Swift Coding Club

Block-Based Kit

Welcome to the Swift Coding Club!

Learning to code teaches you how to solve problems and work together in creative ways. And it helps you build apps that bring your ideas to life.

Swift Coding Clubs are a fun way to learn to code and design apps. Activities built around Swift, Apple's coding language, help you collaborate as you learn to code, prototype apps, and think about how code can make a difference in the world around you.

You don't have to be a teacher or a coding expert to run a Swift Coding Club. The materials are self-paced, so you can even learn alongside your club members. And you can all celebrate your club's ideas and designs with an app showcase event for your community.

This kit is arranged in three sections:

Get Started

Everything you need to launch a Swift Coding Club.

Learn & Design

Tips and activities for designing club sessions.

Celebrate

Helpful resources to plan and host an app showcase in your community.

Swift Coding Clubs

Block-Based Coding | Ages 8–11

Learn coding basics using visual apps on iPad.

Swift Playgrounds | Ages 11+

Use Swift code to learn coding fundamentals with Swift Playgrounds on iPad.

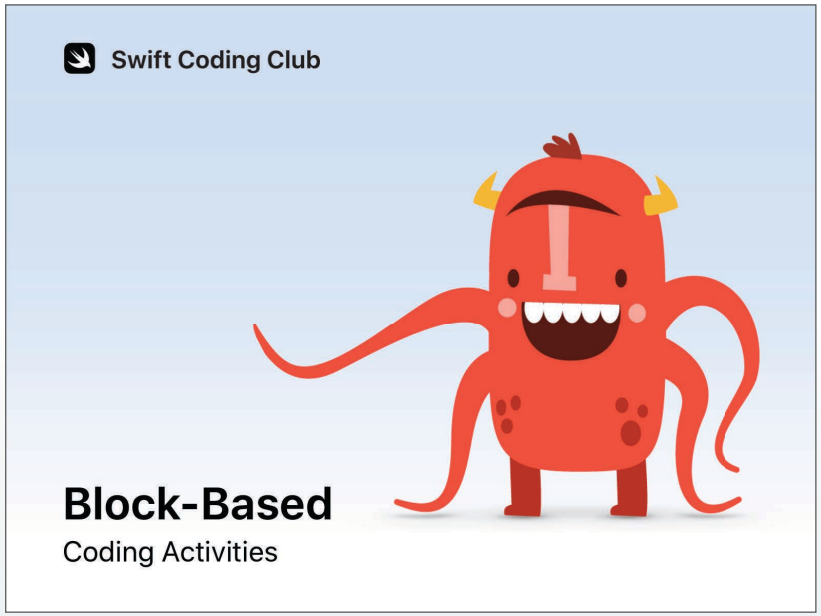
Xcode | Ages 14+

Learn to develop apps in Xcode on Mac.

Get Started

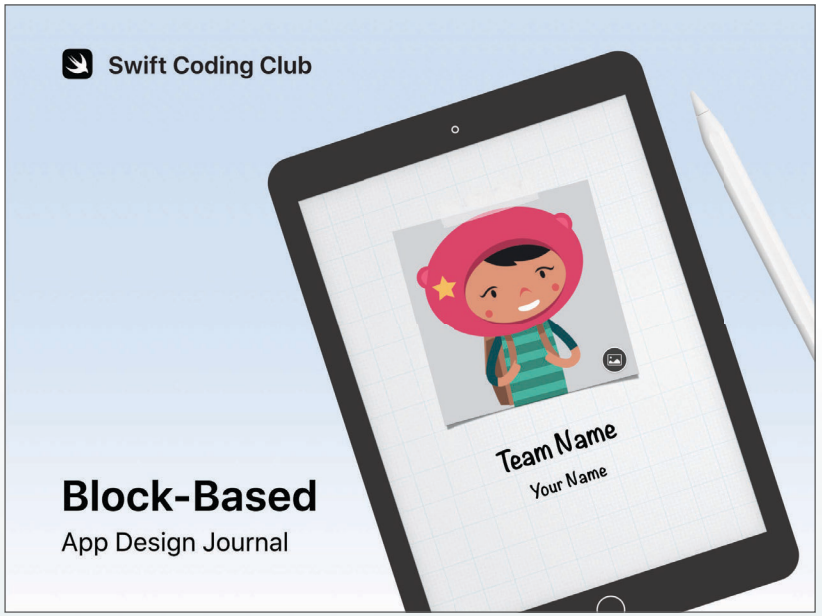
1. Download club materials.

Use AirDrop to share these two guides with club members in your first club meeting. They're also included as part of this document.



Coding Activities
Learn coding concepts with these fun, collaborative activities and use visual blocks to solve puzzles with apps like Tynker and Hopscotch on iPad.

Download Block-Based Coding Activities >



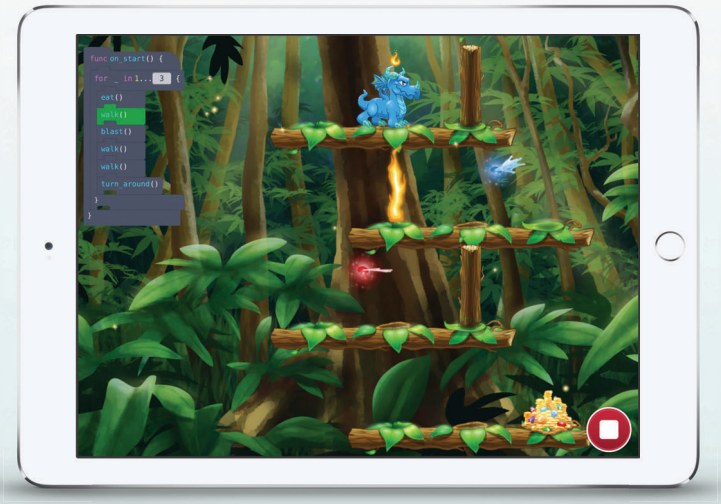
App Design Journal
Explore the app design process with this Keynote journal. Brainstorm, plan, prototype, and evaluate your club's app ideas.

Download Block-Based App Design Journal >

2. Check your tech.

Before your first meeting, be sure you have the following:

- **iPad.** iPad mini 2 or later, iPad Air or later, or iPad Pro running iOS 11 or later. It's best if each person has their own device, but they can also share and code together.
- **A block-based coding app**. We recommend Tynker and Hopscotch. You can use one or both apps in club activities.
- **Keynote.** You'll use the Keynote app on iPad for your app prototypes.
- **Swift Coding Club materials.**



3. Make a plan.

Here are some things to consider:

- Who are your club members? What are their interests? Do they have experience with coding or are they brand-new?
- How often will your club meet? If you're planning a summer camp, how many hours of coding activities will you have?
- What technology is available for the club?
- What are the goals of your club?

