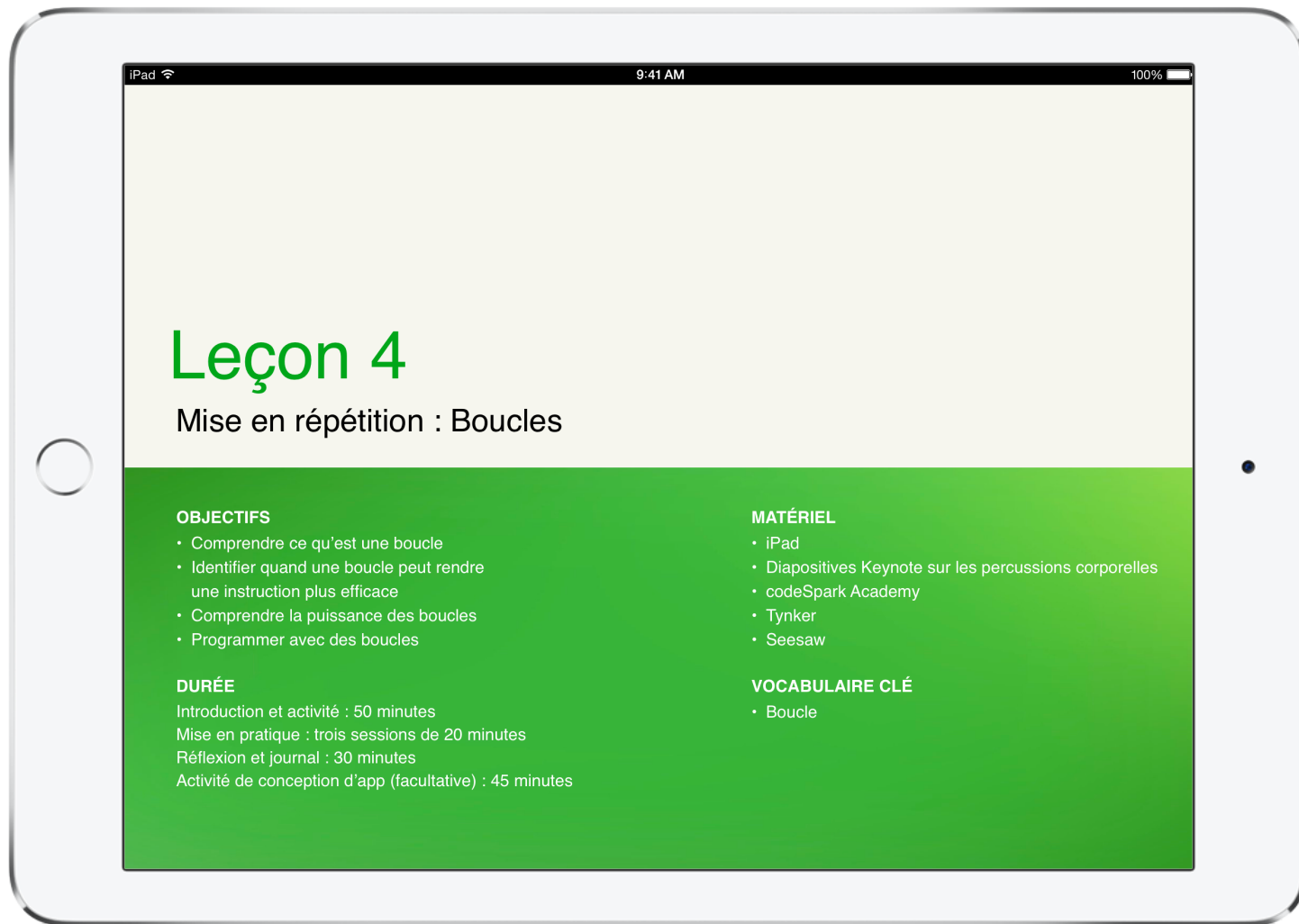




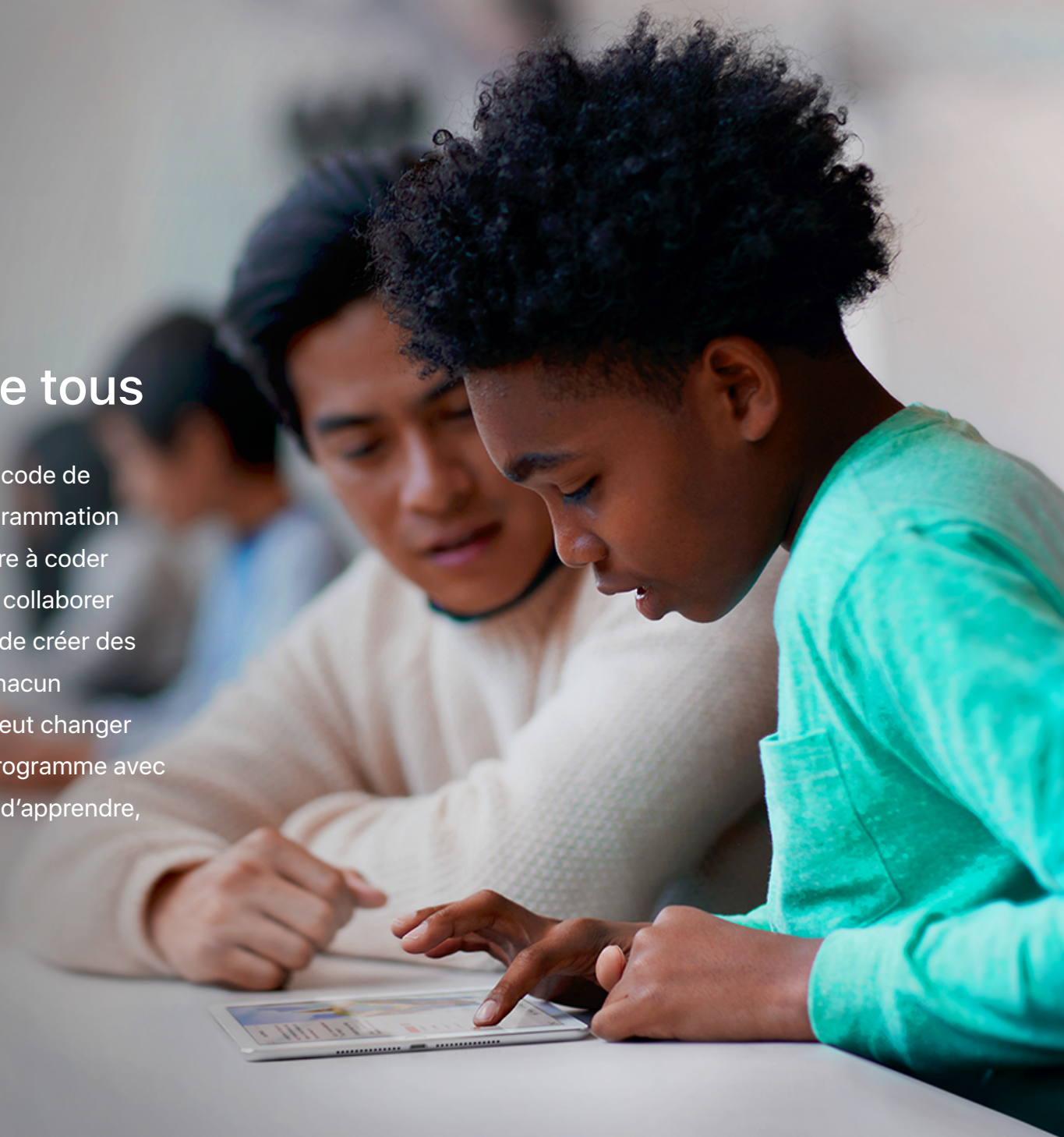
# Guide du programme Débuter en programmation

Septembre 2017



# Le code à la portée de tous

Les technologies ont leur propre langage : le code de programmation. Et nous pensons que la programmation est une compétence fondamentale. Apprendre à coder vous apprend à résoudre des problèmes et à collaborer de manière créative. Cela vous permet aussi de créer des apps qui donnent vie à vos idées. D'offrir à chacun l'opportunité d'imaginer quelque chose qui peut changer le monde. Nous avons élaboré un nouveau programme avec les outils et ressources permettant à chacun d'apprendre, d'écrire et d'enseigner la programmation.



