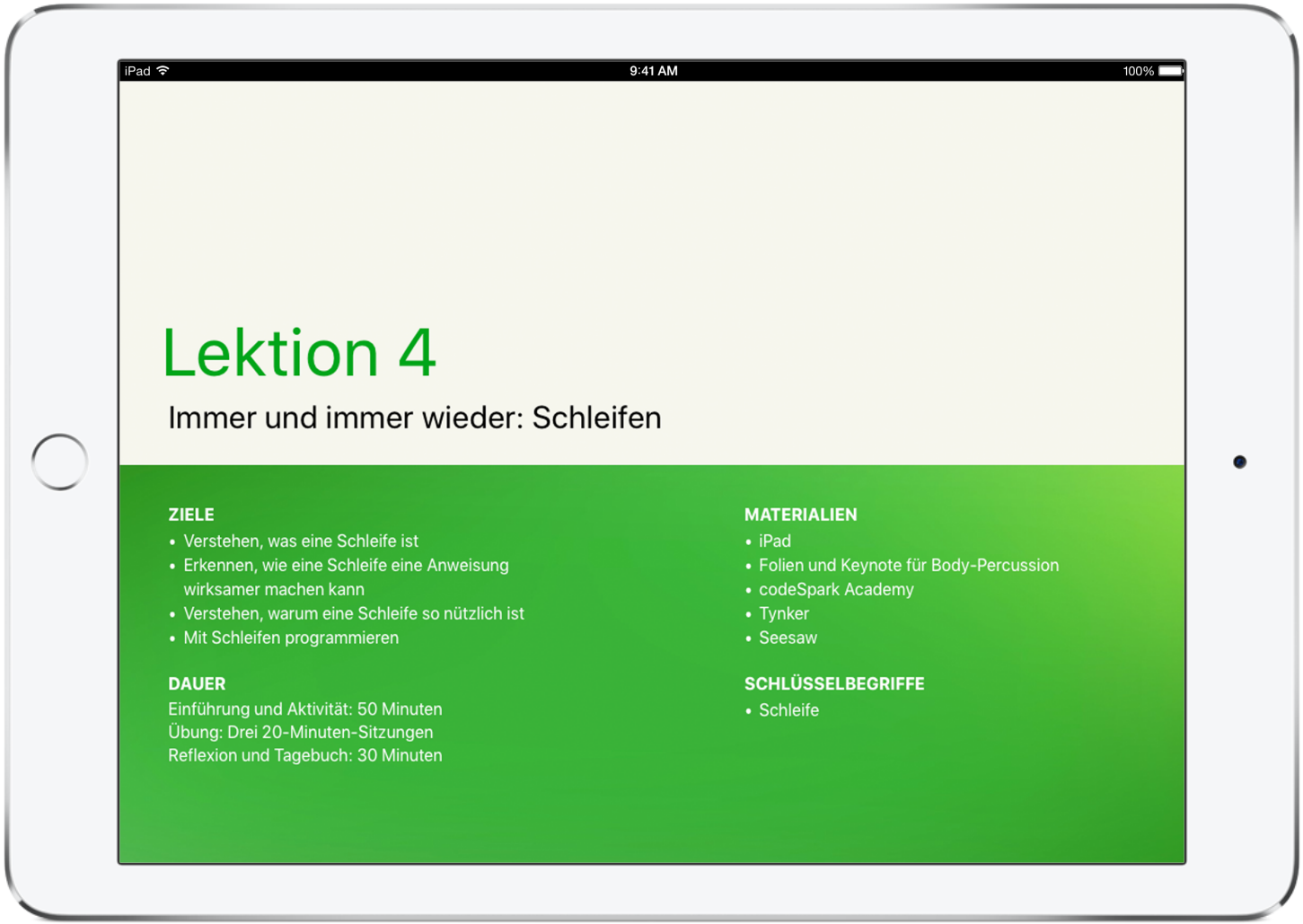
Erste Schritte mit Code – Lehrplanführer