4. Spread the word.

Let people know about your Swift Coding Club. Here are some ideas and resources to attract new members to your club:

- **Announce your club.** Use email, social media, the web, flyers, or word of mouth to let your community know about your club.
- **Host an informational meeting.** Ask potential club members about their interests and what types of apps they'd want to create. Talk about ideas for holding an app design showcase and how members can get involved. You can also share a short video about the club online.

These items can help you promote and personalize your Swift Coding Club:

- **Posters.** Download this free template, then personalize it to create your own poster. Print and display it, or make a digital poster to share online. Be sure to include details for when and where the club will meet and how to join.
- **Stickers and T-shirts.** Use these Swift Coding Club stickers to help promote your club. T-shirts are a great way to recognize members who participate in app showcase events. Download the Swift Coding Club T-shirt template to make shirts for your members.

Swift Coding Club poster

Swift Coding Club sticker

Swift Coding Club T-shirt

Learn & Design

The club materials are designed for you to interweave coding and app design activities. You can also add sessions that support your members' interests. Below is a sample schedule for 30 one-hour club sessions.

Sessions 1–5

- Setup
- Coding Activities 0 & 1
- App Design Journal: Exploring Apps

Sessions 6–10

- Coding Activities 2–4
- App Design Journal: My App Idea

Sessions 11–15

- Coding Activities 5–7
- App Design Journal: Building a Prototype

Sessions 16–20

- Coding Activity 8
- App Design Journal: Building a Prototype

Sessions 21–25

- App Design Journal: Evaluate Your App Idea

Sessions 26–30

- App Design Journal: Build Your App Pitch
- App Design Showcase

To expand on app design and coding activities, consider adding sessions like building a drone obstacle course or creating a robot rescue mission challenge. To prompt app design brainstorming, you might even want to add guest speakers or field trips.

Tips for Club Leaders

Build a leadership team. Having a group of members who help with leading the club can make it much easier and more fun. Which club members have leadership potential? Think about adding officers to your club for events, coding, app design, and more.

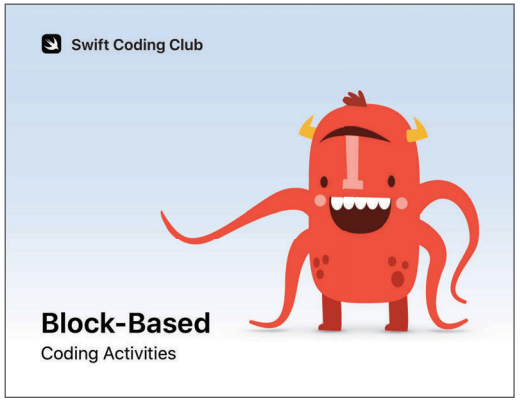
Learn together. Club leaders don't have to know everything. Help your members work on their own research and problem-solving skills and encourage them to help others.

Show off. An app showcase event is a great way to promote your club, app ideas, and coding skills to friends, families, teachers, and the community. It might even help you recruit more members. See page 11 to get tips for holding your own app showcase.

Share ideas. Some members will be interested in making games. Others might want to create apps to help people, tell stories, or control robots. Think about ways for members to work together on projects they care about.

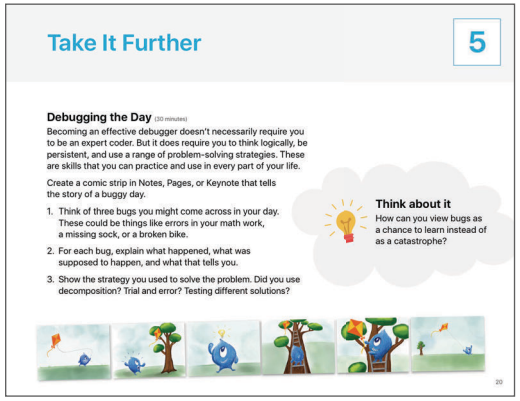
Mix it up. Sometimes members who are more advanced can leave others behind. See if those members can partner up with beginners for pair programming. Teaching someone else is a great way to learn!

Block-Based Coding Activities



Coding activities: Built around block-based coding apps, these collaborative activities introduce fundamental coding concepts and skills.

Coding concepts: In each activity, club members will learn about a fundamental coding concept and explore it in an everyday context. They'll then apply the coding concept to solve puzzles or create projects in a block-based app.



Take It Further: Each coding concept has a Take It Further activity that deepens understanding of the coding concept and fosters communication and teamwork. Members use iPad to apply their understanding in a creative project.

Need more information or want to go deeper?

Coding activities are based on the Get Started with Code curriculum. Learn more >

Learn more about Tynker.

Download the Quick-Start Teacher Guide >

Learn more about Hopscotch.

Download the Quick-Start Teacher Guide >

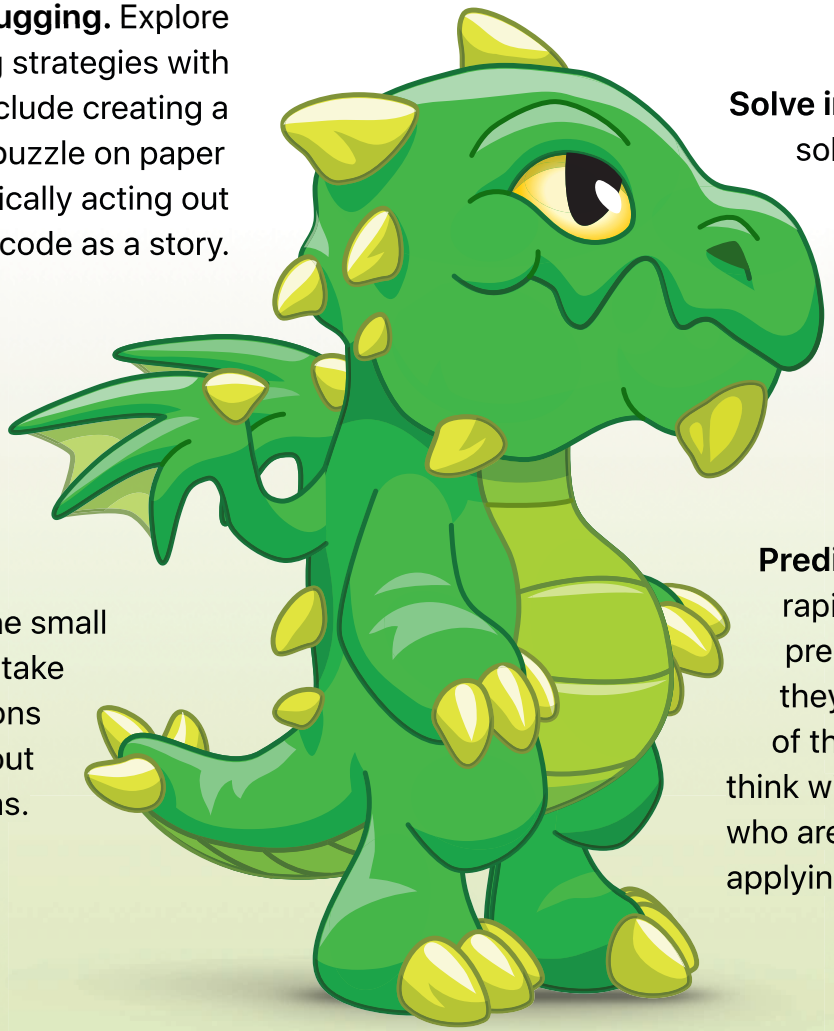
Download the Login Tutorial >

Tips for Learning with Block-Based Apps

Use AirPlay. Show the iPad screen to introduce code members to the app before diving into the puzzles.

