons gérer la complexité, comme dans un problème de maths en décidant quelles informations sont nécessaires à la résolution d'une équation.



La meilleure façon de prévenir les erreurs est d'organiser son code proprement et de le mettre à l'essai souvent. En trouvant des problèmes tôt dans un projet, vous économisez du temps et prévenez des tracas plus tard!

GESTION DES IMPRÉVUS

Avouons-le : quand la technologie nous fait défaut, que nous perdons l'accès à Internet, que les ordinateurs plantent et que nous perdons du travail, nous sentons que nous avons échoué. Nous n'aimons pas ça en tant qu'enseignants, et honnêtement, les apprenants n'aiment pas ça non plus. Heureusement, il y a des façons de se préparer et de réduire les chances que tout cela arrive!

Planifiez pour prévenir les pépins

Plus vous planifiez en cas de pépin technique, plus vous serez confiant quand le cours commencera.

- Pouvez-vous préparer les ordinateurs avant la leçon?
- Écrivez les identifiants et les mots de passe sur le tableau. Vous trouverez un gabarit pour ce faire dans la section « À imprimer ». Utilisez un identifiant universel plutôt que des identifiants pour chaque élève.
- Chargez les ordinateurs portables avant la leçon.
- Décidez si les apprenants peuvent travailler en équipe.

Apprendre en posant des questions et déboguer

Nous avons développé de nombreux exercices de programmation pour initier les apprenants aux concepts de programmation de base. Toutefois, nous vous rappelons qu'ils ne sont que des points de départ. Il est essentiel de permettre aux apprenants de poser des questions et de les laisser suivre leurs idées pour apprendre.

Apprenez les bases de la résolution des problèmes informatiques. Il y a quelques problèmes qui peuvent occuper la majorité du temps mort en classe. Voici les deux plus importants : Si les ordinateurs ne démarrent pas, vérifiez s'ils sont bien branchés. Si ce n'est pas le problème, redémarrez l'ordinateur.

Cherchez le problème sur Google. Google est votre ami! Si quelque chose ne va pas avec l'accès Wi-Fi, le clavier ou l'outil Scratch lui-même, tapez votre problème dans Google et regardez si la solution ne s'y trouve pas. La plupart du temps, vous la trouverez.

Demandez aux apprenants. Ils sont très bons en technologie et peuvent souvent s'entraider pour résoudre leurs problèmes. Un pépin technique est une excellente activité de résolution de problème en grand groupe!

Activités sur papier

Vous n'avez pas besoin d'un ordinateur pour enseigner la programmation! Il y a de nombreuses façons d'enseigner les concepts de programmation de base, comme en faisant l'exercice *Ruby Robot*.

PROCHAINES ÉTAPES



Félicitations, vous avez réussi votre premier cours de programmation! Prenez une minute pour réfléchir à propos de votre expérience et déterminez en quoi celle-ci pourrait influencer votre manière d'enseigner la programmation à vos apprenants.

- Quels sont les problèmes que vous avez rencontrés? Comment avez-vous résolu ces problèmes?
- Quels sont les obstacles que vos apprenants rencontreront, à votre avis?
- Comment pouvez-vous vous préparer pour que les apprenants réussissent ces exercices de base?
- Quelles questions pourriez-vous poser aux apprenants pour faire des liens entre leurs apprentissages et la vraie vie?
- Quelles ressources sont à votre disposition si un apprenant est « coincé » par un obstacle?

Évaluation

Dans le cadre de nos programmes, nous nous éloignons de l'évaluation pour mettre l'accent sur la créativité, la création et la démarche. Malgré cela, nous vous suggérons quelques manières d'évaluer un apprenant pendant son apprentissage.

Concepts de programmation

- L'apprenant a-t-il démontré une compréhension des concepts de programmation en utilisant des blocs appropriés dans sa solution?
- Le projet ou le code finalisé de l'apprenant est-il organisé, logique et débogué?

Démarche

- L'apprenant a-t-il planifié son travail?
- L'apprenant a-t-il bien utilisé son temps?
- L'apprenant a-t-il terminé les exercices ou fait des exercices supplémentaires?

Matière et contenu

- L'apprenant a-t-il fait des liens entre des matières et son projet?

