Code2Learn

From Playgrounds to Curriculum

#TCEA

Facilitators

Trisha Goins

Digital Learning Coach
Coppell ISD
tgoins@coppellisd.com
@heartinlife

Apple Teacher

Brad Cook

Digital Learning Coach
Coppell ISD
bbcook@coppellisd.com
@rbradcook

Apple Teacher

Megan Kozar

Digital Learning Coach
Coppell ISD
mpettit@coppellisd.com
@MrsMeganKozar

Apple Teacher

Brian Timm

Digital Learning Coach
Coppell ISD
btimm@coppellisd.com
@BrianWTimm

Distinguished Educator

How do we increase our capacity to teach and learn how to code?

Overview

- Our Story
- Open Play
- Curriculum Exploration
- Guided Play
- CS Framework & Standards
- Planning Time

That’s Me!

That’s Me!

- I am an administrator
- I am an instructional coach
- I live outside of Texas
- I am a teacher
- I have a background in Computer Science
- I believe everyone can learn to code

Self Assessment

Coppell DLC Twitter: #coppelldlc
coppellisd.com/digitallearning
#TCEA18

Code2Learn:
From Playgrounds to Curriculum

February 5, 2018

Essential Question: How do we increase our capacity to teach and learn to code?

Self Assessment

In my classroom/campus/district, the coding exposure level is

|Broad & Deep Exposure | | | | | Moderate Exposure | | | | |Basic Exposure|

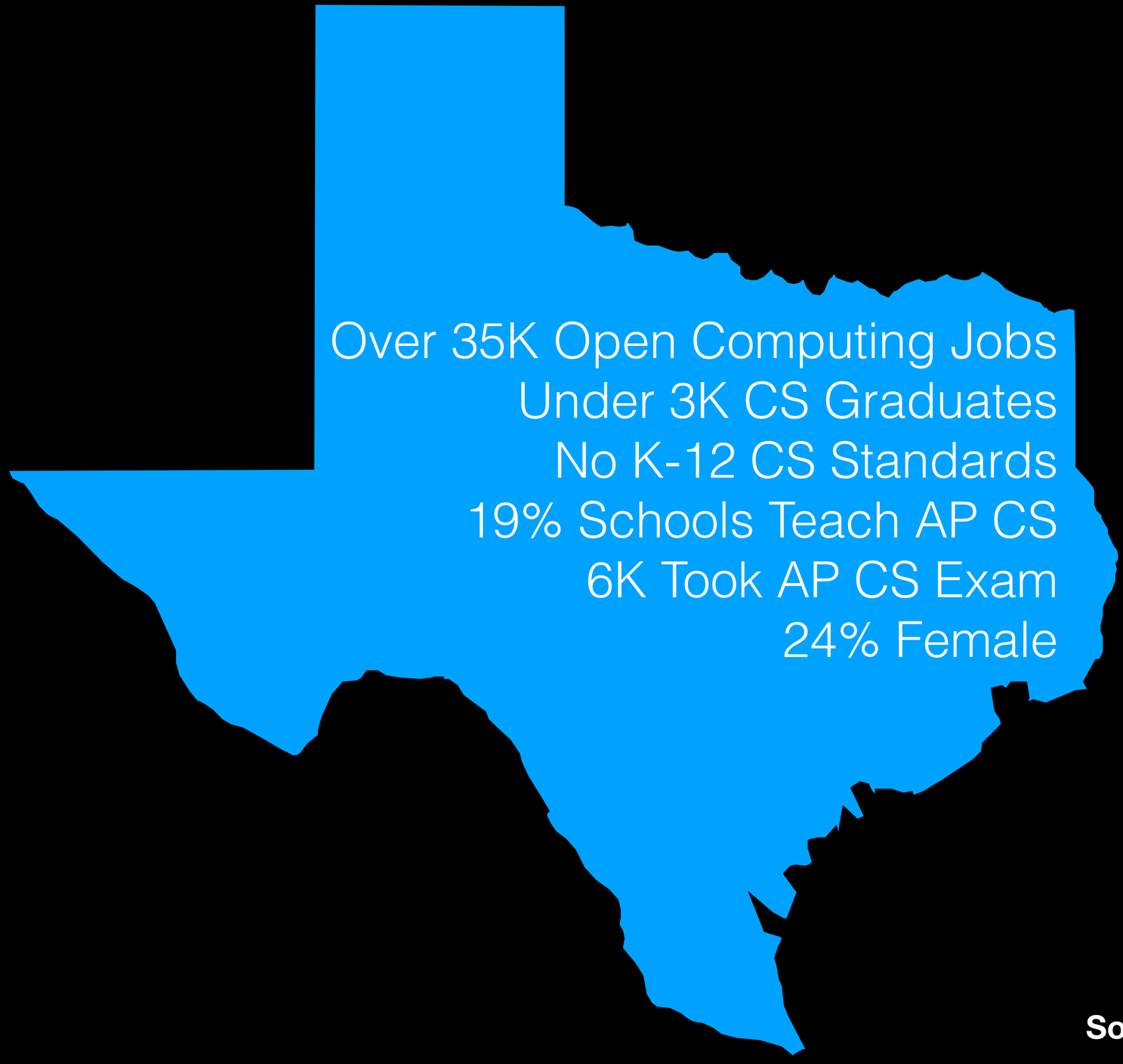
Reflection Point #1: Open Play	Reflection Point #2: Exploring Curriculum	Reflection Point #3: Guided Play

Sample K-12 Computer Science Pathways

	Broad & Deep Exposure	Moderate Exposure	Basic Exposure
Elementary School	Independent special (similar to Science, Music, Art, K to 5)	Integrated into the general classroom	Integrated into the general classroom
Middle School	Integrated into core content areas + Independent course at particular grade level	Independent course at a particular grade level	Integrated into math, science, and other core content areas
High School	Introductory Course + AP Computer Science + Specialized Courses	Introductory Course + Specialized Courses	Introductory Course

Source: k12cs.org


CS in TX

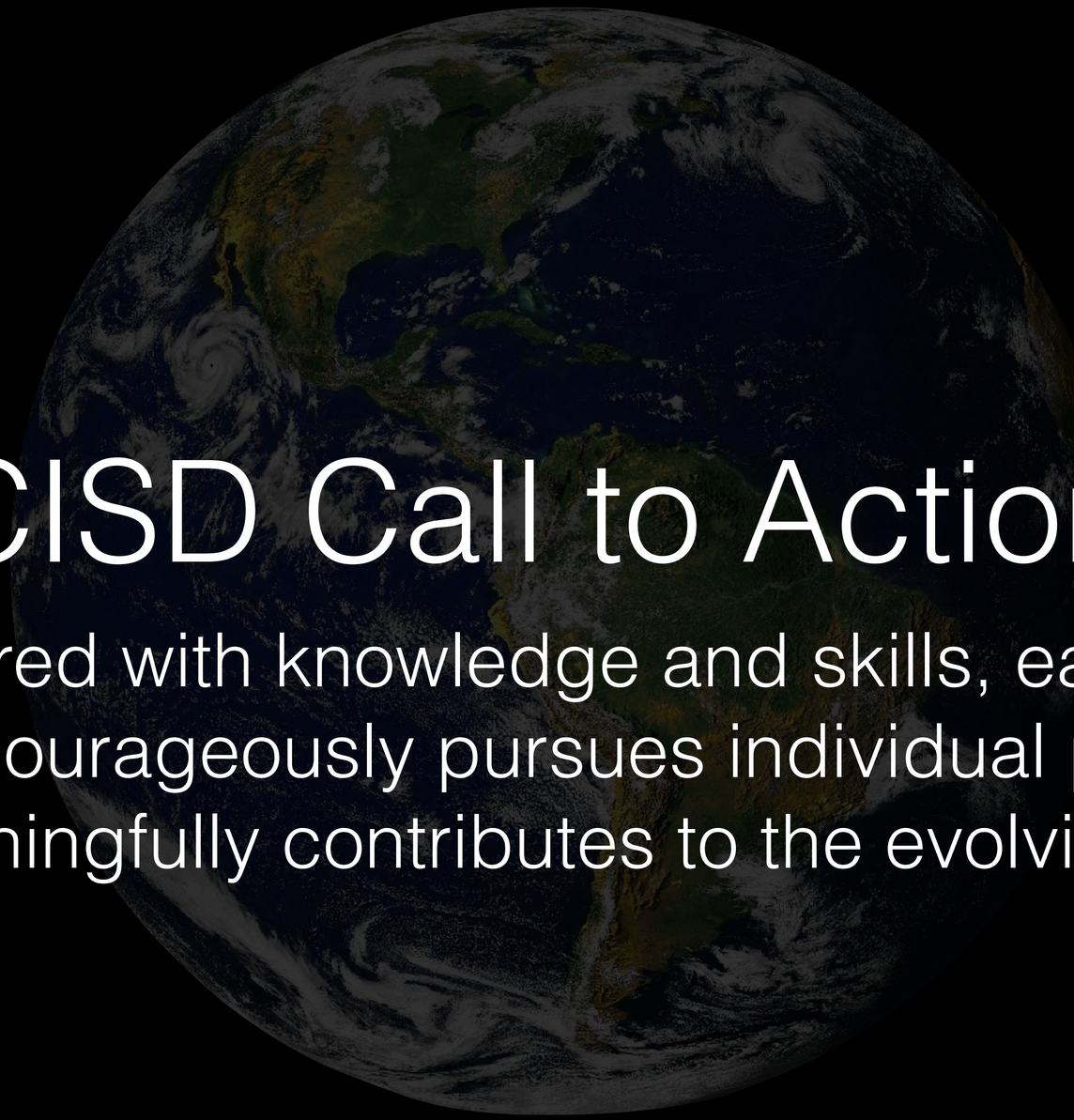


Source: code.org/promote/tx

Our Story

Think - Pair - Share

How could you meaningfully contribute to the world you grew up in?



CISD Call to Action

Empowered with knowledge and skills, each CISD learner courageously pursues individual passions and meaningfully contributes to the evolving world.

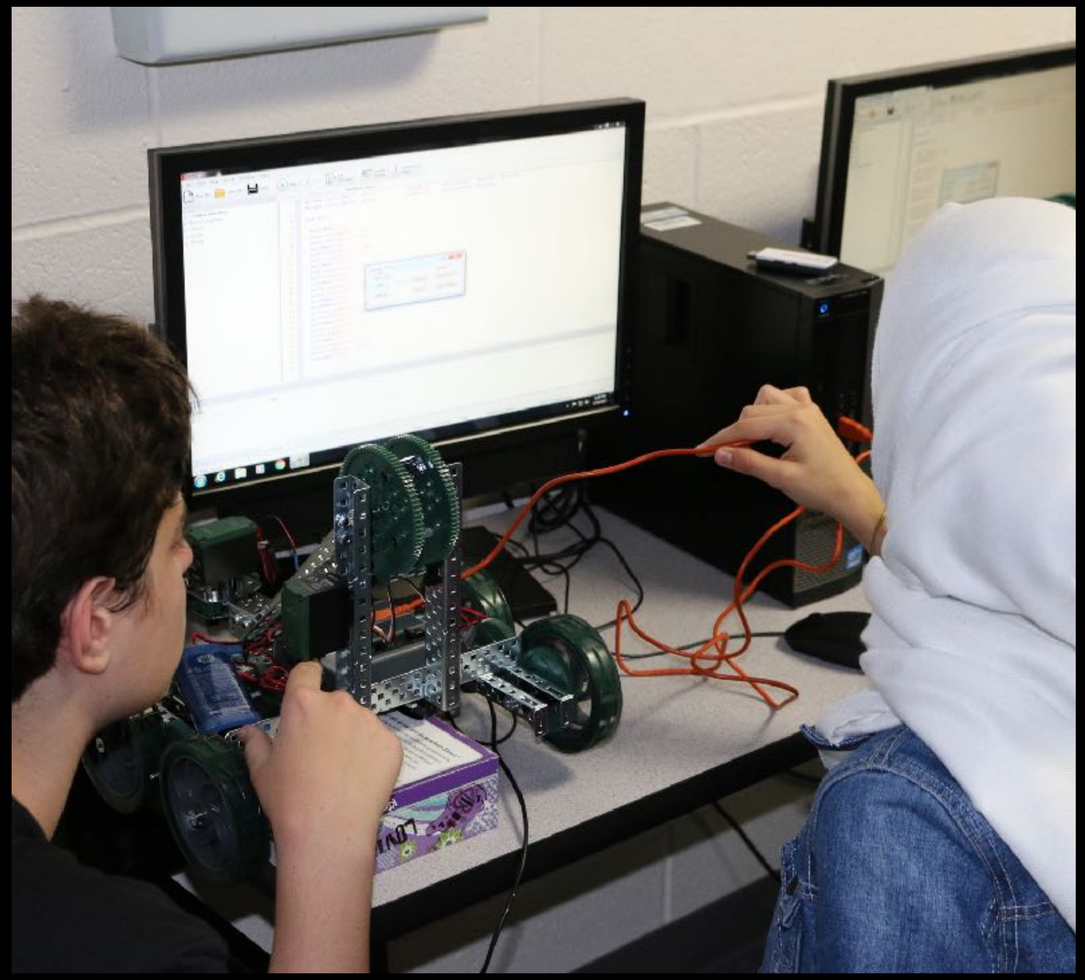
Think - Pair - Share

How can our learners contribute to the world today?