Test solutions. If club members have difficulty with the game, they can learn from testing different solutions.

Focus on debugging. Explore different debugging strategies with members. Examples include creating a graphic representation of a puzzle on paper or as a sketch in Notes, physically acting out the code, or explaining the code as a story.

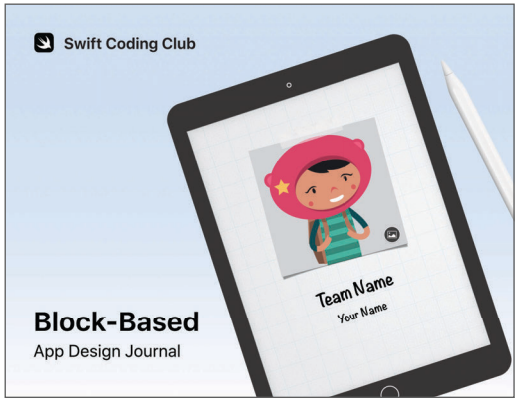


Solve in multiple ways. Each puzzle has many solutions. If members finish early, encourage them to think of different ways to solve the puzzles. They can take a screenshot of each solution and provide an explanation of the solution they feel is best and why.

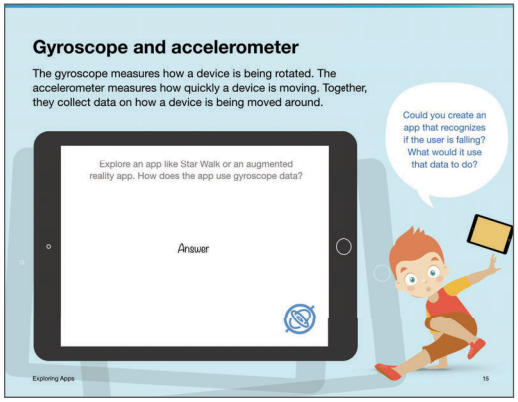
Break down the puzzles. Support struggling members in breaking more complex puzzles down and solving one small piece at a time. They can take screenshots of their solutions to refer to as they build out their full algorithms.

Predict outcomes. Ask members who are rapidly progressing through the puzzles to predict the outcome of their code before they execute it. They can take a screenshot of their code and verbally explain what they think will happen. They can also help students who are stuck by checking their code and applying the same skills to predict the problem.

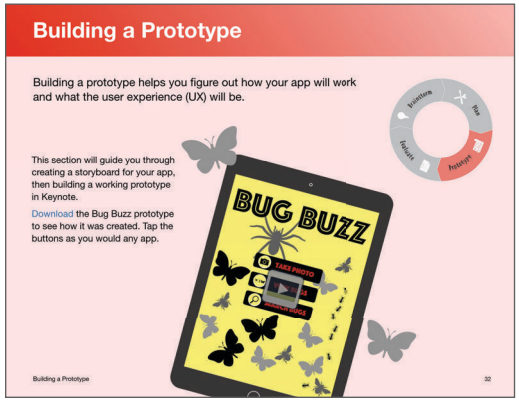
App Design Journal



Coders use this Keynote journal to learn about app features and design an app to solve a problem in the community.



Club members learn how apps work. They explore app audience and purpose, and build a toolbox of iOS features to mix and match in their own apps.



Coders design and share an app prototype. They brainstorm and plan an app to solve a local problem, then create a working prototype in Keynote.

Members create a three-minute app pitch presentation or video and celebrate their work in an app design showcase.

Need Keynote tips?

Download Keynote for iPad Starter Guide iOS 11 >

Celebrate

App design showcase

The app design process and the showcase are powerful opportunities to involve the broader community and explore the potential of apps for solving contemporary problems. The showcase is also the perfect way to show off the talents of your club members!

1. Plan the big event. Set a date for the showcase and invite students, teachers, parents, and community members to attend.

Allow time for each team to present their app pitch and to hold a short Q&A session. If you have a large group, you can split the club into two rounds where members can watch each other's pitches.

Consider finishing the event with a fun slideshow of photos taken throughout club sessions.

2. Design awards. Friendly competition can be a great motivator. Inspire club members by offering awards that recognize specific strengths in app design. Consider awards for:

- Best Engineering
- Best Innovation
- Best Design
- Best Pitch

You could also encourage audience participation with a People's Choice award.

You can download and modify this certificate for different awards.

3. Recruit judges and mentors. Judges and mentors can be teachers or staff, students with expertise in coding, experts from the developer or design industry, members of the school board, local community leaders, or individuals who would benefit from the app idea.

Judges don't have to wait until the showcase to meet the club. Consider inviting them as guest speakers to share their expertise when learners are in the brainstorming or planning phase of their app design.

4. Pick a winner. Judges can use the rubric on the next page to help them evaluate the app pitches and provide feedback. You could also share the rubric with coders before the showcase as part of the evaluation phase of the app design process.

5. Share and inspire. You may want to record the showcase presentations. Share them with the broader community and create a highlight reel to inspire future club members.

Evaluation Rubric

Download >

	Novice (1 point)	Intermediate (2 points)	Proficient (3 points)	Mastered (4 points)	Points
Pitch Content	The pitch shares key information about the app, such as its purpose and target audience.	The pitch clearly explains how the app was designed and how it improves on existing apps that have a similar purpose and audience.	The pitch explains how the app was designed to meet a market demand and how its solution is unique.	The pitch provides evidence of market demand and explains how the app design process ensures app success with key stakeholders.	
Pitch Delivery	The team explains key points during their pitch.	The team delivers their pitch with confidence and enthusiasm.	The team delivers an engaging pitch using a range of audience engagement techniques.	The pitch is well articulated, creative, memorable, and fluid across the team.	
User Interface	The UI design uses design features such as color, font, and images to appeal to the target audience.	The UI design complements the purpose of the app and has thematically consistent screens.	The UI design incorporates consistent, clear visual cues to help the user interact with the content. The prototype includes interactive elements that show how the app would behave.	The UI is elegant, concise, and pleasing, with attention given to color, layout, and readability. It gives feedback to the user about their progress through the app, or the options available to them when they perform an action.	
User Experience	The prototype enables users to explore multiple screens of the app.	The prototype expresses a clear intent for the app and how users can interact with it to accomplish a goal.	The prototype uses consistent and standard navigation techniques to provide the user with a clear and intuitive path through its content.	The prototype enables users to view and interact with its content differently depending on their needs. The prototype addresses accessibility and includes features to protect user privacy or online safety.	
Coding Concepts	The team describes how coding plays a role in making an app.	The team describes how their app design would relate to its code, such as the kind of data it stores or how it reacts to different user inputs.	The team describes how basic coding concepts like data types, conditional logic, and touch events relate to the design of their app.	The team describes specific coding tasks that would be necessary to implement their app, as well as how that code powers its screens and/or interactions.	
Comments				**Total score**	