September 2017

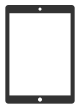
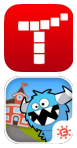
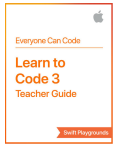


Jeder kann programmieren

Technologie hat eine Sprache: Sie heisst Code. Und wir sind überzeugt, dass Code schreiben bzw. Programmieren eine wichtige Fähigkeit ist. Durch Programmieren lernt man, wie man Probleme lösen und auf kreative Weise zusammenarbeiten kann. Und es hilft dabei, Apps zu entwickeln, die Ideen Wirklichkeit werden lassen. Wir glauben, dass alle die Chance haben sollten, etwas zu erschaffen, das die Welt verändern kann. Deshalb haben wir ein neues Programm mit Tools und Ressourcen entwickelt, mit dem alle Programmieren lernen und unterrichten können.

Jeder kann programmieren – Lehrplan

Das Programm „Jeder kann programmieren“ umfasst eine Reihe von Ressourcen, die Lernende auf ihrem Weg von absoluten Programmierneulingen hin zu Entwicklern ihrer ersten Apps begleiten. Die folgende Tabelle gibt einen Überblick über alle kostenlos verfügbaren Lehr- und Lernressourcen.

Lehrplan	Gerät	Zielgruppe	App	Voraussetzungen	Überblick	Lernmaterialien	Unterstützende Ressourcen	Anzahl enthaltener Unterrichtsstunden
		Kindergarten bis 2. Klasse		Keine	Anfangen, wie Programmierer zu denken, durch praktische Auseinandersetzung mit Programmierkonzepten anhand von visuell-basierten Apps.	• Lektionen in der codeSpark Academy App • Tynker Kurs „Weltraumkadett“	• Erste Schritte mit Code 1: Lehrerhandbuch	30 Stunden, inklusive Lehrerhandbuch und Lektionen in der App
		3. bis 5. Klasse		Keine	Grundlegende Programmierkonzepte erkunden und mit visuell-basierten Apps üben, wie ein Programmierer zu denken.	• Tynker Kurs „Drachenzauber“	• Erste Schritte mit Code 2: Lehrerhandbuch	36 Stunden, inklusive Lehrerhandbuch und Lektionen in der App
		Ab Mittelstufe		Keine	Grundlegende Programmierkonzepte mit echtem Swift Code erlernen.	• Swift Playgrounds App • Lektionen in „Programmieren lernen 1 & 2“ • iTunes U Kurs	• Programmieren lernen 1 & 2: Lehrerhandbuch • Badges für Swift Playgrounds im Apple Teacher Learning Center	Bis zu 85 Stunden, inklusive Lehrerhandbuch und Lektionen in „Programmieren lernen 1 & 2“
		Ab Mittelstufe		Programmieren lernen 1 & 2	Programmierkenntnisse erweitern und Denkweisen eines App-Entwicklers aneignen.	• Swift Playgrounds App • Lektionen in „Learn to Code 3“	• Learn to Code 3: Teacher Guide	Bis zu 45 Stunden, inklusive Lehrerhandbuch und Lektionen in „Learn to Code 3“
		Oberstufe und Hochschule		Keine	Praktische Erfahrung mit den Tools, Techniken und Konzepten sammeln, die benötigt werden, um eine einfache iOS App von Grund auf zu erstellen.	Buch „Einführung in die App-Entwicklung mit Swift“ und Projektdateien	• Einführung in die App-Entwicklung mit Swift: Lehrerhandbuch	90 Stunden
		Oberstufe und Hochschule		Keine	Aufbau von Grundlagen in Swift, UIKit und Networking durch praktische Übungen und geführte Projekte. Am Ende des Kurses können die Schüler eine selbst erdachte App programmieren.	Buch „App Development with Swift“ und Projektdateien	• App Development with Swift: Teacher Guide	180 Stunden