AUTRES RESSOURCES

Éditeur Scratch hors ligne

scratch.mit.edu/scratch2download

ScratchEd

scratched.gse.harvard.edu

ScratchJr

scratchjr.org

Déboguer Scratch

wiki.scratch.mit.edu/wiki/Debugging_Scripts

Scratch Wiki

wiki.scratch.mit.edu

Carte Scratch

scratch.mit.edu/help/cards

FAQ pour Scratch

scratch.mit.edu/info/faq

Site Web de Teachers Learning Code

teacherslearningcode.com

Cours « CS50 Intro to Computational Thinking + Scratch » de l'Université Harvard

Vidéo 1: <http://bit.ly/CS50part1>

Vidéo 2: <http://bit.ly/CS50part2>



À IMPRIMER

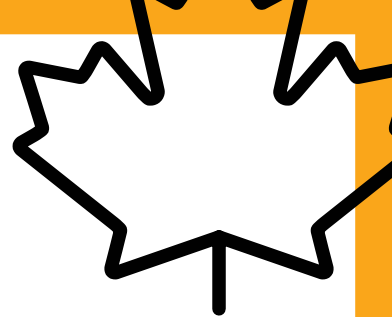
Dans les pages suivantes, vous trouverez des ressources à imprimer qui vous permettront de faire la promotion de votre cours et de bien l'organiser. N'hésitez pas à imprimer toutes les ressources, voire le guide lui-même pour vous aider!

Ressources à imprimer:

1. Affiche promotionnelle pour les apprenants
2. Feuille d'identifiants Scratch à remplir
3. Leçon "Do the Robot" hors ligne
4. Certificat de réussite

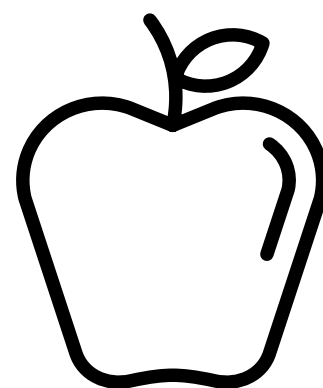
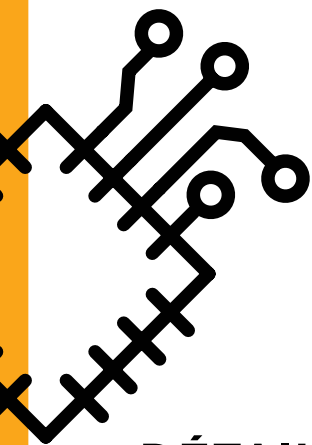


✖ teachers learning code



Comment enseigner la programmation en classe

Enseigner la programmation
pour les débutants



DÉTAILS SUR L'ATELIER

DATE

HEURE

EMPLACEMENT

INSCRIPTION



Un programme éducatif de

CANADA EN PROGRAMMATION



ÇA NE FONCTIONNE PAS?

INSPIREZ et *EXPIREZ* profondément

Votre projet ne bouge plus?

Cliquez sur l'icône « rafraîchir » en haut de votre navigateur 

Ça ne fonctionne toujours pas?

Demandez de l'aide à votre voisin ou au mentor de votre table

Cliquez sur « Conseils » dans le menu du haut, puis cliquez sur l'onglet « Comment faire » ou « Blocs »

Vous ne comprenez pas?

Levez la main et attendez qu'un enseignant ou mentor vienne vous aider

DÉMARRAGE

1



Double-cliquez sur l'icône Google Chrome pour ouvrir le navigateur

2

Comment y accéder:

3

Suivez les étapes :

Créez un NOUVEAU nom d'utilisateur et un mot de passe (cliquez sur « Suivant »)

Sélectionnez votre mois et votre année de naissance, votre sexe et votre pays, et entrez votre adresse courriel (cliquez sur « Suivant »)

Cliquez sur « OK, allons-y », puis sur « Créer » dans le menu du haut

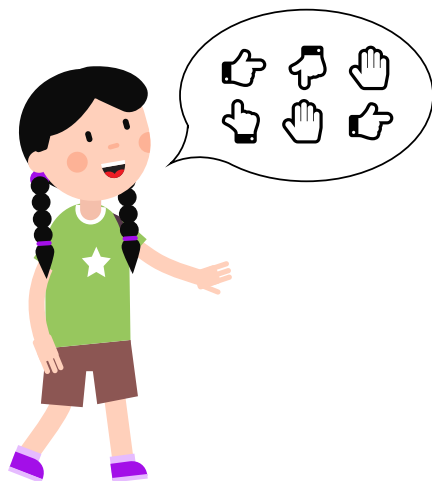
SUR LA
BONNE
VOIE



LEÇON HORS LIGNE « ROBOT »

La leçon

1. Un exercice de base pour inciter les élèves à penser à des directives et à des séquences simples, ou, en des termes de programmation, à la création d'algorithmes.
2. Faites des groupes de deux et demandez à un apprenant de jouer le rôle du « programmeur ». L'autre apprenant jouera celui du « robot ».
3. Assignez à chaque groupe une activité, par exemple, attacher des lacets ou ouvrir une porte.
4. Demandez au « programmeur » d'indiquer à son partenaire (le « robot ») les étapes à suivre pour compléter l'activité seulement avec des mots!
5. Faites de nouveaux groupes de deux.
6. Après l'activité, profitez de l'occasion pour parler de l'importance de créer des directives et des séquences claires et simples.
7. Félicitez les apprenants pour la création de leurs premiers algorithmes et d'un pseudo code!