Swift Coding Club

Block-Based Coding

Certificate of Achievement

Awarded to

For

Signature

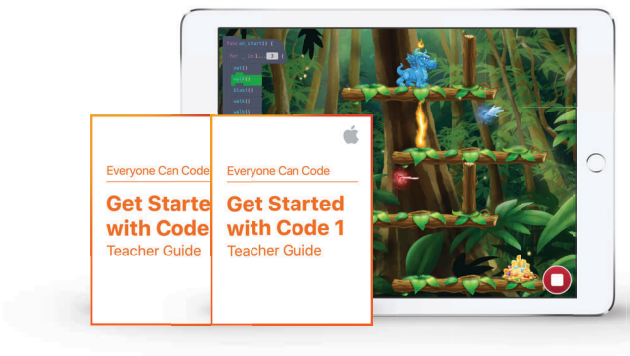
Date

Take It Further

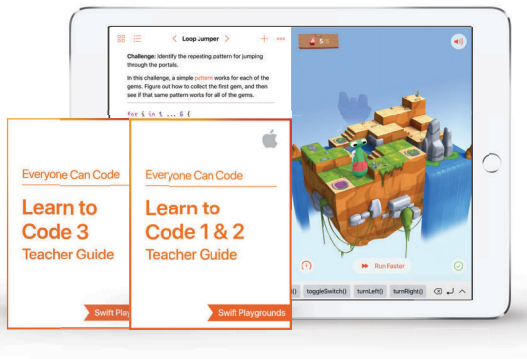
Swift Coding Club is just the beginning of your coding journey. The Everyone Can Code curriculum provides fun, supportive resources to take coders from learning the basics on iPad to building real apps on Mac.

And you don't have to stop at club activities. Comprehensive Teacher Guides also enable teachers to bring coding into the classroom, with step-by-step, curriculum-aligned lessons for students from kindergarten to college.

See all the Everyone Can Code resources >



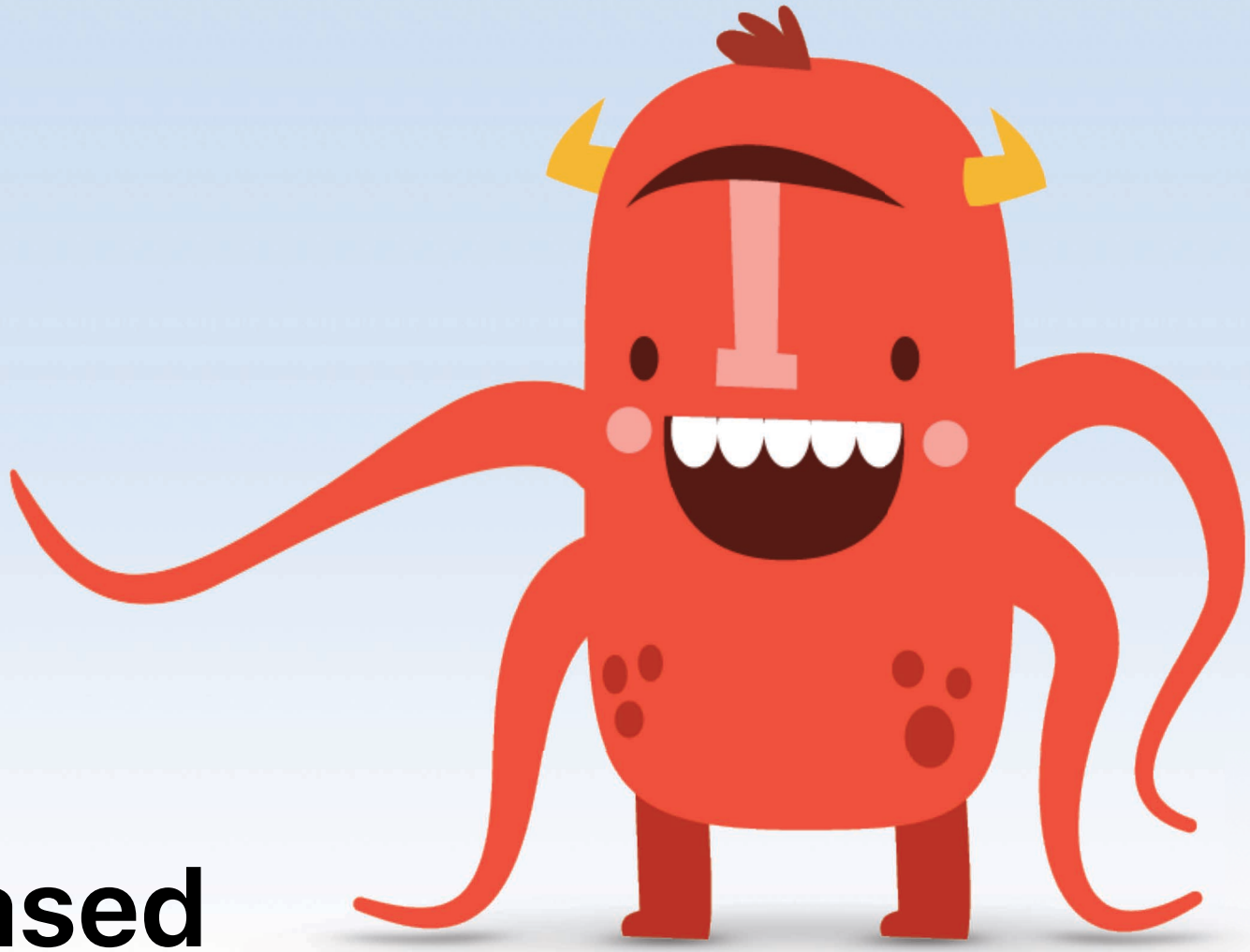
Learn more about the Get Started with Code curriculum >