In the Classroom

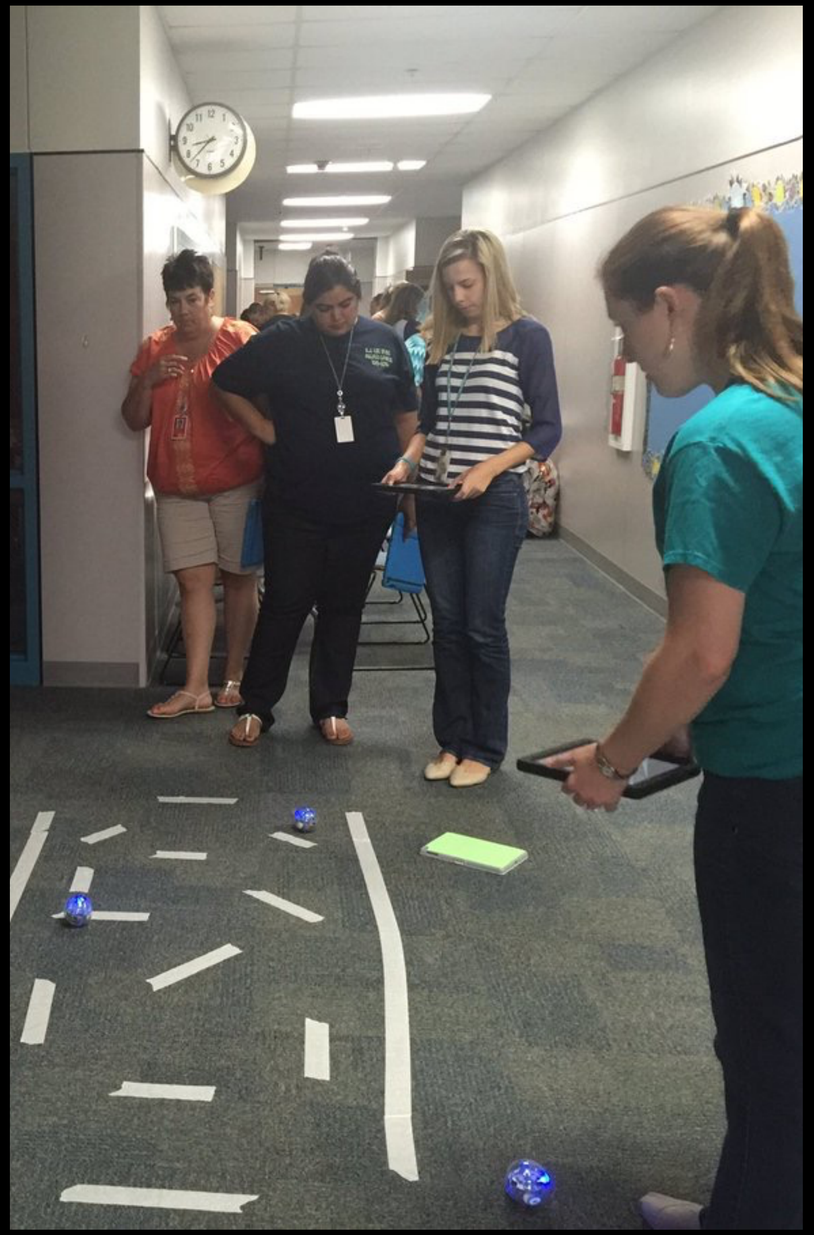
Principles of IT

- CodeHS
- Swift Playgrounds
- VEX Robotics
- HTML
- SQL



CISD

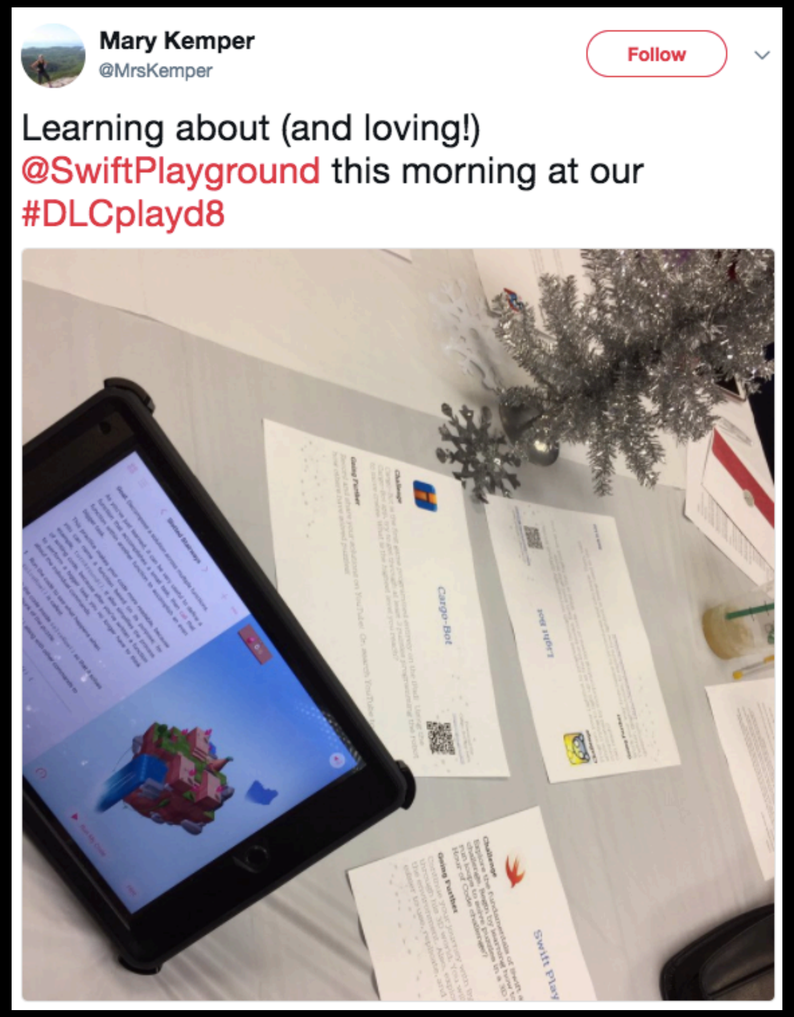
#DLCPlayD8




2ndGradeShannon @2ndgradeshannon · 17 Aug 2016
@NetZeroLee 2nd grade designers ❤️Ed our #dlcplayd8
8

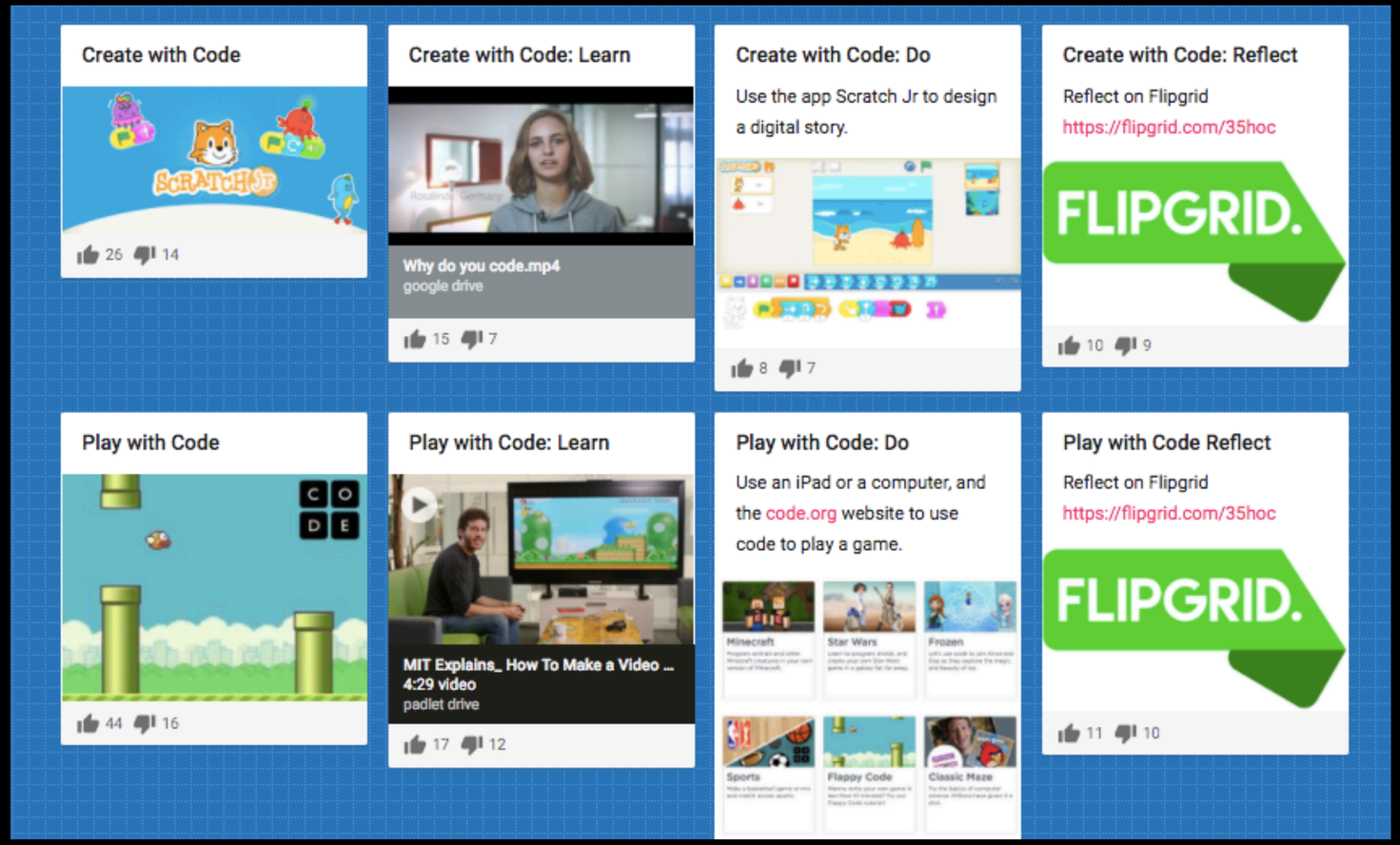

Logan Heath @MissHeathTCE · 17 Aug 2016
3rd grade is loving Osmo coding 😍 #dlcplayd8
1
13


Alli Pryor
@dlcoachalli
Following
Fun times at our #dlcplayd8 at #r10tech conference today! @coppelldlc #coppelldlc


Mary Kemper
@MrsKemper
Follow
Learning about (and loving!) @SwiftPlayground this morning at our #DLCplayd8


Tami Martins
@tkmartins
Following
The great principal race with BlueBot! #dlcplayd8 @coppelldlc
Save

Hour of Code



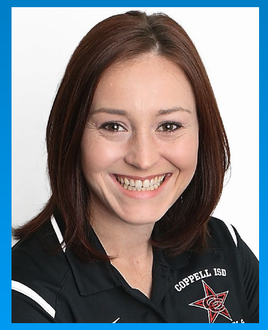
Choice Boards

K-2 | 3-5 | 6-8 | 9-12


Your Story

Screen

Your Role



Teacher

Other Roles

ITS / Media Specialist / Librarian

Campus or Central Admin

Open Play

Open Play

Online Options	Coding Tools	Apps
code.org	Sphero	Tynker
Scratch	Dash	ScratchJr
Khan Academy	Pro-Bot	Swift Playgrounds (iPad Only)
Tynker	Osmo Coding	codeSpark Academy
	Blue-Bot	

These are just a few of the possible resources you can explore during Open Play.
Feel free to explore other coding resources you find!