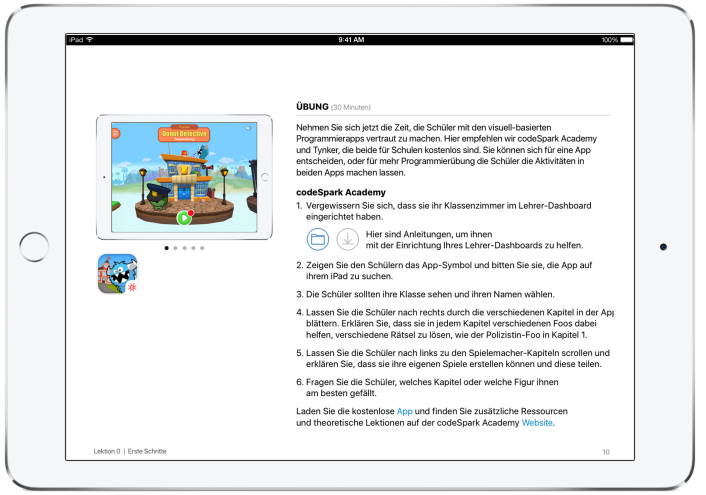
Überblick

Die frühen Schuljahre sind eine hervorragende Zeit, um Programmierkonzepte als eine Art der Auseinandersetzung mit der alltäglichen und digitalen Welt einzuführen und grundlegende Fähigkeiten im rechnerischen Denken zu entwickeln. Apps wie CodeSpark Academy und Tynker, die speziell für jüngere Lernende konzipiert sind, verwenden visuell-basierte Programmierrätsel, um das Problemlösungsvermögen zu entwickeln und Ausdauer und Kreativität zu fördern. codeSpark Academy wurde für Schüler im Alter zwischen fünf und sieben Jahren konzipiert. Das Spiel funktioniert ohne Worte, sodass es sich auch für Schüler eignet, die noch nicht lesen können, eine Leseschwäche haben oder Englisch als Fremdsprache lernen. Mit Tynker fangen Schüler im Alter von 5 bis 11 Jahren an, mit visuellen Blöcken zu experimentieren, und fahren dann mit textbasierter Programmierung fort, indem sie Rätsel lösen und Projekte erstellen.

Im Unterricht

Tynker und codeSpark Academy helfen Ihnen in Kombination mit den Lektionen in den Lehrerhandbüchern zu „Erste Schritte mit Code“ dabei, das Programmieren in den frühen Grundschulunterricht einzubringen. Die Lektionen behandeln wichtige Programmierkonzepte, zeigen aber auch, dass Programmieren eine Art zu denken ist, die sich auf weitere Fächer und den Alltag anwenden lässt.

Die Lehrerhandbücher bieten die Unterstützung, die Sie benötigen, um Ihren Schülern beim Lösen der Programmierrätsel zu helfen – unabhängig von Ihrer Programmiererfahrung. Sie enthalten Ausbauaktivitäten, Aktivitäten zum App-Design, Fragen zur Reflexion, Ideen für das Tagebuch, ein Bewertungsschema und mehr, damit Sie Schülern bei der Vertiefung des Materials helfen können. Sie können die Lektionen als einzelnen Block oder in Abschnitten unterrichten. Korrelationsdiagramme in den Anhängen ermöglichen eine erste Angleichung der Lektionen an die vorläufigen Informatikstandards der Primar- und Sekundarstufe der Computer Science Teachers Association (CSTA) für Stufe 1.



Wichtige Features

codeSpark Academy

Lernen. In diesem Spiel, das ohne Worte funktioniert, lösen die Schüler Rätsel, um grundlegende Informatikkonzepte wie Sequenzieren, Schleifen, bedingte Anweisungen und mehr kennenzulernen.

Erstellen. Die Schüler wenden ihr Wissen an, indem Sie im „Spielemacher“ ihre eigenen Projekte programmieren.