Learn more about the Swift Playgrounds curriculum >

Learn more about the App Development with Swift curriculum >

Swift Coding Club



Block-Based

Coding Activities

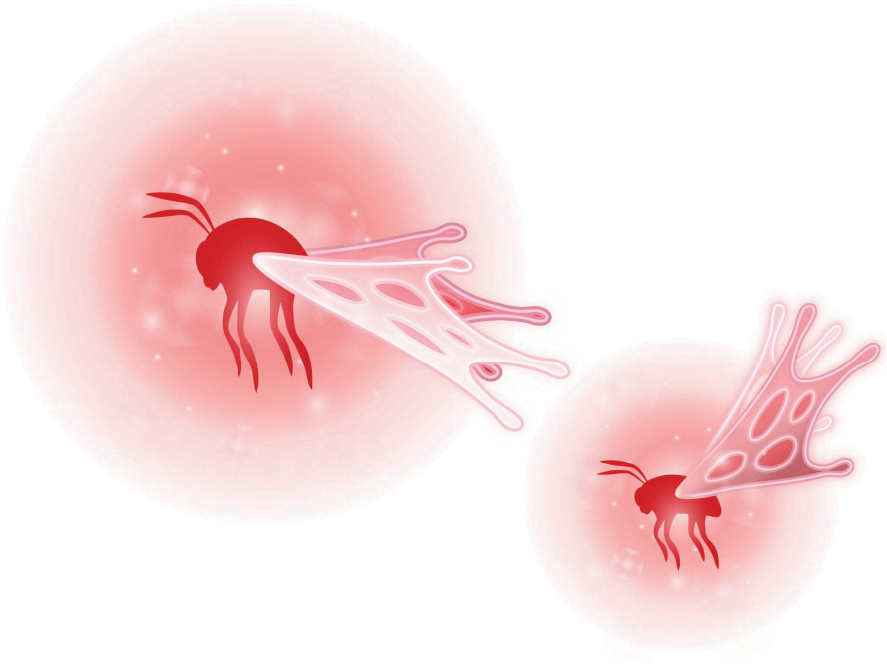
Welcome to the Swift Coding Club!

You want to learn how to code and design apps? Excellent!

As you begin to figure out how apps work, you'll see that all apps are created with code. These coding activities will help you understand coding so that you can build better apps.

Have fun!

Activities



What Is Code?

0

Introduction (10 minutes)

Computers help humans in all kinds of ways. Sometimes they help us by making a job easier or faster to complete. Other times they help us express our creativity in new ways.

But computers don't do much by themselves. They need a human to tell them what to do in a special language we call code. Using code, we can tell computers how to interpret different types of **data**, such as text, images, or audio, and how to share that data back to us.

A person who writes instructions in a **coding** language is called a **developer**. Developers around the world have been able to do amazing things by writing code.

Code is everywhere around us. With a partner, make a list of technologies that rely on code. What do those technologies do? What data do they use? What would the world look like if that technology never existed?

Coding: Telling a computer what to do.

Data: Information we collect and use. There are many types of data.

Developers: People who write code to build their own apps and games.

What Is Code?

0

Practice (15 minutes)

Set up your iPad with the Tynker and Hopscotch apps. Spend some time exploring them.

In the Hopscotch app, open the “Get started with code” collection on the “Start a new project” screen and watch the video “How to use Hopscotch.”

Think about it

- What are some ways computers help you?
- Can you think of a problem computers could help with?

Commands and Sequences

1

Introduction (20 minutes)

Computers follow instructions called **commands**. Each command tells the computer to do a certain action.

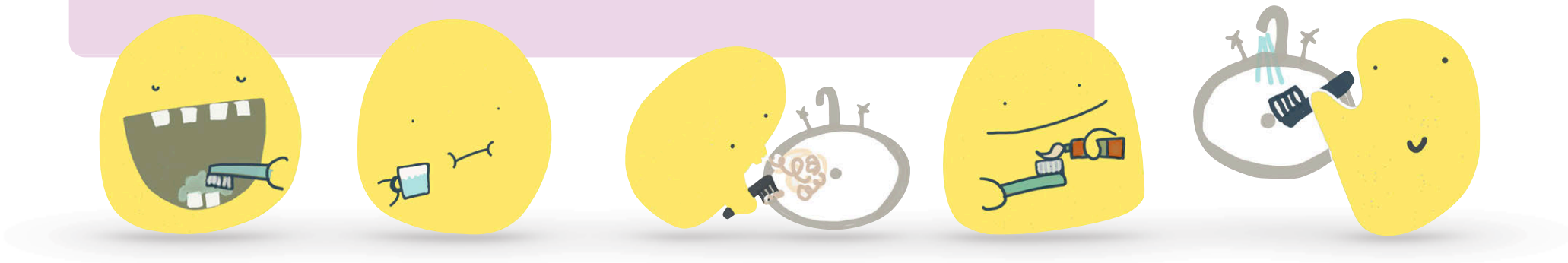
It's also important to think about the order—the **sequence**—in which we give commands. If we mix things up, we might not get the result we were hoping for.

Try giving a partner commands to perform a familiar activity, like brushing their teeth. Your commands should be clear and specific. Your partner can only do exactly what you say.

Were you able to sequence all the steps correctly and provide clear commands?

Command: An instruction to perform a specific action.

Sequence: The order in which things happen, like patterns and events.



Commands and Sequences

1

Practice (15 minutes)

In the Tynker app, solve the puzzles in Space Cadet Lessons 1 and 2. Learn the basics of sequencing and how to move the characters around.

Watch the Hopscotch "Introduction to Sequencing" and "Creating Sequences" videos and create a patterns project.

Think about it

- Why is it important to put steps in the correct sequence?
- Can you think of other activities that have a sequence?
- What would happen if we didn't tell a computer the correct sequence?

Take It Further

1

Crazy dance time (30 minutes)

Dances are choreographed sequences of moves. You're going to create your own dance sequence and teach the group.

1. Choose a song that makes you move.
2. Check out this Keynote dance sequence for some inspiration.
3. With a partner, create and photograph your own dance moves.
4. Use the photographs to each build your own Keynote dance sequences. Order and copy the slides to create your choreography.
5. Swap presentations and perform each other's dances. Could you each follow the sequence correctly?

Think about it

Even though you used the same moves, was your dance the same as your partner's? How was it different?

Algorithms

2

Introduction (20 minutes)

Now that you know about commands and sequences, it's time to put them together to solve problems. A developer who writes code to solve a particular problem is writing an **algorithm**—a step-by-step set of instructions to solve a problem or complete a task.

Algorithms are everywhere! Recipes, instructions, directions, and processes are all examples of algorithms we use every day.

Work together to list as many different algorithms as you can.

Try being a computer yourself. Have a partner command you to do something, like copy a drawing that you can't see, or tie your shoes. But only do exactly what they say, no more, no less.

And don't forget! You're a computer, so you can't ask questions. You can only do what you're told!

Take turns being the "commander."

Did you get the hang of giving clear commands in the right sequence to achieve a task? Nice job. You just created your first algorithm!