Reflection

Coppell DLC Twitter: #coppelldlc
coppellisd.com/digitallearning
#TCEA18

Code2Learn:
From Playgrounds to Curriculum

February 5, 2018

Essential Question: How do we increase our capacity to teach and learn to code?

Self Assessment

In my classroom/campus/district, the coding exposure level is

|Broad & Deep Exposure | | | | | Moderate Exposure | | | | |Basic Exposure|

Reflection Point #1: Open Play	Reflection Point #2: Exploring Curriculum	Reflection Point #3: Guided Play

Curriculum Exploration

Courses by
code.org

Courses

Selecting the right course for your class

Kindergarten	1st Grade	2nd Grade	3rd Grade	4th Grade	5th Grade
Course A	Course B	Course C	Course D	Course E	Course F
Pre-Reader Express Course		Express Course			

Computer Science Discoveries

Recommended for Grades 6-10

Computer Science Discoveries (CS Discoveries) is an introductory computer science course that empowers students to create authentic artifacts and engage with computer science as a medium for creativity, communication, problem solving, and fun.

View course | Lesson plans

Computer Science Principles

Recommended for Grades 9-12

Computer Science Principles is a course designed to prepare students (and teachers) who are new to computer science for the AP CS Principles exam. The course covers many topics including the Internet, Big Data and Privacy, and Programming and Algorithms.

View course | Lesson plans

Courses

Selecting the right course for your class

Kindergarten	1st Grade	2nd Grade	3rd Grade	4th Grade	5th Grade
Course A	Course B	Course C	Course D	Course E	Course F
Pre-Reader Express Course		Express Course			

Computer Science Discoveries

Recommended for Grades 6-10

Computer Science Discoveries (CS Discoveries) is an introductory computer science course that empowers students to create authentic artifacts and engage with computer science as a medium for creativity, communication, problem solving, and fun.

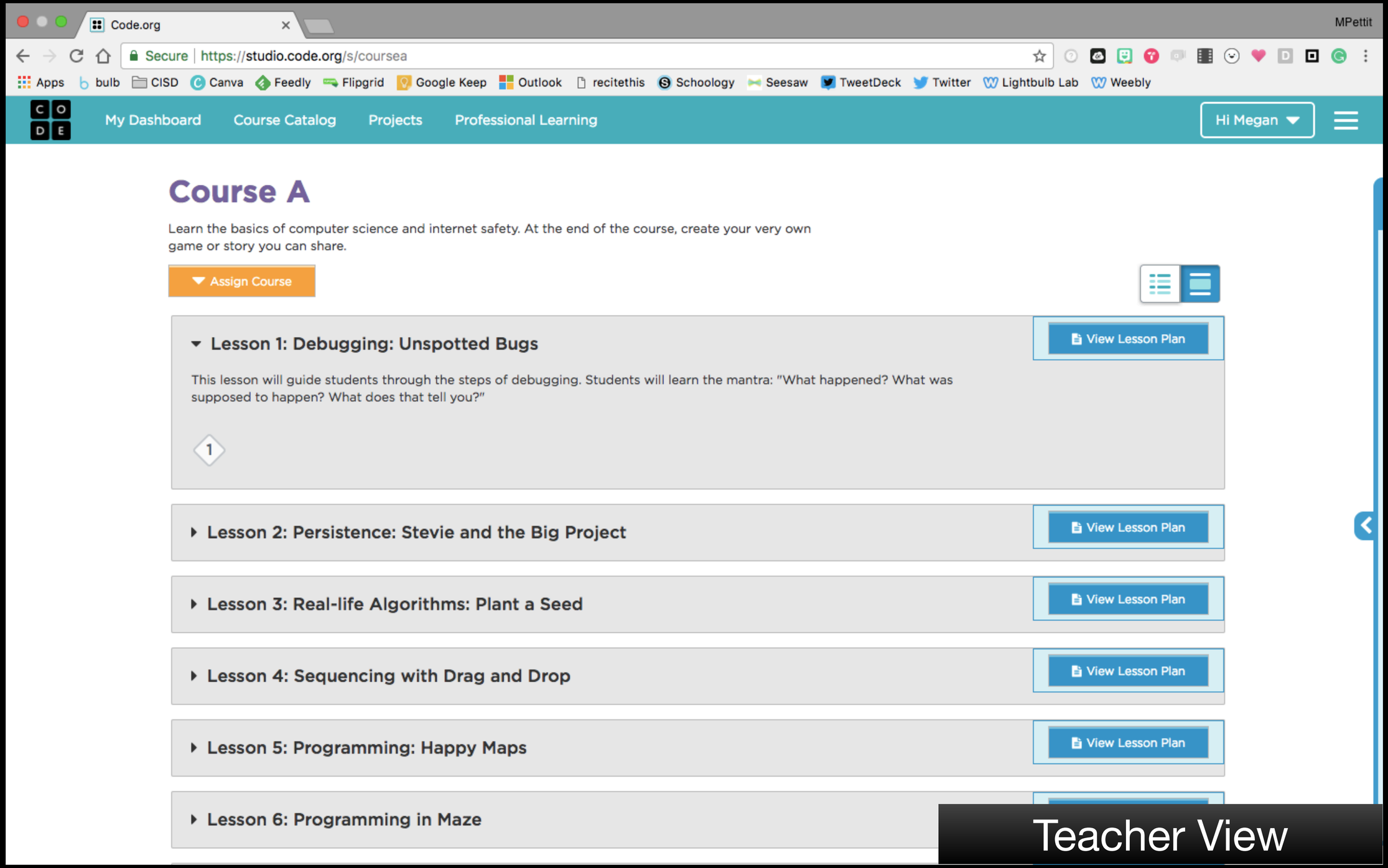
View course | Lesson plans

Computer Science Principles

Recommended for Grades 9-12

Computer Science Principles is a course designed to prepare students (and teachers) who are new to computer science for the AP CS Principles exam. The course covers many topics including the Internet, Big Data and Privacy, and Programming and Algorithms.

View course | Lesson plans


Code.org
MPettit
Secure | https://studio.code.org/s/coursea
Apps
bulb
CISD
Canva
Feedly
Flipgrid
Google Keep
Outlook
recitethis
Schoology
Seesaw
TweetDeck
Twitter
Lightbulb Lab
Weebly
My Dashboard
Course Catalog
Projects
Professional Learning
Hi Megan
Course A
Learn the basics of computer science and internet safety. At the end of the course, create your very own game or story you can share.
Assign Course
Lesson 1: Debugging: Unspotted Bugs
View Lesson Plan
This lesson will guide students through the steps of debugging. Students will learn the mantra: "What happened? What was supposed to happen? What does that tell you?"
1
Lesson 2: Persistence: Stevie and the Big Project
View Lesson Plan
Lesson 3: Real-life Algorithms: Plant a Seed
View Lesson Plan
Lesson 4: Sequencing with Drag and Drop
View Lesson Plan
Lesson 5: Programming: Happy Maps
View Lesson Plan
Lesson 6: Programming in Maze
Teacher View

Code.org

MPettit

Secure | https://studio.code.org/s/coursea

Apps | bulb | CISD | Canva | Feedly | Flipgrid | Google Keep | Outlook | recitethis | Schoology | Seesaw | TweetDeck | Twitter | Lightbulb Lab | Weebly

My Dashboard | Course Catalog | Projects | Professional Learning | Hi Megan

Course A

Learn the basics of computer science and internet safety. At the end of the course, create your very own game or story you can share.

Assign Course

Teacher Panel

View page as: Student | Teacher

Lesson 1: Debugging: Unspotted Bugs

View Lesson Plan

Lesson 2: Persistence: Stevie and the Big Project

View Lesson Plan

When students run into a barrier while answering a question or working on a project, it's so easy for them to get frustrated and give up. This lesson will introduce students to the idea that frustration can be an important part of learning. Here, frustration is presented as a step in the creative process, rather than a sign of failure.This lession can be done over one or two class sessions. If you have more time, feel free to draw out the building and revising phase of the Marble Run activity.

1

Lesson 3: Real-life Algorithms: Plant a Seed

View Lesson Plan

Lesson 4: Sequencing with Drag and Drop

View Lesson Plan

Lesson 5: Programming: Happy Maps

View Lesson Plan

Lesson 6: Programming in Maze

Teacher View

Lesson 1: Debugging: Unspotted Bugs

Bug | Debugging | Persistence | Unplugged

Overview

This lesson will guide students through the steps of debugging. Students will learn the mantra: "What happened? What was supposed to happen? What does that tell you?"

Purpose

Research shows that some students have less trouble debugging a program than writing one when they first learn to code. In this lesson, we introduce the idea of debugging in a real world sense.

The goal in this lesson is to teach students steps to spot a bug and to increase persistence by showing them that it's normal to find mistakes. In later lessons, students will debug actual programs on Code.org.

Agenda

- **Warm Up (12 min)**
  - **Unspotted Bugs**
  - **Vocabulary**
- **Marble Run Breakdown (10 - 20 min)**
  - **Debug the Run**
- **Wrap Up (10 - 20 min)**
  - **Journaling**
- **Extended Learning**
  - **Real Life Bug Hunting**

Objectives

Students will be able to:

- Express that they have noticed when something goes differently than what is expected.
- Identify what the expected result was before an error occurs.
- Determine and describe the difference between what was expected and what actually happened in the event of an error.

Preparation

- [ ] Review the Unspotted Bugs Story (**Unspotted Bugs - Online Story**)
- [ ] Pre-read Unspotted Bugs to identify appropriate questions for your classroom
- [ ] Follow instructions in the **Marble Run - Teacher Prep Guide** to make a Marble Run (which will be arranged incorrectly at the start)
- [ ] Give a **Think Spot Journal - Reflection Journal** to each student

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Marble Run** - Teacher
- **Unspotted Bugs** - Sto
- **Think Spot Journal** -

Teacher View

It can be distracting to a learner when they have unanswered questions or doubts. To end this lesson, we'll give everyone the chance to get those out so that they can reflect on what they've been taught.

Encourage students to share their thoughts and questions either with the whole class or with an elbow partner.

Reflect:

Once they've had time to ponder their own thoughts, get the students thinking about the purpose of the lesson that they just learned. Why did you do this activity? How will it help them later? Can they think of buggy things that they've seen in the real world?