Selbstgesteuert. Zum Unterrichten, Lernen oder Spielen ist keine Programmiererfahrung nötig. Der Lehrplan und das Lehrer-Dashboard für Fortschrittsberichte stehen Lehrkräften kostenlos in 10 Sprachen zur Verfügung.

Tynker

Programmierumgebung. Die Schüler arbeiten sich in ihrem eigenen Tempo durch strukturierte Programmierrätsel, um Konzepte zu lernen und auf kreative Weise anzuwenden.

Automatische Beurteilung. Ein Lehrer-Dashboard beurteilt anhand von Rätseln, Tests und Codeanalysen, wie gut die Schüler die Fähigkeiten beherrschen.

Swift Feature. Während die Schüler Rätsel lösen, können Sie zwischen visuellen Blöcken und Swift Blöcken hin und her wechseln und sich so als Vorbereitung auf künftige Programmieraufgaben mit Swift vertraut machen.

Erste Schritte – Lehrerhandbücher

Dateien zum Laden. Vorlagendateien für Schüleraktivitäten und Keynote Präsentationen unterstützen das Lernen der Schüler im Unterricht.

Lösungsschlüssel. Dank Lösungen für Tynker und codeSpark Academy Rätsel können Sie Schülern, die feststecken, ganz einfach weiterhelfen.

Beispiele für Schülerarbeiten. Sehen Sie, wie die Aktivitäten aussehen können.

Reflexionen. Diese Fragen und Anregungen für Diskussionen in der Klasse helfen Ihnen, die Verbindung zwischen der Anwendung des Konzepts inner- und ausserhalb einer Programmierumgebung zu überprüfen und zu festigen.

Tipps und Beispiele. Ideen für die Erweiterung oder Vereinfachung der Lektionen sind durchgehend enthalten.

Aktivitäten zum App-Design. In diesen Lektionen werden die Schüler durch den Designprozess geführt. Dabei lernen sie, wie sie eine App-Idee entwickeln und den Prototyp für eine App erstellen, mit der sie ein Problem in ihrer Klasse oder Schule lösen können.

Kursgliederungen

Erste Schritte mit Code 1

Durch das interaktive, praktische Erkunden von Programmierkonzepten im Kontext alltäglicher Situationen beginnen die Schüler, wie Programmierer zu denken. Sie lernen etwas über Befehle, Sequenzen, Schleifen, Ereignisse und Algorithmen. In Zusammenarbeit üben die Schüler das Vorhersagen der Ergebnisse Ihres Codes sowie das Debuggen ihres eigenen Codes und des Codes von anderen. Zudem üben Sie in visuell-basierten Programmierapps das Anwenden ihrer Fähigkeiten, indem sie Rätsel lösen und ihre eigenen Kreationen entwerfen. In optionalen Übungen zum App-Design werden die Schüler durch den Entwicklungsprozess geführt. Dabei lernen sie, wie sie eine App-Idee entwickeln und den Prototyp für eine App erstellen, mit der sie ein Problem in ihrer Klasse oder Schule lösen können.

Lektion 0 – Erste Schritte. Finden Sie heraus, was die Schüler bereits über Apps und das Programmieren wissen, richten Sie die Arbeitswand der Klasse ein und stellen Sie den Schülern die wichtigsten Apps vor, die sie in den Lektionen nutzen werden. Die Schüler lernen die verschiedenen Rollen in einem App-Designteam kennen.

Lektion 1 – Die richtige Reihenfolge: Einführung in die Sequenzierung. Die Schüler erkunden alltägliche Sequenzen, konstruieren auf Basis einer bekannten Geschichte eine Sequenz und lösen in visuell-basierten Programmierapps mithilfe einfacher Sequenzen Rätsel. Die Schüler erfahren, dass Apps für unterschiedliche Zwecke entwickelt werden.