Algorithm: A step-by-step set of instructions to solve a problem or complete a task.

Algorithms

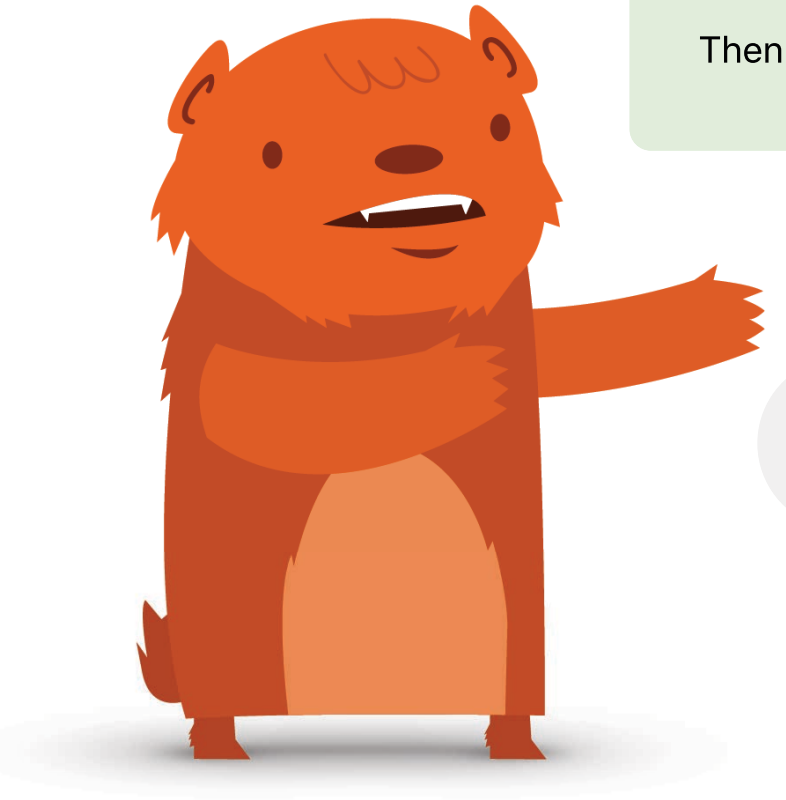
2

Practice (15 minutes)

In Tynker, solve the puzzles in Space Cadet Lesson 3 or Dragon Spells Lesson 1. How many different solutions to the puzzles could you find?

Watch the Hopscotch "Flexible Sequencing" video and explore which parts of the algorithm can be sequenced in any order as you build a face.

Then check out the "Algorithms" video and create a fun game.



Think about it

- What commands have you learned?
- What was your longest algorithm?
- How would you explain why sequencing is important in Dragon Spells and/or Hopscotch?

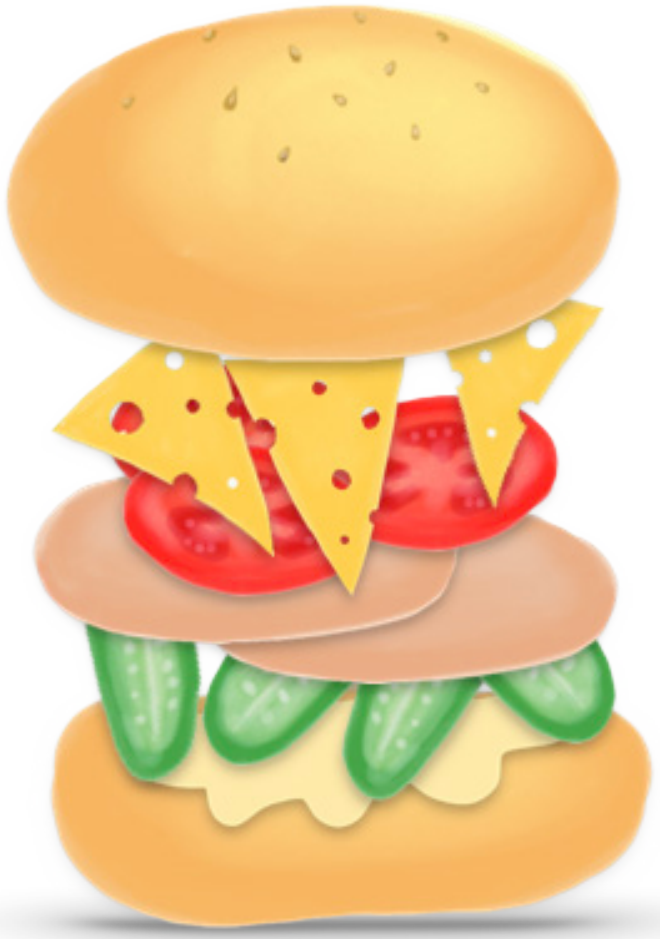
Take It Further

2

Make My Sandwich (30 minutes)

Create an algorithm to make your signature sandwich. Share your algorithm with a partner and see if they can follow the steps to create your tasty masterpiece!

1. Download the Keynote template.
2. Create your sandwich by dragging fillings into the buns. Give your sandwich a name.
3. On a new slide, write an algorithm for making your sandwich so that others can make it just the way you do.
4. Swap algorithms with a partner. Were you able to create each other's sandwiches?



Loops

3

Introduction (20 minutes)

When we talk to computers, we look for easier ways to tell them what to do. Imagine teaching a computer to read a book. You wouldn't want to have to command it to turn each page—think of all those commands! Instead, you can use a **loop**. You can look at the number of pages in the book, then command the computer to "read the page" and "turn the page" as many times as required.

Loop: An instruction to repeat a set of commands for a certain number of times.

Music is rich with loops, especially with rhythms. Try looping some rhythms to create a body percussion piece. Practice performing the loop below:

x 2

Create two new moves; you can use any part of your body.

Sequence the moves and work out how many times you need to repeat the sequence to create a 30-second performance.

Share your loop with a friend. Aren't loops an easy way to teach someone repeated sets of commands?

Loops

3

Practice (15 minutes)

In the Tynker app, solve the puzzles in Dragon Spells Lesson 3. Try to identify a repeating pattern, such as walking and jumping, then shorten the algorithm by putting that code into a loop.

Watch the Hopscotch “Loops” video and explore how to draw shapes with loops.

Think about it

- Is it easy to spot loops in sequences?
- How do loops help you when you write code?

Take It Further

3

Loopy Day (30 minutes)

One of the most challenging aspects of constructing loops is identifying repeating patterns.

Try telling the story of your day using as many loops as possible.

Here's an example to get you started:

Alarm goes off.
Wake up.
Hit sleep button.
Go back to sleep.

X 4

Alarm goes off.
Wake up.
Turn off alarm.
Get up.

Swap stories with a partner. Can you spot any loops they missed?

Think about it

- Did you spot any loops within loops? These are called *nested loops*.
- Loops help us write more efficient code. Why do you think efficiency is a good thing?