Students should finish by drawing or writing in their journal. Possible topics include:

- How do you feel when something that you are working on acts buggy?
- How many times do you think you should try to fix a bug before you give up?
- What would you do if you notice that something is buggy, but you don't know how to fix it?

Lesson Tip

Say:

What do you think we learned in this lesson?

- Debugging
- How to solve a problem
- How to make a marble go
- How do you think that can help us in other places?

Extended Learning

Real Life Bug Hunting

Take your students outside. Do you see any signs of bugs? What are they? Now look closer... can you find the actual bug?

Lesson Tip:

The signs of real-live bugs won't be as dramatic as upside down trees, but it might be dead leaves, spots on flowers, or slime on the sidewalk. Have the students brainstorm these before going outside to look for them.

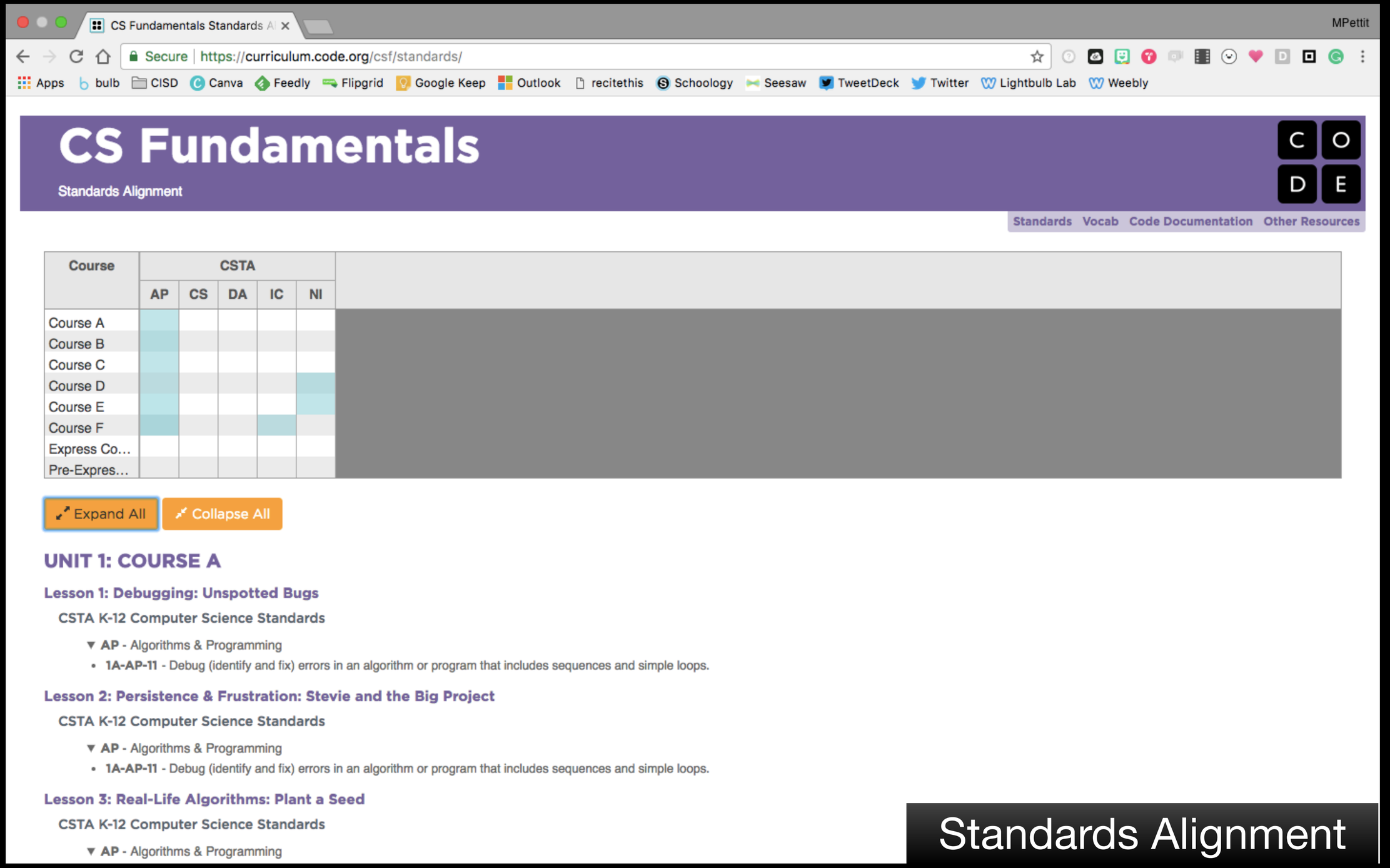
Standards Alignment

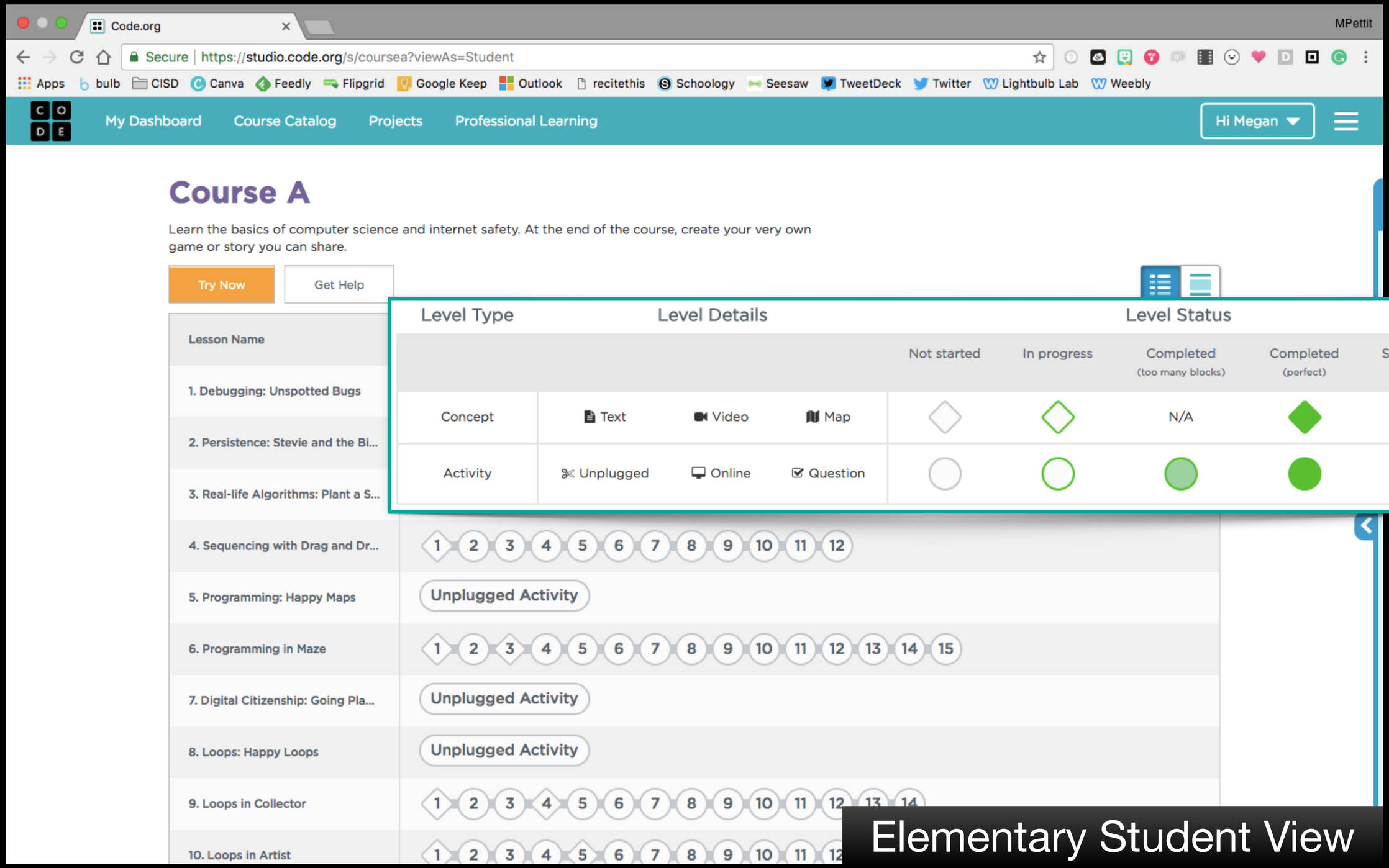
View full course alignment

CSTA K-12 Computer Science Standards

▶ **AP** - Algorithms & Programming

Lesson Standards Alignment


CS Fundamentals Standards A
MPettit
Secure | https://curriculum.code.org/csf/standards/
Apps bulb CISD Canva Feedly Flipgrid Google Keep Outlook recitethis Schoology Seesaw TweetDeck Twitter Lightbulb Lab Weebly
CS Fundamentals
Standards Alignment
C O D E
Standards Vocab Code Documentation Other Resources
Course
CSTA
AP CS DA IC NI
Course A
Course B
Course C
Course D
Course E
Course F
Express Co…
Pre-Expres…
Expand All
Collapse All
UNIT 1: COURSE A
Lesson 1: Debugging: Unspotted Bugs
CSTA K-12 Computer Science Standards
▼ AP - Algorithms & Programming
• 1A-AP-11 - Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.
Lesson 2: Persistence & Frustration: Stevie and the Big Project
CSTA K-12 Computer Science Standards
▼ AP - Algorithms & Programming
• 1A-AP-11 - Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.
Lesson 3: Real-Life Algorithms: Plant a Seed
CSTA K-12 Computer Science Standards
▼ AP - Algorithms & Programming
Standards Alignment


Code.org
MPettit
Secure | https://studio.code.org/s/coursea?viewAs=Student
Apps bulb CISD Canva Feedly Flipgrid Google Keep Outlook recitethis Schoology Seesaw TweetDeck Twitter Lightbulb Lab Weebly
My Dashboard
Course Catalog
Projects
Professional Learning
Hi Megan
Course A
Learn the basics of computer science and internet safety. At the end of the course, create your very own game or story you can share.
Try Now
Get Help
Lesson Name
1. Debugging: Unspotted Bugs
2. Persistence: Stevie and the Bi...
3. Real-life Algorithms: Plant a S...
4. Sequencing with Drag and Dr...
5. Programming: Happy Maps
6. Programming in Maze
7. Digital Citizenship: Going Pla...
8. Loops: Happy Loops
9. Loops in Collector
10. Loops in Artist
Unplugged Activity
Unplugged Activity
Unplugged Activity
Level Type
Level Details
Level Status
Not started
In progress
Completed (too many blocks)
Completed (perfect)
Submitted
Concept
Text
Video
Map
N/A
N/A
Activity
Unplugged
Online
Question
Elementary Student View

Courses

Selecting the right course for your class

Kindergarten	1st Grade	2nd Grade	3rd Grade	4th Grade	5th Grade
Course A	Course B	Course C	Course D	Course E	Course F
Pre-Reader Express Course		Express Course			

Computer Science Discoveries

Recommended for Grades 6-10

Computer Science Discoveries (CS Discoveries) is an introductory computer science course that empowers students to create authentic artifacts and engage with computer science as a medium for creativity, communication, problem solving, and fun.

View course | Lesson plans

Computer Science Principles

Recommended for Grades 9-12

Computer Science Principles is a course designed to prepare students (and teachers) who are new to computer science for the AP CS Principles exam. The course covers many topics including the Internet, Big Data and Privacy, and Programming and Algorithms.

View course | Lesson plans

Courses

Selecting the right course for your class

Kindergarten	1st Grade	2nd Grade	3rd Grade	4th Grade	5th Grade
Course A	Course B	Course C	Course D	Course E	Course F
Pre-Reader Express Course		Express Course			

Computer Science Discoveries

Recommended for Grades 6-10

Computer Science Discoveries (CS Discoveries) is an introductory computer science course that empowers students to create authentic artifacts and engage with computer science as a medium for creativity, communication, problem solving, and fun.