Lektion 2 – Schritt für Schritt: Sequenzen erstellen. Indem sie erforschen, wie wichtig die Reihenfolge beim Sequenzieren von Anweisungen ist, lernen die Schüler, dass dieselben Anweisungen anders geordnet eine andere Sequenz ergeben. Dabei kreieren sie ihren eigenen verrückten Tanz. Sie lernen weitere Befehle kennen und lösen komplexere Probleme in den Programmierapps. Die Schüler vergleichen verschiedene Apps, die dem Benutzer neues Wissen vermitteln.

Lektion 3 – Freie Wahl: Flexible Sequenzierung. Die Schüler lernen, dass einige Schritte in einer Sequenz flexibel angeordnet werden können, und erstellen ihre eigenen flexiblen Sequenzen. Sie erkunden die verschiedenen Arten, auf die ein Rätsel gelöst werden kann, und teilen Ihre Codelösungen mit anderen Schülern. Die Schüler befassen sich mit Apps, die dem Benutzer helfen, mit anderen Menschen in seinem schulischen Umfeld zu kommunizieren.

Lektion 4 – Immer und immer wieder: Schleifen. Die Schüler machen Schleifen in alltäglichen Situationen ausfindig und werden beim Erstellen ihrer eigenen Body-Percussion-Schleifen kreativ. Sie erkunden, wie Schleifen beim Programmieren dargestellt werden, und verwenden Schleifen, um ihren Code zu optimieren und zu vereinfachen. Die Schüler erfahren, wie sie die Benutzeroberfläche gestalten und eine unterhaltsame und benutzerfreundliche App entwickeln.

Lektion 5 – Fehler beheben: Debugging. Um die Wichtigkeit von Beharrlichkeit und Debugging in unterschiedlichen Zusammenhängen zu verstehen, prüfen die Schüler ihre neu erworbenen Programmierkenntnisse und wenden sie an, um eine Herausforderung zu meistern und die Lösungen anderer Schüler zu debuggen. Sie üben das Vorhersagen des Ergebnisses ihres Codes und spüren Fehler auf, wenn der Code nicht wie vorgesehen ausgeführt wird. Die Schüler befassen sich eingehender mit der Entwicklung von Apps, die helfen, ein Problem zu lösen.

Lektion 6 – Aufforderungen: Ereignisse und Aktionen. Die Schüler erkunden, wie Ereignisse das Spielen ihrer App ermöglichen und Code spannender und reaktionsschneller machen können. Dabei lernen sie, mit Ereignissen zu programmieren. Sie machen sich Gedanken darüber, wie wir im Alltag Ereignisse anstossen, und erstellen eine Roboter-Fernbedienung, um das Aufrufen von Ereignissen zu üben. Die Schüler beginnen mit der Entwicklung eigener Apps.

Lektion 7 – Es geht, wenn die Regel befolgt wird: IF-Anweisungen. Die Schüler lernen bedingte Anweisungen kennen und erfahren, wie sie IF-Anweisungen im Alltag erkennen. Sie denken über die IF-Anweisungen nach, die in bekannten Brettspielen als Spielregeln fungieren. Die Schüler programmieren mit IF-Anweisungen, um ihren Code besser auf Umgebungsbedingungen abzustimmen. Die Schüler stellen die Funktionsweise ihrer Apps mithilfe von Flussdiagrammen dar.

Lektion 8 – Lösungen finden: Algorithmen. Die Schüler führen alles bisher Gelernte zusammen und erstellen Algorithmen, die mithilfe einer Reihe von Schritten ein Problem lösen. Sie beginnen mit einfachen Rezepten und fahren dann mit dem Entwerfen und Programmieren ihres eigenen Irrgartenspiels fort. Die Schüler erstellen Prototypen ihrer Apps und führen ein Abschlussprojekt durch, in dem sie ihre Schritte bei die Entwicklung ihrer Apps dokumentieren.

Kursgliederungen (Forts.)