# Programme Le code à la portée de tous

Le programme Le code à la portée de tous inclut toute une gamme de ressources qui guident les élèves, de leur initiation à la programmation à la conception de leurs premières apps. Le tableau ci-dessous offre un aperçu de toutes les ressources pédagogiques disponibles gratuitement.

Programme	Appareil	Public	App	Prérequis	Présentation	Supports pédagogiques	Ressources d'assistance	Nombre d'heures de cours incluses
		De la maternelle au CE1		Aucun	Commencer à penser comme des programmeurs, avec découverte pratique des concepts de programmation à l'aide d'apps visuelles.	<ul style="list-style-type: none"> <li>Leçons de l'app codeSpark Academy</li> <li>Cours Cadet de l'espace sur Tynker</li> </ul>	<ul style="list-style-type: none"> <li>Débuter en programmation 1 : Guide d'enseignement</li> </ul>	30 heures, en comptant le Guide d'enseignement et les leçons de l'app
		Du CE2 au CM2		Aucun	Explorer les concepts de base de la programmation et s'entraîner à penser comme des programmeurs à l'aide d'apps visuelles.	<ul style="list-style-type: none"> <li>Cours Sorts Dragon sur Tynker</li> </ul>	<ul style="list-style-type: none"> <li>Débuter en programmation 2 : Guide d'enseignement</li> </ul>	36 heures, en comptant le Guide d'enseignement et les leçons de l'app
		À partir du collège		Aucun	Apprendre les bases de la programmation à l'aide du code Swift.	<ul style="list-style-type: none"> <li>App Swift Playgrounds</li> <li>Leçons Apprendre à coder 1 et 2</li> <li>Cours iTunes U</li> </ul>	<ul style="list-style-type: none"> <li>Apprendre à coder 1 et 2 : Guide d'enseignement</li> <li>Badges Swift Playgrounds Apple Teacher Learning Center</li> </ul>	Jusqu'à 85 heures, en comptant le Guide d'enseignement et les leçons Apprendre à coder 1 et 2
		À partir du collège		Apprendre à coder 1 et 2	Approfondir ses compétences en programmation et apprendre à penser comme un développeur d'apps.	<ul style="list-style-type: none"> <li>App Swift Playgrounds</li> <li>Leçons Learn to Code 3</li> </ul>	<ul style="list-style-type: none"> <li>Learn to Code 3: Teacher Guide</li> </ul>	Jusqu'à 45 heures, en comptant le Guide d'enseignement et les leçons Learn to Code 3
		Lycée et université		Aucun	Acquérir une expérience pratique grâce aux outils, techniques et concepts nécessaires pour concevoir une app iOS simple de A à Z.	Manuel Introduction au développement d'apps avec Swift et fichiers de projet	<ul style="list-style-type: none"> <li>Introduction au développement d'apps avec Swift : Guide d'enseignement</li> </ul>	90 heures
		Lycée et université		Aucun	Acquérir des connaissances solides sur Swift, UIKit et les technologies réseau grâce à des travaux pratiques et des projets guidés. À l'issue du cours, les élèves sont capables créer une app issue de leur imagination.	Livre App Development with Swift et fichiers de projet	<ul style="list-style-type: none"> <li>App Development with Swift: Teacher Guide</li> </ul>	180 heures