View course | Lesson plans

Computer Science Principles

Recommended for Grades 9-12

Computer Science Principles is a course designed to prepare students (and teachers) who are new to computer science for the AP CS Principles exam. The course covers many topics including the Internet, Big Data and Privacy, and Programming and Algorithms.

View course | Lesson plans

Code.org

MPettit

Secure | https://studio.code.org/s/csd1?viewAs=Teacher

Apps bulb CISD Canva Feedly Flipgrid Google Keep Outlook recitethis Schoology Seesaw TweetDeck Twitter Lightbulb Lab Weebly

CSD Unit 1 - Problem Solving

Unit 1 is a highly interactive and collaborative introduction to the field of computer science, as framed within the broader pursuit of solving problems. You'll practice using a problem solving process to address a series of puzzles, challenges, and real world scenarios. Next, you'll learn how computers input, output, store, and process information to help humans solve problems. The unit concludes with a project in which you design an application that helps solve a problem of your choosing.

Assign Unit | Teacher resources

Survey

Chapter 1: The Problem Solving Process

- Lesson 1: Intro to Problem Solving — View Lesson Plan
- Lesson 2: The Problem Solving Process — View Lesson Plan
- Lesson 3: Exploring Problem Solving — View Lesson Plan

Chapter 2: Computers and Problem Solving

- Lesson 4: What is a Computer?

Secondary Teacher View

Code.org

MPettit

Secure | https://studio.code.org/s/csd1?viewAs=Student

Apps | bulb | CISD | Canva | Feedly | Flipgrid | Google Keep | Outlook | recitethis | Schoology | Seesaw | TweetDeck | Twitter | Lightbulb Lab | Weebly

My Dashboard | Course Catalog | Projects | Professional Learning | Hi Megan

View all units

CSD Unit 1 - Problem Solving

Unit 1 is a highly interactive and collaborative introduction to the field of computer science, as framed within the broader pursuit of solving problems. You'll practice using a problem solving process to address a series of puzzles, challenges, and real world scenarios. Next, you'll learn how computers input, output, store, and process information to help humans solve problems. The unit concludes with a project in which you design an application that helps solve a problem of your choosing.

Continue | Get Help

Survey

Chapter 1: The Problem Solving Process

- Lesson 1: Intro to Problem Solving
- Lesson 2: The Problem Solving Process
- Lesson 3: Exploring Problem Solving

Chapter 2: Computers and Problem Solving

Secondary Student View

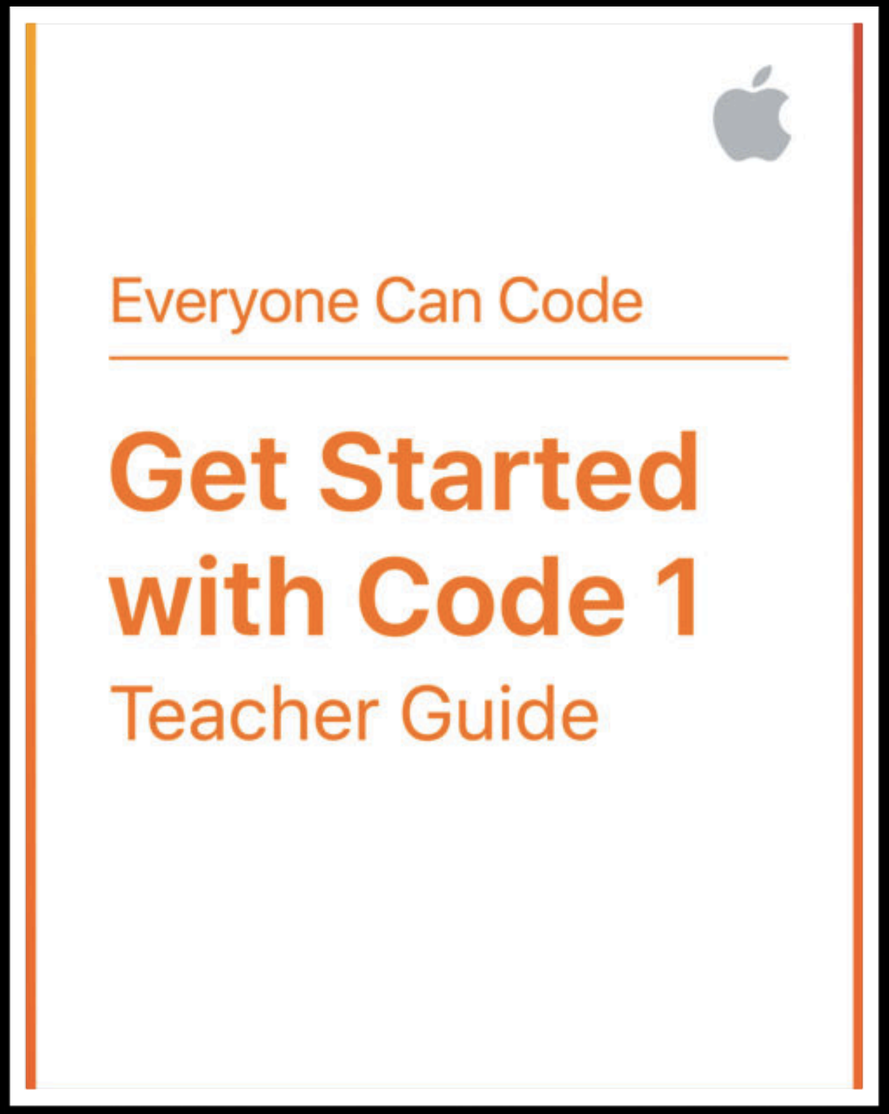
Everyone Can Code

apple.com/education

Teaching Code

Everyone Can Code

- Get Started with Code 1 (K-2nd)
- Get Started with Code 2 (3rd-5th)
- Learn to Code 1 & 2 (6th-8th)
- Learn to Code 3 (6th-8th)
- Intro to App Development with Swift (9th-12th)
- App Development with Swift (9th-12th)



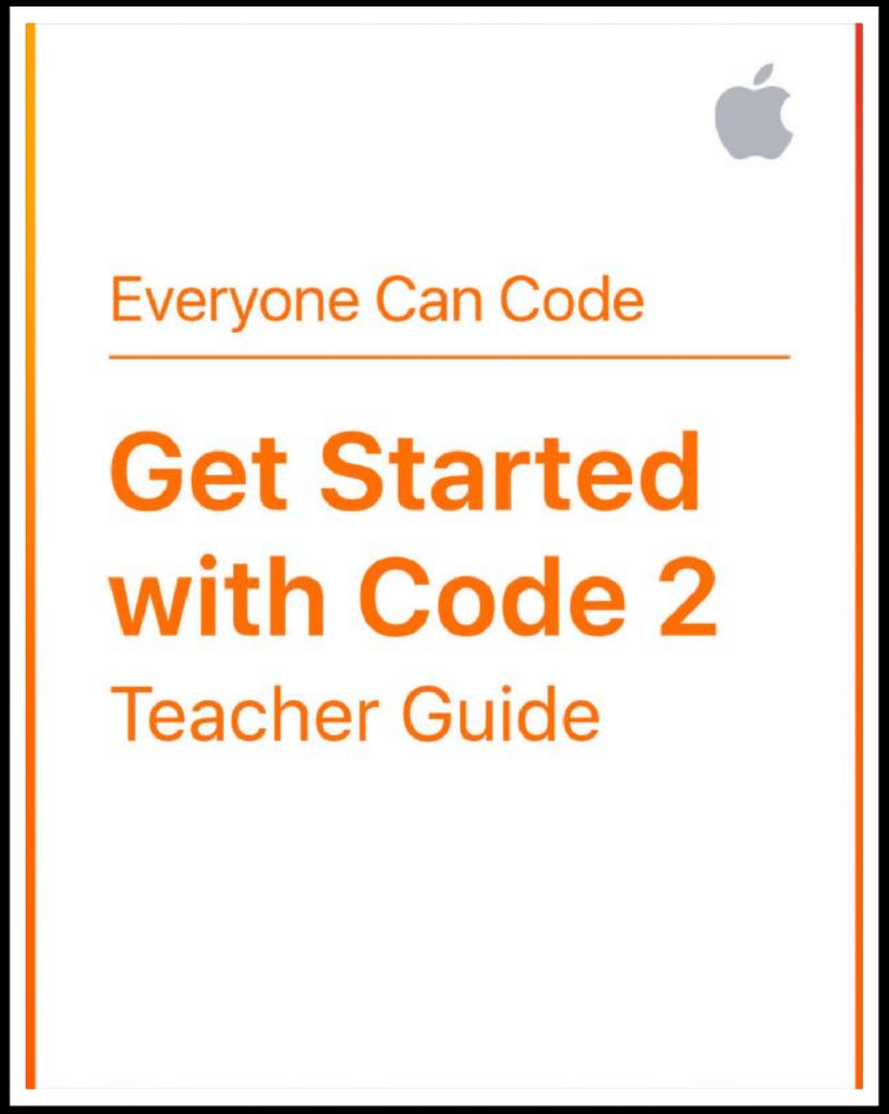
Grade Level: K-2

Device: iPad

<u>Apps</u>

- codeSpark (The Foos)
- Tynker
  - Space Cadet

Hours: 30

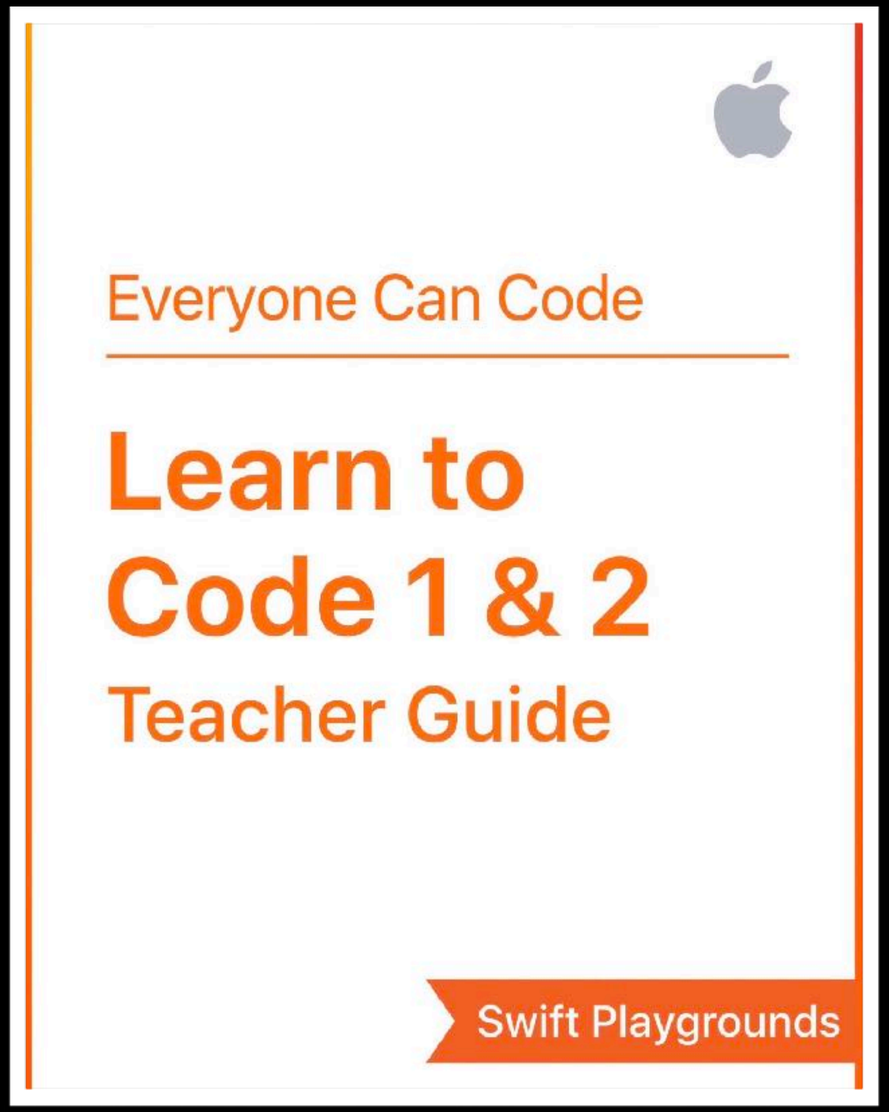


Grade Level: 3-5

Device: iPad

Apps: Tynker (Dragon Spells)