Erste Schritte mit Code 2

In „Erste Schritte mit Code 2“ erkunden Schüler grundlegende Programmierkonzepte und lernen, wie ein Programmierer zu denken. Sie erfahren etwas über Algorithmen, Funktionen, Schleifen, bedingte Anweisungen und Variablen und lernen die Designgrundlagen für Benutzeroberflächen kennen. In Gruppen- und Einzelarbeit festigen die Schüler ihre Programmierkenntnisse durch das Lösen realer Programmierprobleme, das gegenseitige Testen ihrer Codes und das Entwerfen von Programmen für eine Reihe von Robotern. Zudem wenden sie diese Fähigkeiten in Tynker an, indem sie eine Reihe von Problemen lösen und die in Unterrichtsaktivitäten gelernten Konzepte anwenden. In optionalen Übungen zum App-Design werden die Schüler durch den Entwicklungsprozess geführt. Dabei lernen sie, wie sie eine App-Idee entwickeln und den Prototyp für eine App erstellen, mit der sie ein Problem in ihrer Klasse oder Schule lösen können.

Lektion 0 – Erste Schritte. Finden Sie heraus, was die Schüler bereits über Apps und das Programmieren wissen, machen Sie sie mit einer visuell-basierten Programmierapp wie Tynker vertraut und richten Sie ihre digitalen Tagebücher mit einer App wie Seesaw ein. Die Schüler lernen die App-Design-Challenge kennen.

Lektion 1 – In Schritten denken: Probleme mit Algorithmen lösen. Die Schüler erkennen, dass Algorithmen eine Reihe von Anweisungen zum Lösen eines Problems oder Ausführen einer Aufgabe sind. In der Tynker App lassen die Schüler einen Drachen ihrer Wahl schlüpfen und erstellen dann Algorithmen, um Rätsel zu lösen, während sie Sequenzierfähigkeiten erlernen. In Unterrichtsaktivitäten entwerfen und testen die Schüler Algorithmen. Die Schüler beginnen damit, Ideen für Apps zu sammeln, mit denen ein Problem gelöst werden kann.

Lektion 2 – In Fehlerbehebungen denken: Debugging. Die Schüler beschäftigen sich damit, Fehler in ihren Algorithmen und ihrem Code zu finden und zu beheben. In Tynker ändern sie fehlerhafte Algorithmen, um ein korrektes Programm zum Lösen des Rätsels zu erstellen. Die Schüler erfahren, wie Tastaturen in Apps verwendet werden, und erarbeiten Ideen, um Tastaturen in ihre eigenen App-Entwürfe einzubinden.

Lektion 3 – In Kreisen denken: Schleifen suchen. Nachdem ihnen Schleifen als sich wiederholende Muster vorgestellt wurden, entwerfen und testen die Schüler einen Algorithmus, um eine verdrehte Schlange zu erstellen. In Tynker suchen sie nach Mustern und nutzen for-Schleifen, um Rätsel zu lösen. Die Schüler sammeln Ideen, wie sie integrierte Features wie die Kamera oder das Mikrofon in ihre App einbinden könnten.

Lektion 4 – In Häppchen denken: Komposition und Dekomposition. Um einen Algorithmus für ein Becherlied zu entwickeln, zerlegen die Schüler den Ablauf in kleinere Teile. In Tynker lösen sie Probleme, indem sie sie in kleinere Teilprobleme zerlegen. Die Schüler überlegen, wie sie mit dem Touchscreen ihre App interaktiver gestalten können.

Lektion 5 – In Kategorien denken: Abstraktion. Die Schüler erkunden die Begriffe Ähnlichkeit und Generalisierung, während sie Objekte in Sets anordnen. Anschliessend erklären sie ihre Überlegungen. In Tynker nutzen sie Abstraktion, um Ähnlichkeiten zwischen Problemen auszumachen, und lösen mit all ihren neuen Programmierwerkzeugen zunehmend komplexere Rätsel. Die Schüler diskutieren über den Einsatz von Tools wie Bluetooth, um sich mit Geräten in der Nähe zu verbinden.