Decomposition

4

Introduction (20 minutes)

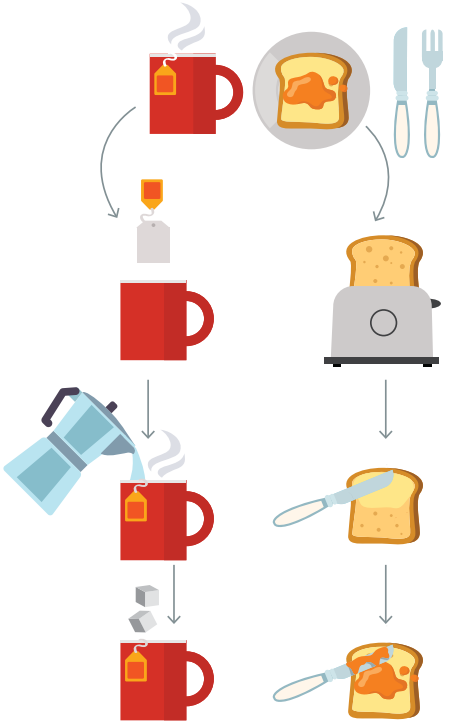
Think of a big problem, like launching a rocket into space. That problem is so big and complex, it can make your head spin. However, humans have landed on the moon. So how did we do it?

To solve any complex problem, the first step is to break it down into smaller parts. This makes the task easier to understand and tackle. Developers call this process **decomposition**. Once you solve all the small parts, you'll have solved the big problem! You can also think about this in terms of an easy problem, like making breakfast.

Learn to play a complex routine by breaking it into parts.

1. Watch this Cup Song Video and try doing the routine yourself.
2. Working in pairs or small groups, break the video into smaller sections and write an algorithm for each section.
3. Combine your section algorithms to create an algorithm for the whole routine and try performing it. You've used decomposition to solve the problem!

Decomposition: Breaking a large problem into smaller parts that are easier to understand.



Decomposition

4

Practice (15 minutes)

In the Tynker app, solve the puzzles in Dragon Spells Lesson 4. Remember to break each puzzle into smaller parts before you start trying to solve it.

Think about it

- What strategies did you use to decompose the problem?
- How can decomposition help us solve problems?

Take It Further

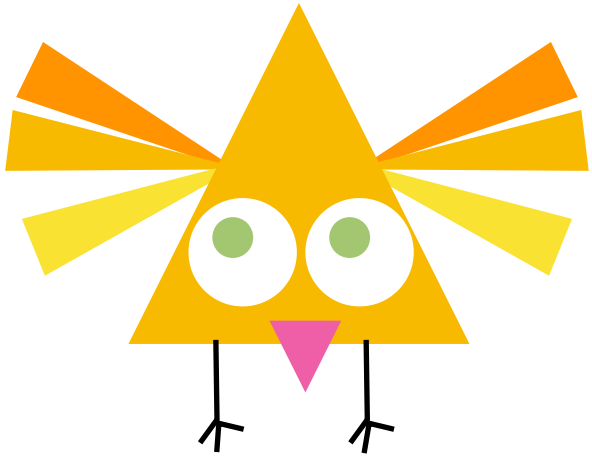
4

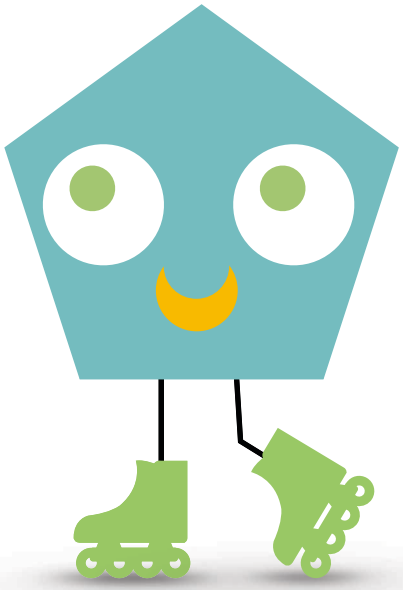
Creative Shapes Challenge (30 minutes)

Build a creature using shapes in Keynote. Get as creative as you like. The only rule is you can use only the Shapes menu.

Group all the objects that make up your creature.

Swap creatures with a friend. Your challenge is to re-create their creature and develop a set of instructions so others can re-create it too. How will you break the problem down into manageable tasks?





Think about it

- How did decomposing the problem help you to solve it?
- Now that you've decomposed the problem, would it be easier to change elements of the creature?

Debugging

5

Introduction (20 minutes)

Has your code achieved the result you were hoping for every time? You probably needed to adjust your code to get the outcome you wanted. Coding is a conversation with a computer, and sometimes communication breaks down. Working out what went wrong and then fixing it is called **debugging**. Let's practice our debugging skills.

Bug: An error in your code.

Debugging: The process of finding and fixing the error.

In the game below, the player is supposed to replace the emoji with the words in the key. Unfortunately, something has gone wrong. Can you spot the **bug** and figure out the correct quote?

Here's the key:

🦁 = secondary

❤️ = Jobs

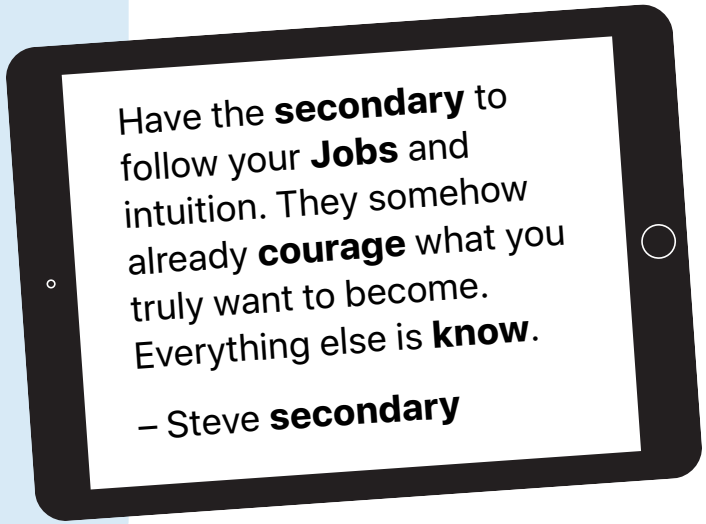
🧠 = courage

🕑 = heart

👨‍💻 = know

Here's what the game returned

What do you think was meant to happen? What part of the algorithm **is** working? How could you fix it?



Debugging

5

Practice (40 minutes)

In the Tynker app, solve the puzzles in Space Cadet Lesson 5 or Dragon Spells Lesson 2. For these puzzles, some code is given, but it's buggy! First try the code provided to see what happens, then see if you can fix it to achieve the desired result.

Watch the Hopscotch "Debugging" video to learn debugging strategies.