Hours: 36

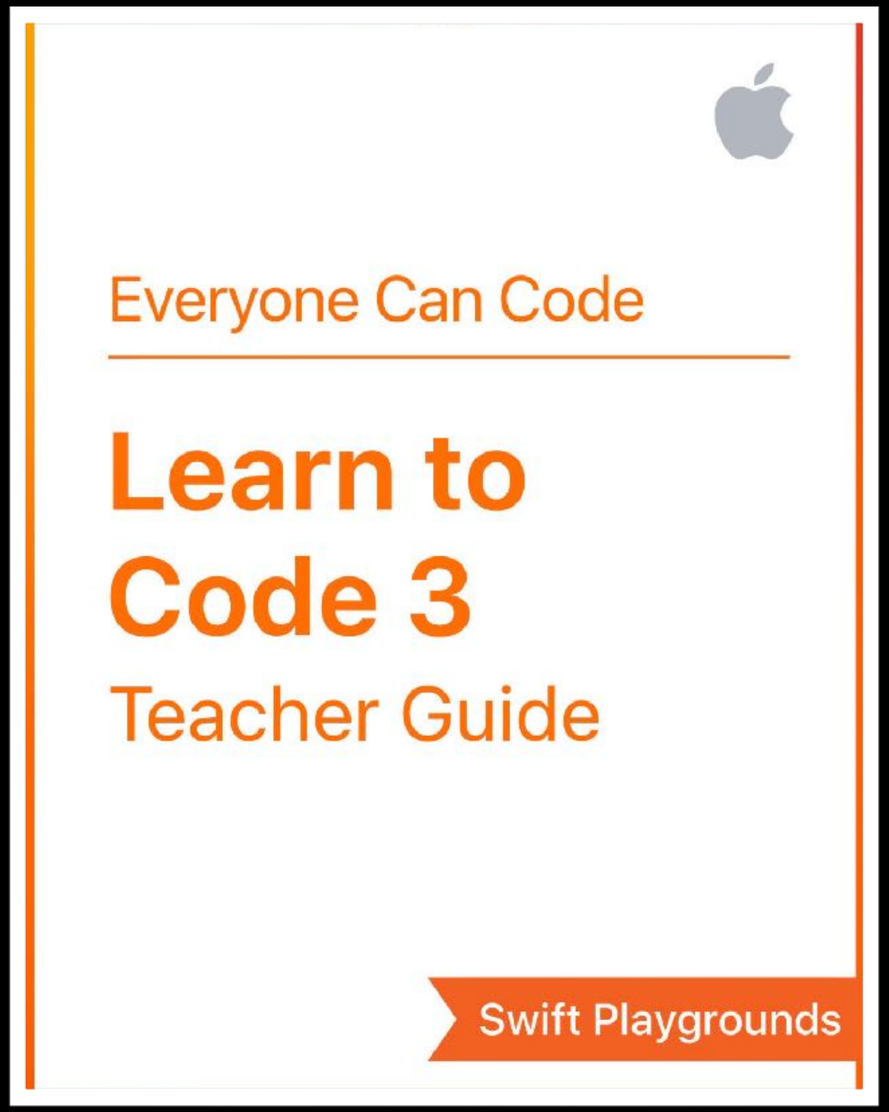


Grade Level: 6-8

Device: iPad

Apps: Swift Playgrounds

Hours: 85

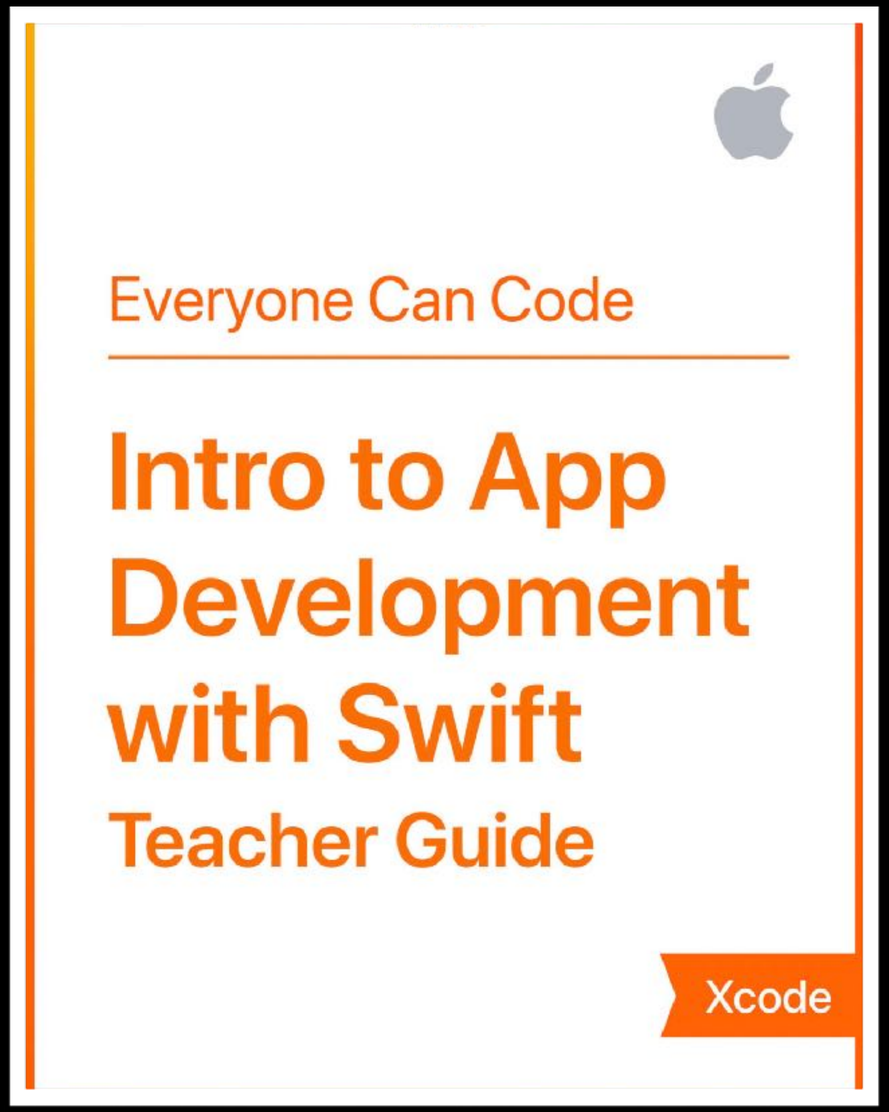


Grade Level: 6-8

Device: iPad

Apps: Swift Playgrounds

Hours: 45

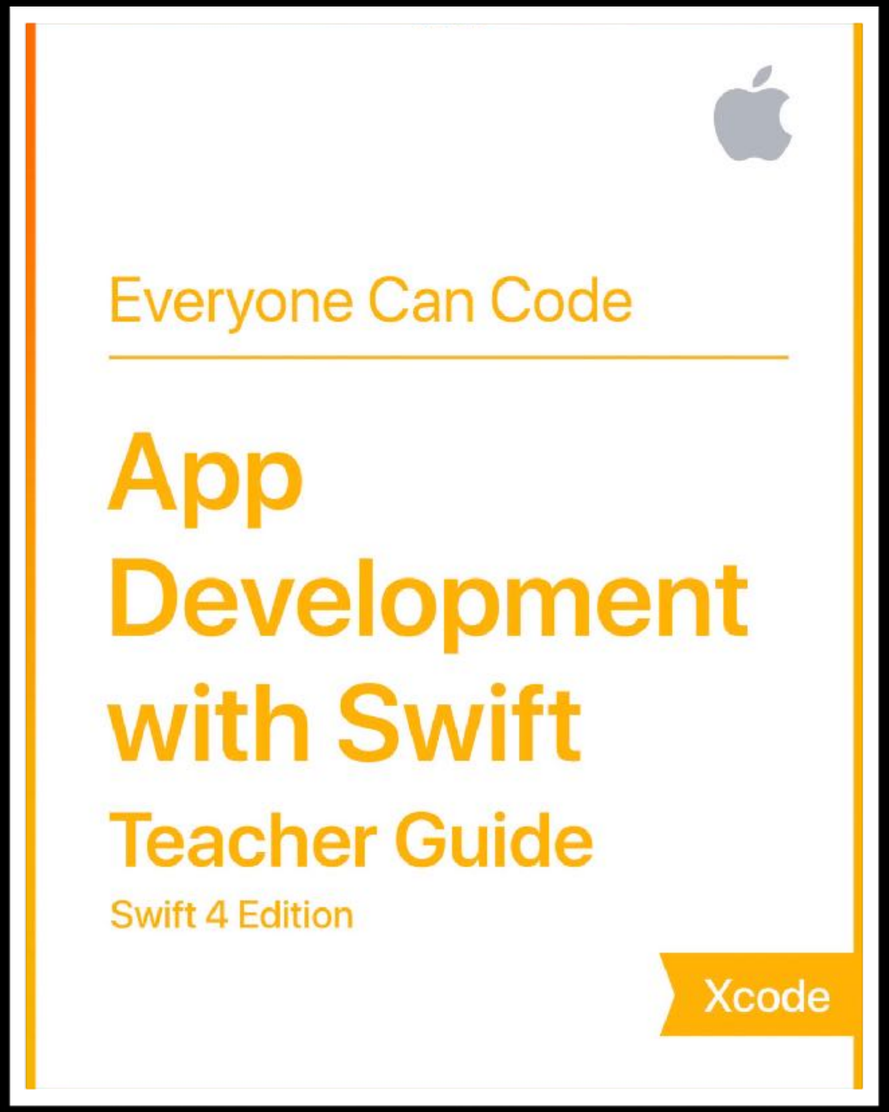


Grade Level: 9-12

Device: Mac

Apps: Xcode

Hours: 90



Grade Level: 9-12

Device: Mac

Apps: Xcode

Hours: 180

About the lessons

INTRODUCTION

In this section, you'll define the coding concept and spend a few minutes as a class discussing an everyday situation that's related to the coding concept. The goal is for students to see that the concept is not used just in coding environments; it's a way of thinking that they can apply to other topics and situations.

ACTIVITY

Students now complete an activity that expands on the discussion and lets them explore and apply the concept. These activities are designed to deepen understanding of the coding concept and to foster communication and teamwork. The activities also leverage iPad to capture student work, which can be used for formative assessment.

PRACTICE

At this point in the lesson, students will apply the coding concept and actually code in Swift Playgrounds. In the app, students are guided through puzzles as they learn to code. The list of coding skills students will develop from completing the puzzles is included to give you insight into their experiences with the app. Answer keys are also provided so you can give additional support if needed.

REFLECTION

You'll find a few suggested questions and prompts for class discussion in this section. They're designed for review, to reinforce the connection between applying the concept inside and outside a coding environment, and to broaden student understanding about what computer scientists do.