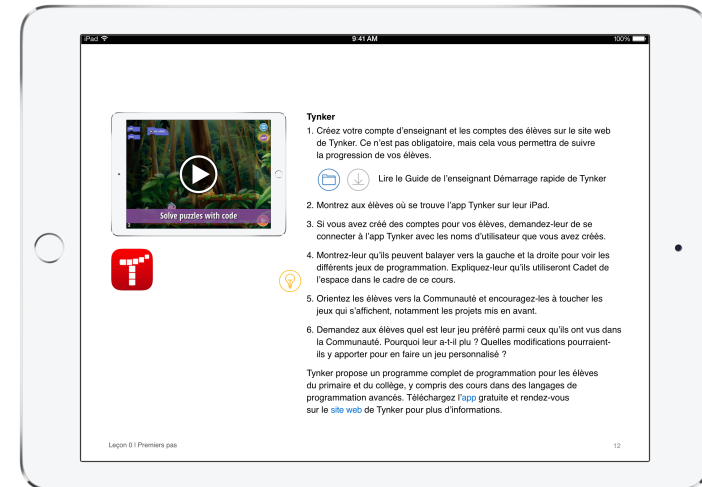
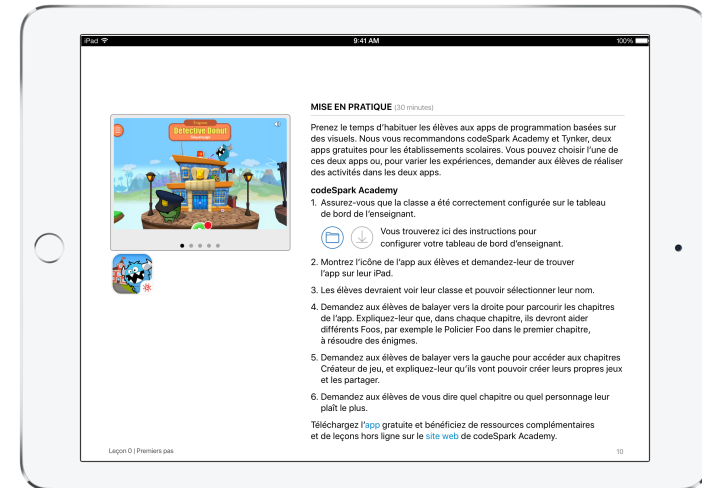
# Présentation

Les premières années d'école sont idéales pour présenter les concepts de programmation comme une manière de penser le monde numérique et le quotidien, ainsi que pour développer les compétences de base en réflexion informatique. Les apps spécialement conçues pour les plus jeunes apprenants, comme codeSpark Academy et Tynker, font appel à des puzzles de programmation visuels pour développer les compétences de résolution de problèmes, inciter à la persévérance et promouvoir la créativité. codeSpark Academy s'adresse aux élèves âgés de cinq à sept ans. Son interface dépourvue de texte permet à tous les élèves de jouer, y compris ceux n'ayant pas encore appris à lire, en cours d'apprentissage du français ou rencontrant des difficultés en lecture. Dans Tynker, les élèves âgés de cinq à onze ans commencent par travailler avec des blocs visuels, puis passent à la programmation écrite pour résoudre des énigmes et concevoir des projets.

## En classe

Tynker, codeSpark Academy et les leçons des Guides d'enseignement Débuter en programmation sont conçues pour vous aider à introduire la programmation dans les classes de début de primaire. Les leçons mettent en avant les concepts clés de la programmation et montrent que le code correspond à une manière de penser pouvant être appliquée à d'autres secteurs d'apprentissage et à la vie quotidienne.

Les Guides d'enseignement vous permettent d'aider vos élèves à résoudre les puzzles de programmation, quel que soit votre propre niveau de compétence. Les activités connexes, les activités de conception d'apps, les questions de réflexion, les invitations à compléter le journal, la grille de notation et les autres rubriques ont pour but de vous aider à améliorer la compréhension des élèves vis-à-vis des thèmes abordés. Vous pouvez dispenser les cours en une fois, ou les diviser en plusieurs parties. Les tableaux des correspondances figurant dans les annexes fournissent des informations de conformité préliminaire avec les normes informatiques de l'enseignement primaire et secondaire de la CSTA (Computer Science Teachers Association) pour le niveau 1.