Objectifs d'apprentissage

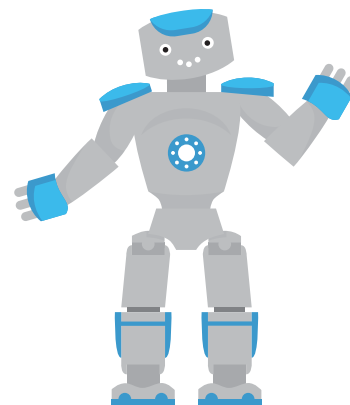
- Créer des commandes de base avec la programmation
- Algorithmes

À préparer

- Exemples d'activités à compléter à deux, par exemple : attacher des lacets, ouvrir une porte, faire la macarena

Prolongements et modifications

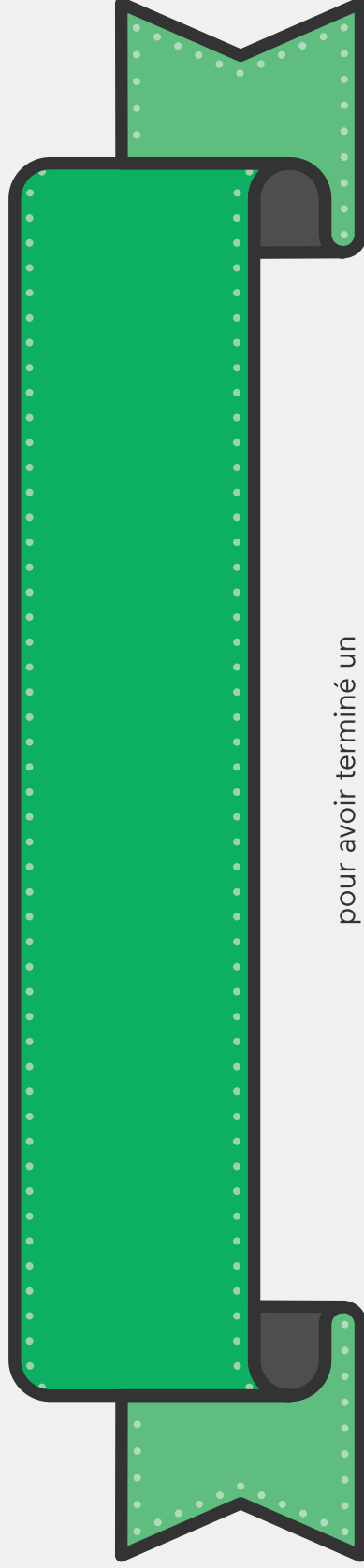
- Demandez aux élèves d'écrire leur code et de le partager avec d'autres groupes afin que ceux-ci puissent l'exécuter
- Choisissez un extrait de code à regarder en grand groupe. Les apprenants remarquent-ils des répétitions? Peuvent-ils simplifier les algorithmes?





FÉLICITATIONS

Ce certificat est décerné à



pour avoir terminé un

atelier Kids Learning Code

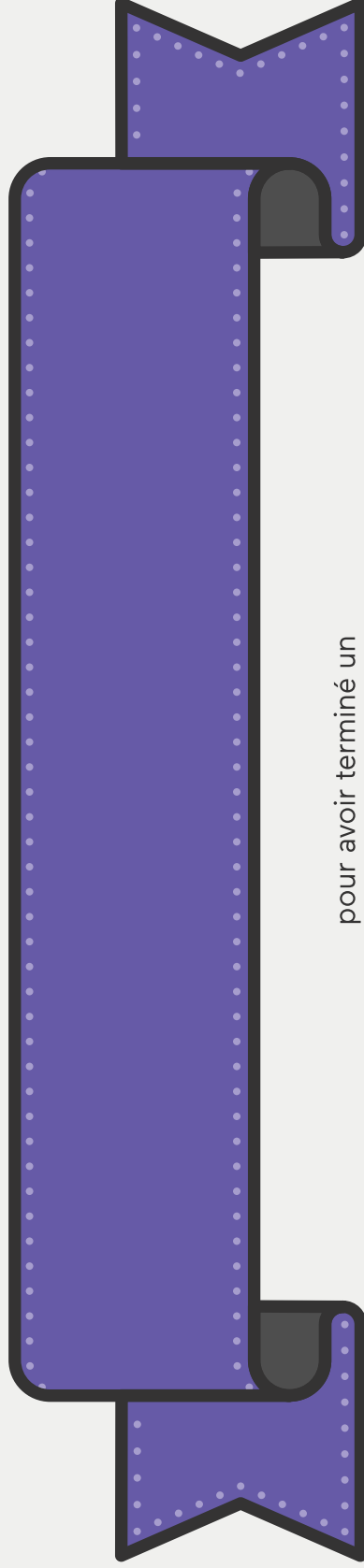
et démontré une compréhension
des concepts de programmation de base.