JOURNAL

A key requirement of this Swift Playgrounds course is that students create a portfolio of their work using an app like Seesaw. For each lesson, ask students to upload their work from the activity along with photos or videos of their coding from the app. Seesaw includes features like voice recording and mark-up tools for students to reflect on their work. This allows you to track student progress and use their portfolios for assessment.

Download Seesaw from the App Store

You can use the rubric provided here to assess student work. The rubric is embedded in a gradebook to help you easily track progress.

Tap to view the rubric.

Tap to download the gradebook.

About the interactives

WHY

Better understand the goal of a discussion or an activity.

EXAMPLES

Here you'll get examples you could use to foster your discussion or to help students understand what they should be doing.

TIPS

Get tips that could help you facilitate or enhance the lesson.

DID YOU KNOW

For your curious students, get additional facts and tips about coding.

STUDENT WORK EXAMPLES

Find examples of completed student work so you can see what the activities should look like.

KEYNOTE SLIDES

You'll find simple Keynote slides you can use during class to guide your students through the lessons. Add any of the Why, Examples, Tips, and Did You Know information to the speaker notes to help guide you through the lesson. Use the Keynote presentations as is, as templates you
can customize, or simply as inspiration for how to make the lesson your own. Tap the Keynote icon to preview the slides in the book. Tap the arrow icon to download them to your iPad.

CODING SKILLS

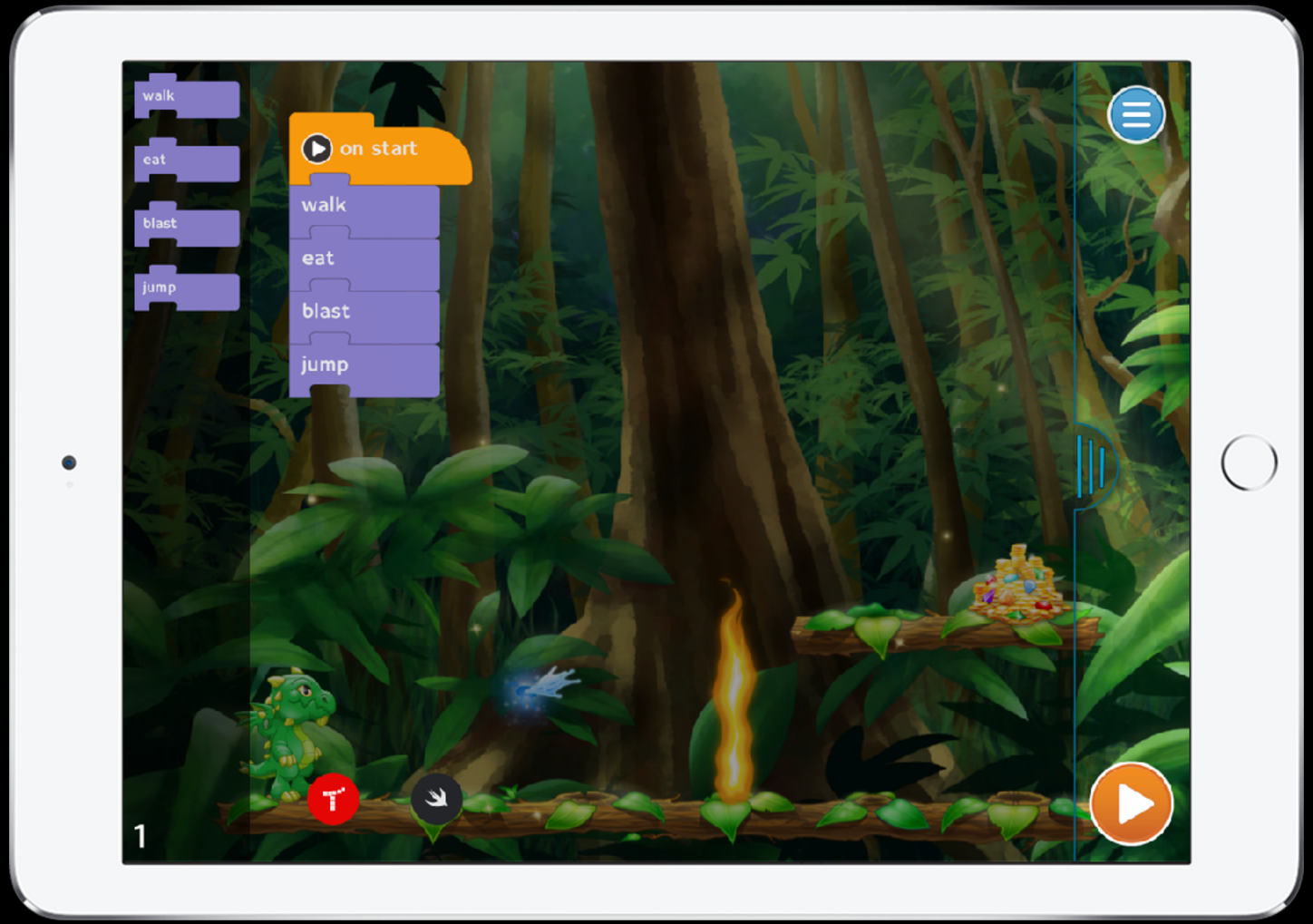
Better understand what skills your students will gain from their experiences in Swift Playgrounds.

ANSWER KEY

Get easy access to the answer keys and use them to provide support for students who are stuck at specific points in the Swift Playgrounds puzzles. A few puzzles are open-ended and have no solution, and many puzzles have multiple solutions.