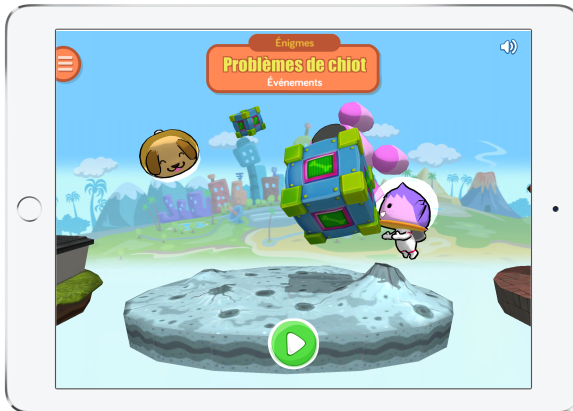
# Principales caractéristiques

## codeSpark Academy

**Formation.** Dans ce jeu uniquement basé sur des visuels, les élèves doivent résoudre des puzzles pour apprendre des concepts informatiques essentiels, comme le séquençement, les boucles, les instructions conditionnelles, etc.

**Développement.** Les élèves mettent ensuite leurs connaissances en application en programmant leurs propres projets dans le Créateur de jeu.

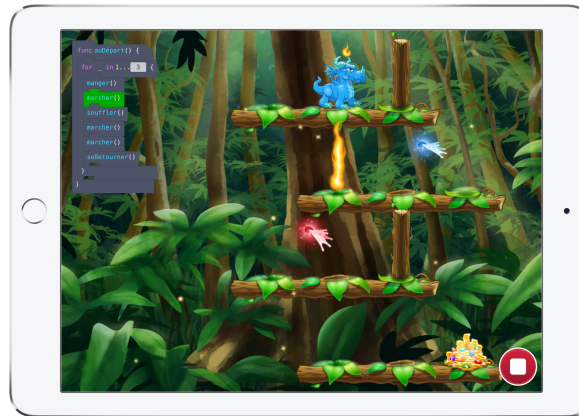
**Autonomie.** Il n'est pas nécessaire d'avoir une expérience préalable en programmation pour enseigner, apprendre ou jouer. Le programme et le tableau de bord de l'enseignant pour suivre la progression des élèves sont disponibles gratuitement et en 10 langues.



## Tynker

### Environnement de programmation.

Les élèves avancent à leur propre rythme dans des puzzles de programmation structurés, qui leur font découvrir les concepts pour ensuite les appliquer de manière créative.



**Évaluation automatique.** Un tableau de bord de l'enseignant évalue les compétences des élèves dans les puzzles, les questionnaires et les analyses de code.

**Fonctionnalité Swift.** Lorsqu'ils travaillent sur les puzzles, les élèves peuvent passer de l'affichage de blocs visuels à l'affichage de blocs de code Swift, pour se familiariser avec le langage et se préparer à pratiquer la programmation.

## Guides d'enseignement Débuter en programmation

**Fichiers téléchargeables.** Les modèles pour les activités des élèves et les présentations Keynote accompagnent l'apprentissage des élèves en classe.

**Solutions.** Les solutions aux puzzles de Tynker et codeSpark Academy vous permettent d'aider les élèves bloqués.

**Exemples de travaux d'élèves.** Regardez à quoi ressemblent les activités.

**Réflexion.** Ces questions et ces invitations à la discussion en classe sont conçues pour vous aider à réviser, et à renforcer la connexion entre l'application de la notion à l'intérieur et à l'extérieur de l'environnement de programmation.

**Astuces et exemples.** Des idées pour compléter ou simplifier les leçons sont intégrées tout au long des guides.

**Activités de conception d'apps.** Ces leçons guident les élèves tout au long d'un processus de conception en vue de conceptualiser et de prototyper une idée d'app permettant de résoudre un problème affectant la classe ou l'école.

# Plan du cours

## Débuter en programmation 1

Les élèves découvrent les concepts de programmation à travers des situations concrètes et des activités pratiques et interactives, ce qui leur apprend à penser comme des programmeurs. Ils découvrent ce que sont les commandes, les séquences, les boucles, les événements et les algorithmes. Avec la collaboration, les élèves s'entraînent à anticiper le résultat de leur code ainsi qu'à corriger leur code et celui des autres. Ils vont également s'entraîner à exploiter leurs compétences dans des apps visuelles, à résoudre des puzzles et à concevoir des créations de leur propre chef. Des activités de conception facultatives guident les élèves tout au long d'un processus de conception en vue de conceptualiser et de prototyper une idée d'app permettant de résoudre un problème affectant la classe ou l'école.