✘ kids
learning
code



FÉLICITATIONS

Ce certificat est décerné à



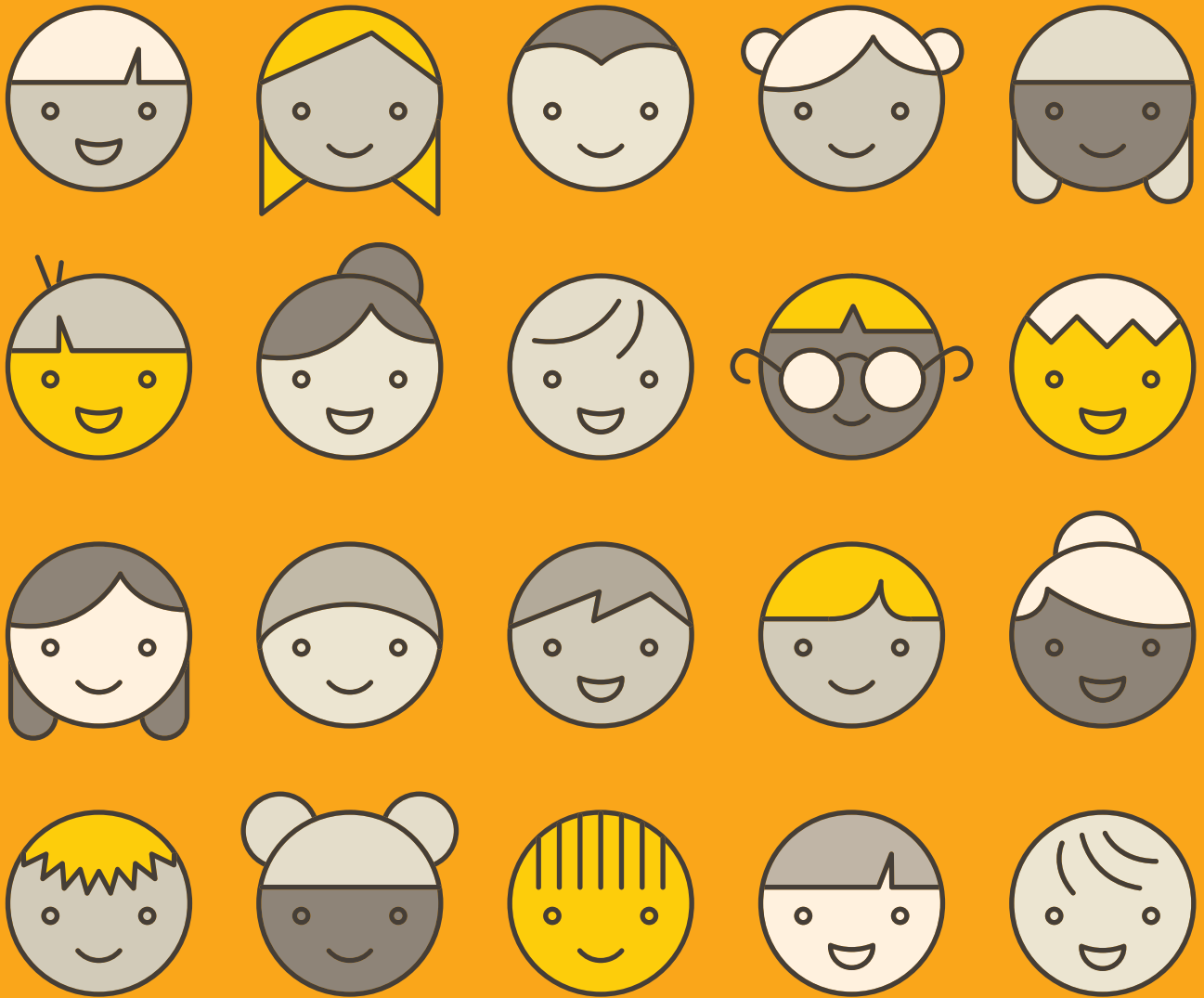
pour avoir terminé un

atelier Girls Learning Code

et démontré une compréhension
des concepts de programmation de base.

✖ girls
learning
code





MERCI!

Communiquez avec nous:

501, 129 Spadina Avenue
Toronto, ON M5V 2L3

info@canadalearningcode.ca
canadalearningcode.ca

✳️ teachers learning code