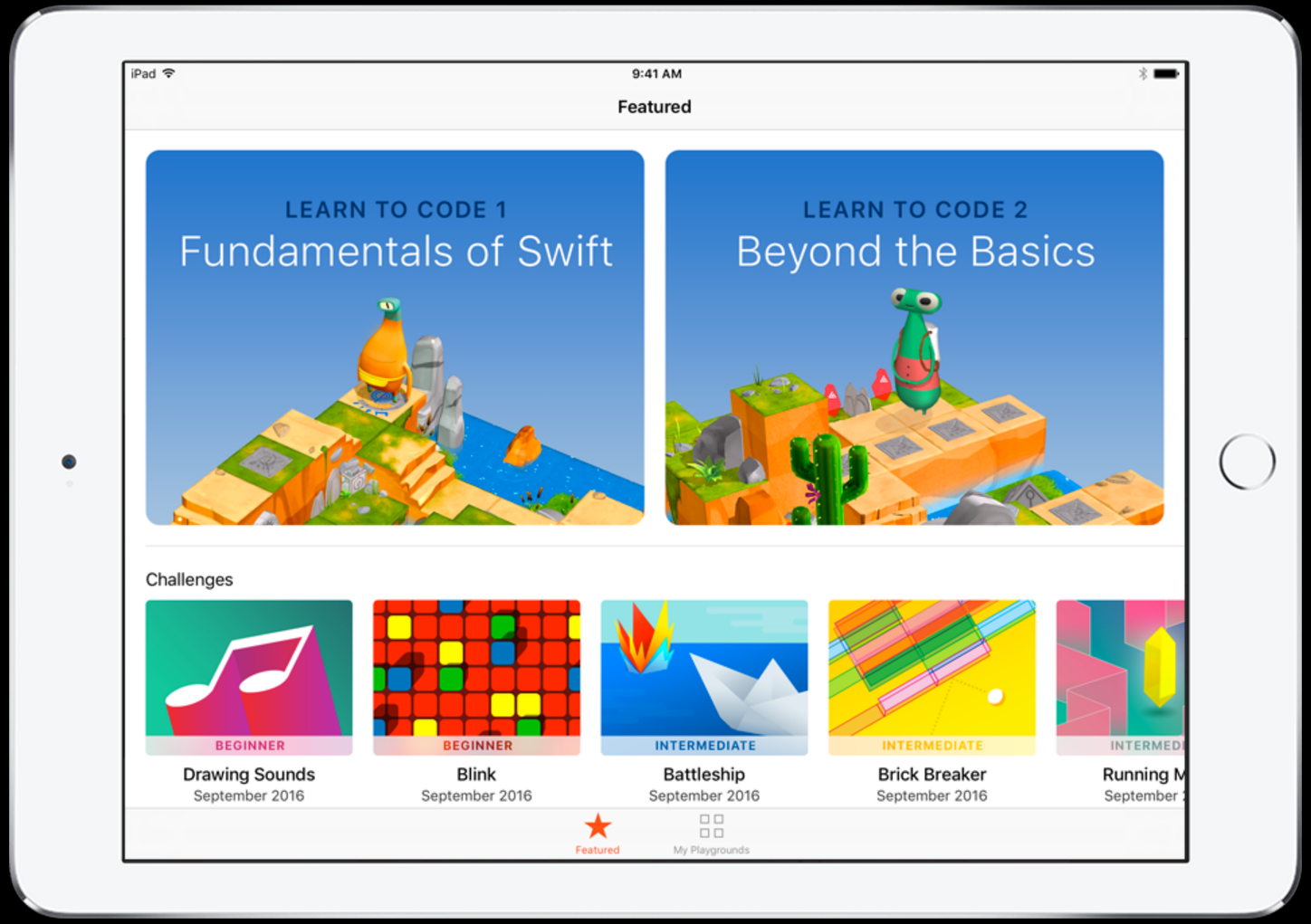
codeSpark

Tynker

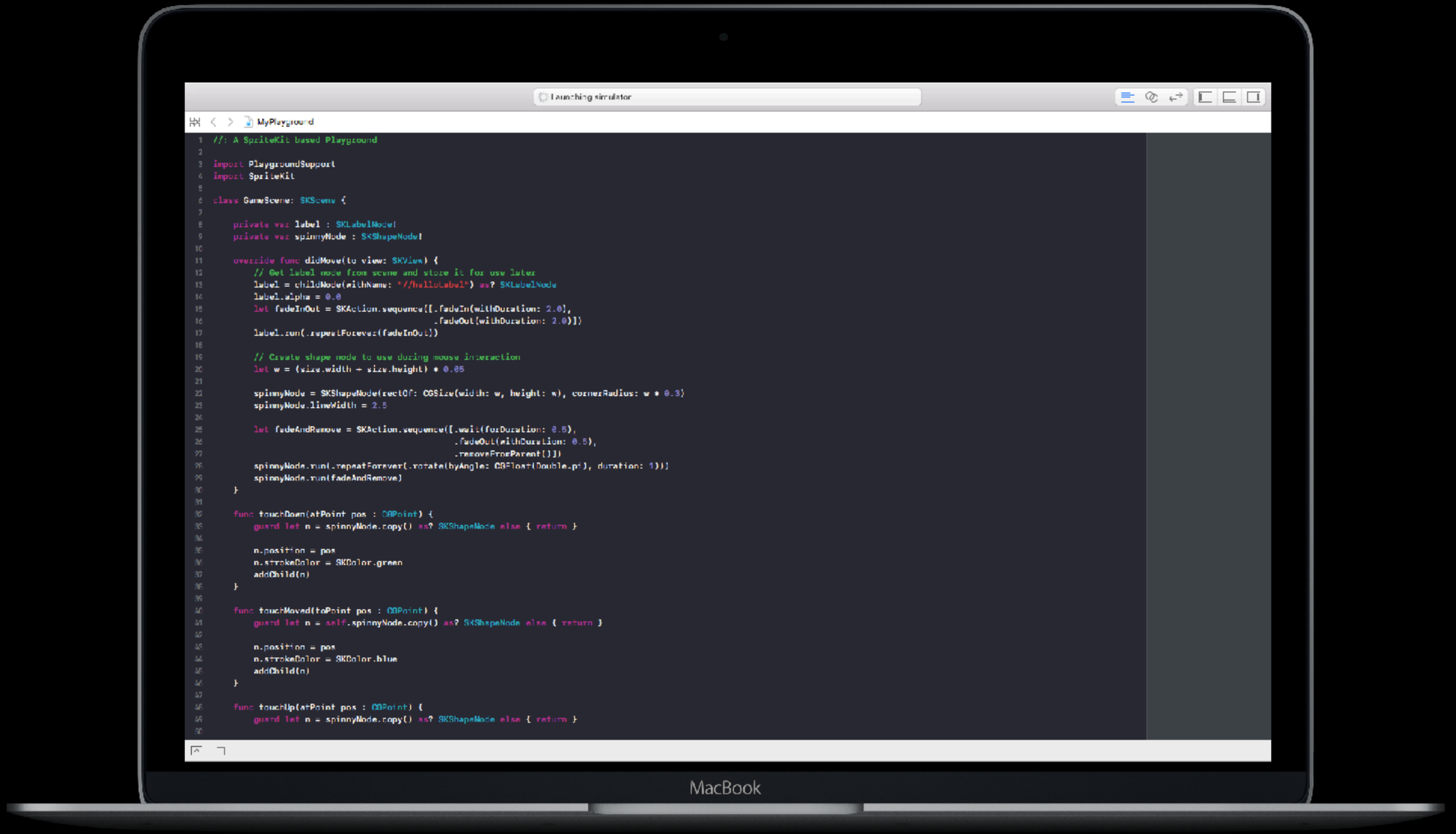


Tynker

Swift Playgrounds



Xcode


Launching simulator
MyPlayground
//: A SpriteKit based Playground
import PlaygroundSupport
import SpriteKit
class GameScene: SKScene {
private var label : SKLabelNode!
private var spinnyNode : SKShapeNode!
override func didMove(to view: SKView) {
// Get label node from scene and store it for use later
label = childNode(withName: "//helloLabel") as? SKLabelNode
label.alpha = 0.0
let fadeInOut = SKAction.sequence([.fadeIn(withDuration: 2.0),
.fadeOut(withDuration: 2.0)])
label.run(.repeatForever(fadeInOut))
// Create shape node to use during mouse interaction
let w = (size.width + size.height) * 0.05
spinnyNode = SKShapeNode(rectOf: CGSize(width: w, height: w), cornerRadius: w * 0.3)
spinnyNode.lineWidth = 2.5
let fadeAndRemove = SKAction.sequence([.wait(forDuration: 0.5),
.fadeOut(withDuration: 0.5),
.removeFromParent()])
spinnyNode.run(.repeatForever(.rotate(byAngle: CGFloat(Double.pi), duration: 1)))
spinnyNode.run(fadeAndRemove)
}
func touchDown(atPoint pos : CGPoint) {
guard let n = spinnyNode.copy() as? SKShapeNode else { return }
n.position = pos
n.strokeColor = SKColor.green
addChild(n)
}
func touchMoved(toPoint pos : CGPoint) {
guard let n = self.spinnyNode.copy() as? SKShapeNode else { return }
n.position = pos
n.strokeColor = SKColor.blue
addChild(n)
}
func touchUp(atPoint pos : CGPoint) {
guard let n = spinnyNode.copy() as? SKShapeNode else { return }
MacBook

Curriculum Exploration


Reflection

Coppell DLC Twitter: #coppelldlc
coppellisd.com/digitallearning
#TCEA18

Code2Learn:
From Playgrounds to Curriculum

February 5, 2018

Essential Question: How do we increase our capacity to teach and learn to code?

Self Assessment

In my classroom/campus/district, the coding exposure level is

|Broad & Deep Exposure | | | | | Moderate Exposure | | | | |Basic Exposure|

Reflection Point #1: Open Play	Reflection Point #2: Exploring Curriculum	Reflection Point #3: Guided Play

Guided Play

Guided Play

Online Options	Coding Tools	Apps
code.org	Sphero	Tynker
Scratch (web version)	Dash	ScratchJr
Khan Academy	Pro-Bot	Swift Playgrounds (iPad Only)
Tynker website	Osmo Coding	codeSpark Academy
	Blue-Bot	

These are just a few of the possible resources you can explore during Guided Play.
Feel free to explore other coding resources you find!

Code.org

Challenge

Find an applicable course from the course catalog below.

studio.code.org/courses

Going Further

Once you've found an applicable course, click "Try Now" to walk through the materials and activities.

Back to Menu

Scratch Website

Challenge

Visit the Scratch Website, scratch.mit.edu. Click on “Tips” at the top of the screen, and choose a tutorial to explore.

Going Further

On the Scratch website, click on “Explore” at the top of the window, find a project, and click “See Inside.” Click on “Remix” near the upper right corner to add your own ideas to the project.

To find out more about utilizing Scratch in your classroom, visit scratch.mit.edu/educators/ and scratched.gse.harvard.edu/resources/all

Back to Menu

Khan Academy

Challenge

Explore the programming part of Khan Academy. Create an account in khanacademy.com with your Google account, then explore the Intro to JS Drawing Basics module: tinyurl.com/dlcdrawkhan

Going Further

Explore other computer science & programming resources at Khan Academy khanacademy.org/computing

Back to Menu

Tynker.com

Challenge

Open the Tynker website tynker.com. Click on “PLAY” at the top of the window. Select “Coding Games” to find activities to learn.

Going Further

Create a Tynker teacher account at tynker.com. Explore the free 6-week programming course with lesson ideas to use in your classroom. tinyurl.com/dlctynker

Back to Menu

Sphero

Back to Menu

Dash

Download the Wonder App for Dash & Dot Here
goo.gl/sjPkP1

Challenge

Complete a few Scroll Quest Challenges with Dash or Dot to see what treasures you can “unlock”. If the Challenge has been completed, you may opt to “replay” it.

Going Further

Want to further explore the adventures of Dash and Dot? Great! Visit play.makewonder.com

Back to Menu

Pro-Bot

Challenge

Use ProBot's keypad to work on directional language and programming through sequences of forwards, backwards, left and right 90 degree turns.

Going Further

Using the manual or experimenting try drawing a square, triangle or parallelogram.

Back to Menu

Osmo Coding

Download the Osmo here
tinyurl.com/jylytko

Challenge

Help Awbie make his way through the forest with the coding blocks. You may choose to follow the tutorial or design your own adventure as you explore using the different blocks. You may replay the different levels.

Going Further

Continue to explore by working through more challenging levels of Osmo Coding using Awbie.

Back to Menu

Blue-Bot

Download BlueBot app here
tinyurl.com/appbluebot

Challenge

Use Blue-Bot's keypad to work on directional language and programming through sequences of forwards, backwards, left and right 90 degree turns.

Work your way through the card Mat. Start your journey at a specified location on the mat, and try to program Blue-bot to a specific destination on the mat.

Going Further

Download the Blue-Bot app, and pair the Blue-Bot to the iPad. Program Blue-Bot to move between specified areas of the mat using the Blue-Bot app on the iPad.

Back to Menu

Tynker

Download the Tynker App
goo.gl/rFYTvV

Challenge

Open the Tynker App and use block coding to play the “Space Cadet game.

Going Further

Go to the “Workshop” in the Tynker app. Select one of the “Beginner Projects” or use the “Blank Template” option.

Create a Tynker teacher account at tynker.com. Explore the free 6-week programming course with lesson ideas to use in your classroom.
tinyurl.com/dlctynker

Back to Menu

ScratchJr

Download the Scratch Jr. App Here
goo.gl/9D0iDw

Challenge

Create your own interactive story or game by creating a new project in “My Projects”. Have fun snapping the blocks together to make the characters move, jump, dance, and sing!

Going Further

To learn more about Scratch, Jr., visit scratchjr.org.

Additional information about utilizing Scratch in your classroom, can be found at scratch.mit.edu/educators and scratched.gse.harvard.edu/resources/all

Back to Menu

Swift Playgrounds

Download Swift Playgrounds here
apple.com/swift/playgrounds

Challenge

Explore the fundamentals of Swift and Byte's world in the Hour of Code challenge. Begin by learning how to write simple commands, functions and run loops to solve puzzles in a 3D world. How far can you get through the Hour of Code challenge?

Going Further

Continue your journey with Byte and learning the fundamentals of Swift through his 3D world. You will begin adding in conditional code to changes in the environment. Also, explore operators and while loops to make your code easier to use, replicate, and use in new worlds.

Back to Menu

codeSpark Academy

goo.gl/tmgKth

Challenge

Work through several levels of The Foos. Can you earn 3 stars for each level consistently by programming your Foo to move through the game? Watch out...the levels get harder as you progress!

Going Further

Learn ways to incorporate the thinking skills practiced in The Foos into the classroom by checking out the resources at tinyurl.com/foosk1.

Back to Menu


Reflection

Code2Learn:
From Playgrounds to Curriculum

February 5, 2018

Essential Question: How do we increase our capacity to teach and learn to code?

Self Assessment

In my classroom/campus/district, the coding exposure level is

|Broad & Deep Exposure | | | | | Moderate Exposure | | | | |Basic Exposure|

Reflection Point #1: Open Play	Reflection Point #2: Exploring Curriculum	Reflection Point #3: Guided Play

Framework

K12CS.ORG

Standards

Tech Apps

CSTA Standards

“I Can” Statements

CSTA CS "I can..." Statements

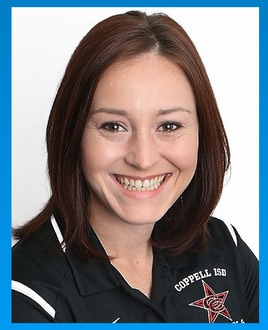
(taken from Progression of Computer Science Teachers Association (CSTA) K-12 Computer Science Standards, Revised 2017)

Concept	Subconcept	Standard	Grade	I Can Statement
	Algorithms	**1A-AP-08** Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks. (P4.4)	1st Grade	We/I can create sets of step-by-step directions (algorithms) to complete a daily task. We/I can complete a task by following a set of step-by-step directions (algorithms).
			2nd Grade	I can create algorithms to complete a daily task. I can complete a task by following algorithms.
	Variables	**1A-AP-09** Model the way programs store and manipulate data by using numbers or other symbols to represent information. (P4.4)	Kinder	We/I can use numbers and symbols to show information the way programs do.
			1st Grade	I can use numbers and symbols to represent information the way programs do.
			2nd Grade	I can use numbers and symbols that represent information the way programs do to collect and edit data.
	Control	**1A-AP-10** Develop programs with sequences and simple loops to express ideas or address a problem. (P5.2)	Kinder	We/I can design programs with steps that go in order (sequences) to share ideas or solve a problem. We/I can design programs with steps that repeat (loops) to share ideas or solve a problem.
			1st Grade	I can design programs with steps that go in order (sequences) to share ideas or solve a problem. I can design programs with steps that repeat (loops) to share ideas or solve a problem.
			2nd Grade	I can design programs with sequences and simple loops to express ideas or solve a problem.


4 Corners

Screen

Your Coding Journey?



Just Started

Implementing without Curriculum

Haven't Started

Implementing with Curriculum

Planning

Code2Learn:
From Playgrounds to Curriculum
February 5, 2018

Planning

Current Reality	What I Wish For	What I Can Control

Closing Shop

How do we increase our capacity to teach and learn how to code?

Brian Timm
btimm@coppellisd.com
@BrianWTimm

Megan Pettit
mpettit@coppellisd.com
@MrsMeganKozar

Survey Link

#TCEA

Trisha Goins
tgoins@coppellisd.com
@heartinlife

Brad Cook
bbcook@coppellisd.com
@rbradcook

Credits

- All images downloaded from pixabay.com
- k12cs.org
- https://www.csteachers.org/general/custom.asp?page=standards
- ritter.tea.state.tx.us/rules/tac/chapter126/index.html