**Leçon 0 — Premiers pas.** Découvrez ce que vos élèves savent des apps et de la programmation, installez le mur de travail de la classe et présentez-leur les apps qu'ils utiliseront le plus au cours des leçons. Les élèves découvrent les différents rôles répartis au sein d'une équipe de conception d'apps.

**Leçon 1 — Mise en ordre : Introduction au séquençement.** Les élèves découvrent des séquences dans la vie quotidienne, construisent une séquence d'après une histoire familière, et utilisent des séquences simples pour résoudre des puzzles dans des apps de programmation visuelles. Les élèves explorent les différents objectifs que remplissent les apps.

**Leçon 2 — Mise en mouvement : Création de séquences.** Les élèves constatent l'importance de l'ordre lors de la mise en séquence des instructions, apprennent que l'on peut utiliser les mêmes actions et les réorganiser pour créer une nouvelle séquence, et mettent au point une danse originale. Ils découvrent de nouvelles commandes et résolvent des problèmes plus complexes dans des apps de programmation. Les élèves comparent différentes apps aidant les utilisateurs à découvrir de nouvelles idées.

**Leçon 3 — Mise en décision : Séquençement modulable.** Les élèves apprennent que l'ordre de certaines étapes d'une séquence peut être modifié, puis ils créent leurs propres séquences modulables. Ils cherchent différentes solutions pour résoudre les puzzles et partagent leurs solutions de programmation avec les autres élèves. Les élèves explorent des apps permettant aux utilisateurs d'entrer en contact avec des membres de leur environnement social.

**Leçon 4 — Mise en répétition : Boucles.** Les élèves identifient des boucles dans des situations du quotidien et explorent leur créativité en créant des boucles de percussions corporelles. Ils voient comment les boucles sont représentées en programmation et s'en servent pour simplifier l'écriture de leur code. Les élèves découvrent la conception d'interfaces utilisateur et en quoi celles-ci peuvent rendre une app à la fois ludique et conviviale.

**Leçon 5 — Mise en correction : Débogage.** Pour comprendre l'importance de la persistance et être capables de déboguer dans différentes situations, les élèves révisent et appliquent leurs nouvelles compétences en programmation pour relever un défi et déboguer le travail des autres élèves. Ils s'entraînent à anticiper le résultat de leur code et à identifier les bugs lorsque l'exécution ne se déroule pas comme prévu. Les élèves se lancent dans la conception d'apps pour aider à résoudre un problème.

**Leçon 6 — Mise en action : Événements et actions.** Les élèves découvrent comment les événements peuvent rendre le jeu sur l'app et le code plus captivants et plus réactifs, et apprennent à programmer en intégrant des événements. Ils réfléchissent à la manière dont nous provoquons des événements dans la vie quotidienne et créent une télécommande de robot pour s'entraîner à appeler des événements. Les élèves commencent à concevoir leurs propres apps.

**Leçon 7 — Mise en condition : Instructions if.** Les élèves découvrent les instructions conditionnelles et apprennent à identifier des instructions if dans leur vie quotidienne. Ils réfléchissent à des instructions if qui servent de règles dans des jeux de société qu'ils connaissent. Les élèves programment avec des instructions if, et apprennent à faire en sorte que leur code s'adapte aux conditions de l'environnement. Les élèves utilisent des diagrammes de flux pour présenter le fonctionnement de leurs apps.

**Leçon 8 — Mise en résolution : Algorithmes.** En s'appuyant sur toutes les connaissances acquises jusqu'à présent, les élèves conçoivent des algorithmes intégrant une série d'étapes permettant de résoudre un problème. Ils commencent par des recettes simples et continuent jusqu'à concevoir et programmer leur propre labyrinthe. Les élèves créent des prototypes de leurs apps et réalisent un projet final en documentant le processus de conception d'apps qu'ils ont suivi.