Lektion 6 – In Mustern denken: Funktionen bilden. Die Schüler erstellen einen Ablauf für einen Command Bot, wobei sie den Ablauf in Funktionen zerlegen. Anschliessend tauschen sie Algorithmen aus, um die vorhergesagten und tatsächlichen Ergebnisse zu testen. In Tynker nutzen die Schüler Namen und rufen Funktionen auf, während sie eine Reihe von Anweisungen wiederverwenden, um effizienter zu programmieren. Die Schüler lernen die verschiedenen Arten von Daten kennen, die ihre App mithilfe von GPS erfassen könnte.

Kursgliederungen (Forts.)

Lektion 7 – In Voraussetzungen denken: Bedingte Anweisungen. Die Schüler gehen auf ein virtuelles Reiseabenteuer, wobei sie ihr Ziel mit einer Reihe von qualifizierenden Bedingungen mithilfe von if-Anweisungen festlegen. In Tynker nutzen sie if-Anweisungen, um in den Rätseln mit Entscheidungen und Alternativen umzugehen. Die Schüler erarbeiten innovative Möglichkeiten, um ihre Apps einzigartig zu machen.

Lektion 8 – In Zyklen denken: while-Schleifen und verschachtelte Schleifen. Die Schüler führen einen virtuellen Donut-Stand und nutzen verschachtelte und while-Schleifen, um Algorithmen für den Donut Bot zu erstellen, damit dieser ausreichend Donuts für alle Kunden glasiert. In Tynker nutzen die Schüler Schleifen, um ihren Code zu kürzen. Die Schüler bilden App-Designteams und beginnen mit der Entwicklung eines Prototypen für ihre selbst erdachte App.

Lektion 9 – In die Kiste rein und raus denken: Variablen, Ein- und Ausgabe. Die Schüler verwenden Variablen, um einen Algorithmus für eine Poetryslam-Jamsession zu entwerfen, und führen auf Basis von Publikumsbeiträgen einen Song oder Rap auf. In Tynker nutzen sie Variablen, um komplexere Rätsel zu lösen, wobei sie alle bisher gelernten Programmierkenntnisse einsetzen. Die Schüler befragen künftige Benutzer, um ihre App besser auf die Zielgruppe abstimmen zu können.

Lektion 10 – Praktisch denken: UI entwerfen. Die Schüler untersuchen, was gutes Design ausmacht, und entwerfen ein Schild für ihre Schule. In der Tynker Aktivität nutzen sie alle erworbenen Kenntnisse und schliessen die Lektionen von „Erste Schritte mit Code 2“ ab. Die Schüler erfahren mehr über die Benutzeroberfläche und das Benutzererlebnis und erstellen ein Moodboard für das Design ihrer App. Als Abschlussprojekt erstellen Sie eine Präsentation, in der sie ihre App vorstellen.

Weitere Informationen

Ressourcen zu „Erste Schritte mit Code“ laden

- Tynker
- codeSpark Academy
- Erste Schritte mit Code 1
- Erste Schritte mit Code 2

Swift Playgrounds Ressourcen laden

- Programmieren lernen 1 & 2 – iTunes U Kurs
- Programmieren lernen 1 & 2 – Lehrerhandbuch
- Learn to Code 3 – Teacher Guide
- Swift Playgrounds App

Handbücher für die App Development with Swift laden

- Einführung in die App-Entwicklung mit Swift
- Einführung in die App-Entwicklung mit Swift – Lehrerhandbuch
- App Development with Swift
- App Development with Swift: Teacher Guide

Weitere Ressourcen

- Erfahren Sie mehr über das Programm Jeder kann programmieren.
- Erfahren Sie mehr über Swift.
- Erfahren Sie mehr über Xcode.
- Vernetzen Sie sich mit anderen Lehrkräften in den Apple Developer Forums.
- Erfahren Sie mehr über codeSpark Academy.
- Erfahren Sie mehr über Tynker.

Änderungen an den Features vorbehalten. Einige Features sind u. U. nicht in allen Regionen oder Sprachen verfügbar.