# Plan du cours (suite)

## Débuter en programmation 2

Dans Débuter en programmation 2, les élèves découvrent les concepts essentiels de la programmation et s'entraînent à penser comme des programmeurs. Ils apprennent à utiliser les algorithmes, les fonctions, les boucles, les instructions conditionnelles, les variables, et découvrent les bases de la conception d'une interface utilisateur. Les élèves vont travailler seuls et en groupes pour renforcer leurs compétences en programmation à travers la résolution de véritables problèmes de code, la vérification des lignes de code des autres élèves et la conception de programmes pour plusieurs types de robots. Ils vont également mettre en pratique leurs compétences dans Tynker, en résolvant toute une série de problèmes et en appliquant les concepts appris lors des activités en classe. Des activités de conception facultatives guident les élèves tout au long d'un processus de conception en vue de conceptualiser et de prototyper une idée d'app permettant de résoudre un problème affectant la classe ou l'école.

**Leçon 0 — Premiers pas.** Découvrez ce que vos élèves savent des apps et de la programmation, présentez-leur une app de programmation visuelle comme Tynker et préparez leur journal numérique avec une app comme Seesaw. Les élèves découvrent un défi de conception d'apps.

**Leçon 1 — Réfléchir par étapes : Résolution de problèmes avec des algorithmes.** Les élèves découvrent les algorithmes sous la forme d'un ensemble d'instructions permettant de résoudre un problème ou de réaliser une tâche. Dans l'app Tynker, les élèves font éclore le dragon de leur choix, puis créent des algorithmes pour résoudre des puzzles et améliorer leurs compétences en séquençement. Les élèves conçoivent et testent des algorithmes au cours des activités en classe. Les élèves commencent à réfléchir à des apps pouvant contribuer à résoudre un problème.

**Leçon 2 — Réfléchir par solutions : Débogage.** Les élèves découvrent comment identifier et corriger les erreurs dans leurs algorithmes et leur code. Dans Tynker, ils modifient les algorithmes contenant des bugs afin d'aboutir à un programme valide et capable de résoudre les problèmes. Les élèves découvrent le rôle que jouent les claviers dans les apps et comment ils peuvent en faire usage pour leurs propres idées d'apps.

**Leçon 3 — Réfléchir par boucles : Boucles for.** Après avoir découvert les boucles en tant que motifs à répétition, les élèves conçoivent et testent un algorithme dans le but de créer un Serpent à boucles. Dans Tynker, ils utilisent des boucles for pour résoudre des puzzles en identifiant leurs motifs. Les élèves réfléchissent à la façon dont leur app pourrait utiliser l'appareil photo et le micro intégrés.

**Leçon 4 — Réfléchir par blocs : Composition et décomposition.** Les élèves décomposent une chorégraphie en une série de mouvements de danse afin de définir un algorithme pour une musique de Gobelet musical. Dans Tynker, les élèves résolvent des problèmes en les décomposant en sous-problèmes moins complexes. Les élèves réfléchissent à la façon dont l'écran tactile peut rendre leur app plus interactive.

**Leçon 5 — Réfléchir par ensembles : Abstraction.** Les élèves classent les objets dans des groupes et expliquent leur logique pour explorer la similarité et la généralisation. Dans Tynker, ils utilisent l'abstraction pour repérer des similitudes entre les problèmes. Les nouveaux outils de programmation dont ils disposent leur permettent de résoudre des puzzles de plus en plus complexes. Les élèves réfléchissent à la façon d'utiliser des outils tels que le Bluetooth pour se connecter à des appareils situés à proximité.

**Leçon 6 — Réfléchir par motifs : Assemblage de fonctions.** Les élèves mettent au point un ensemble de tâches à accomplir pour un robot commandé. Ils en décomposent les fonctions, puis échangent leurs algorithmes avec leurs camarades pour voir s'ils obtiennent le résultat escompté. Dans Tynker, les élèves utilisent des noms et appellent des fonctions pour réutiliser des ensembles de fonctions et gagner en efficacité dans l'écriture de leur code. Les élèves découvrent les types de données auxquelles leurs apps pourraient se connecter par GPS.

**Leçon 7 — Réfléchir par conditions : Instructions conditionnelles.** Les élèves se lancent dans un voyage d'aventure virtuel. Ils déterminent leur destination à partir d'un ensemble de conditions préalables illustrées par des instructions if. Dans Tynker, ils utilisent des instructions if pour gérer la prise de décision et les choix dans les puzzles. Les élèves réfléchissent à des moyens novateurs de rendre leurs apps véritablement uniques.

# Plan du cours (suite)

## **Leçon 8 — Réfléchir par cycles : Boucles while et boucles imbriquées.**

Les élèves gèrent un stand de beignets virtuel et utilisent des boucles while et des boucles imbriquées pour créer les algorithmes destinés au robot à beignets, qui réalise le glaçage des beignets pour tous les clients. Dans Tynker, les élèves utilisent les boucles pour raccourcir leur code. Les élèves constituent des équipes de conception d'applications et commencent à créer le prototype d'une application issue de leur imagination.

## **Leçon 9 — Réfléchir par adaptation : Variables, paramètres d'entrée et de sortie.**

Les élèves utilisent des variables pour concevoir un algorithme dans le cadre d'un concours de slam et de poésie, qu'ils interprètent en intégrant les données fournies par la classe (entrées). Dans Tynker, ils utilisent des variables pour résoudre des puzzles de plus en plus compliqués en s'appuyant sur toutes les compétences de programmation qu'ils ont acquises jusqu'ici. Les élèves réalisent des interviews auprès des utilisateurs pour mieux cibler le public de leur application.

## **Leçon 10 — Réfléchir par la pratique : Conception de l'interface**

**utilisateur.** Les élèves étudient ce qui caractérise une interface bien conçue, et créent un panneau pour leur établissement. Dans l'activité Tynker, ils utilisent toutes les compétences qu'ils ont acquises en complétant les leçons de Débuter en programmation 2. Les élèves découvrent les notions d'interface utilisateur et d'expérience utilisateur, et créent un mood board pour la conception de leur application. Ils rédigent un argumentaire pour leur application dans le projet final.



# Informations complémentaires

## Télécharger les ressources Débuter en programmation

- [Tynker](#)
- [codeSpark Academy](#)
- [Débuter en programmation 1](#)
- [Débuter en programmation 2](#)

## Télécharger les ressources Swift Playgrounds

- [Learn to Code 1 & 2: iTunes U Course](#)
- [Apprendre à coder 1 et 2 : Guide d'enseignement](#)
- [Learn to Code 3: Teacher Guide](#)
- [App Swift Playgrounds](#)

## Télécharger les guides App Development with Swift

- [Introduction au développement d'applications avec Swift](#)
- [Introduction au développement d'applications avec Swift : Guide d'enseignement](#)
- [App Development with Swift](#)
- [App Development with Swift: Teacher Guide](#)

## Ressources complémentaires

- En savoir plus sur le programme [Le code à la portée de tous](#).
- En savoir plus sur [Swift](#).
- En savoir plus sur [Xcode](#).
- Communiquez avec d'autres enseignants sur les [forums de développeurs Apple](#).
- En savoir plus sur [codeSpark Academy](#).
- En savoir plus sur [Tynker](#).

Les fonctionnalités sont sujettes à modification. La disponibilité des fonctionnalités peut varier en fonction des zones géographiques et des langues.

© 2017 Apple Inc. Tous droits réservés. Apple, le logo Apple, iTunes U, Keynote, et Xcode sont des marques d'Apple Inc., déposées aux États-Unis et dans d'autres pays. Swift et Swift Playgrounds sont des marques d'Apple Inc. Les autres noms de produits et de sociétés mentionnés dans ces pages peuvent être des marques déposées de leurs sociétés respectives. Les caractéristiques des produits sont susceptibles d'être modifiées sans préavis. Les informations contenues dans ce document sont fournies à titre indicatif uniquement ; Apple n'assume aucune responsabilité quant à leur utilisation. Septembre 2017