Think about it

- What is your main challenge with debugging?
- Have you ever had to debug something other than code?

Take It Further

5

Debugging the Day (30 minutes)

Becoming an effective debugger doesn't necessarily require you to be an expert coder. But it does require you to think logically, be persistent, and use a range of problem-solving strategies. These are skills that you can practice and use in every part of your life.

Create a comic strip in Notes, Pages, or Keynote that tells the story of a buggy day.

1. Think of three bugs you might come across in your day. These could be things like errors in your math work, a missing sock, or a broken bike.
2. For each bug, explain what happened, what was supposed to happen, and what that tells you.
3. Show the strategy you used to solve the problem. Did you use decomposition? Trial and error? Testing different solutions?

Think about it

How can you view bugs as a chance to learn instead of as a catastrophe?

Events

6

Introduction (20 minutes)

What happens when you flip a light switch on? Unless something is broken, the light should turn on. In coding, we'd call flipping the switch an action, and the light turning on an **event**. The same action of flipping a switch might also prompt other events, like a light turning off or a fan turning on.

With iPad, actions can be touches, changes of position, or even saying "Hey Siri!" Each of these actions can prompt a range of events.

Event: An action that causes something to happen.

How well can you respond to events?

1. Grab a dice or ask Siri to roll a dice for you. Roll the dice (the event) and perform the matching action with friends.
2. Come up with your own actions to make things more interesting.

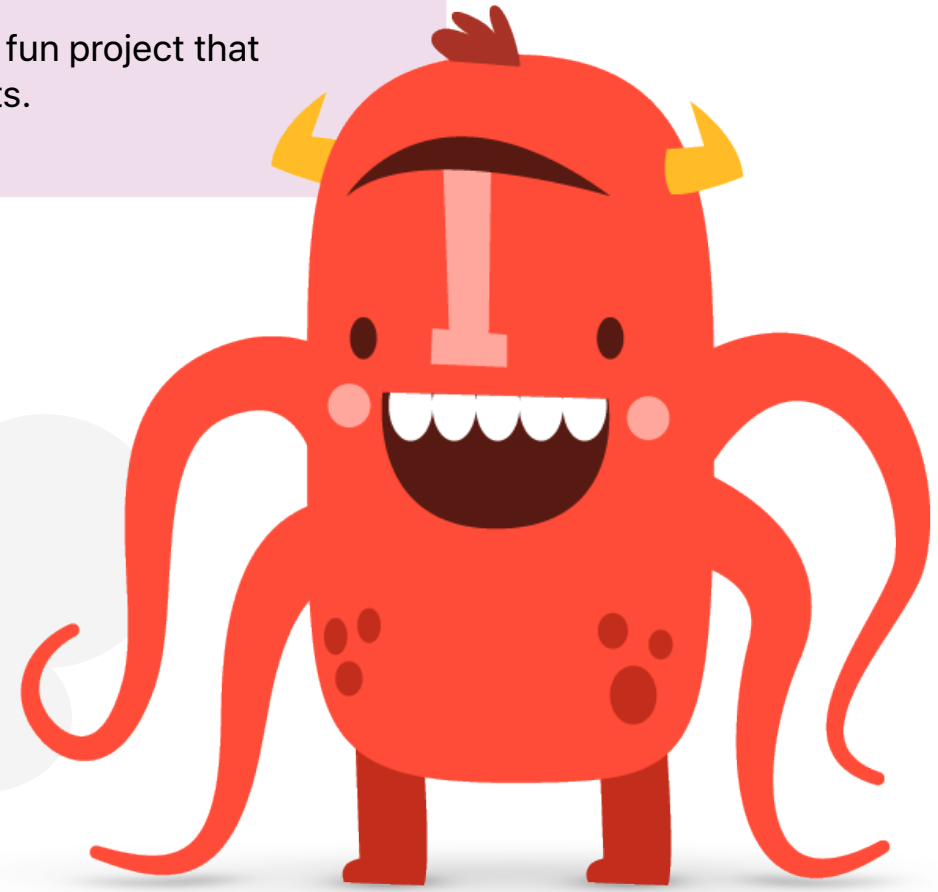
Events

6

Practice (40 minutes)

In Tynker, solve the puzzles in Space Cadet Lesson 6 and learn how to use touch, collision, tilt, and messaging events.

Watch the Hopscotch "Events" video and create a fun project that responds to touch, shaking, tilt, and collision events.



Think about it

- What sort of events happen in your favorite app?
- How do events make apps more interactive?

Take It Further

6

Interactive Information (30 minutes)

You can build touch events into a Keynote presentation to create interactive ways of sharing information. Use this Keynote presentation to create an interactive map of the coding vocabulary you've learned so far.

1. Open the Keynote presentation and tap Play. Tap the word <c*oding*> and see the definition pop up. The rest of the definitions are waiting for you to complete.
2. Pinch the slide to go back into slide view. Tap and turn on Show Presenter Notes. Presenter notes explain what information needs to be added to each slide.
3. Add definitions for each slide. Try to use your own words.
4. Tap Play again. See how your vocabulary map comes to life with touch events!

To challenge yourself further, see if you can add animations to make your concept map even more engaging. Or explore how the presentation uses links to create an interactive experience and see if you can add additional terms or relationships.

Think about it

- In your interactive map, what events occurred following the touch action?
- As you were using Keynote, what touch events did you observe in the app?

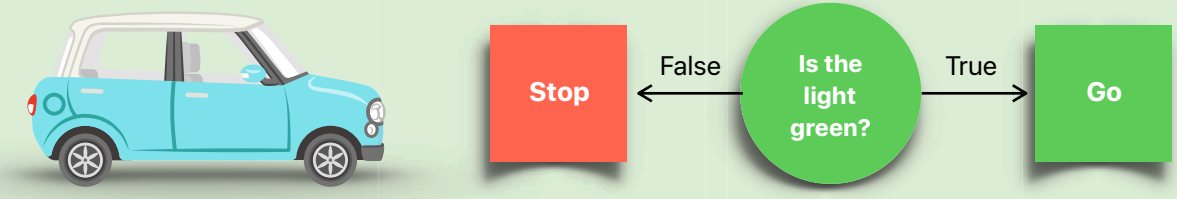
Conditional Statements

7

Introduction (20 minutes)

So far, you've written commands in a certain sequence to create algorithms to solve problems. But what if we need to plan for different situations? We need a way for the computer to make a decision.

Imagine driving a car and coming to a stoplight. The **condition** for moving forward is that the light has to be green. So, **if** "light is green" is true, keep going. But we also need an action for when the light isn't green. We need to stop! So, **if** light is green, keep going, or **else**, stop.



The game Simon Says is a great example of conditional thinking in action. We perform the action only **if** the leader says "Simon says." As a group, play a couple of rounds of Simon Says.

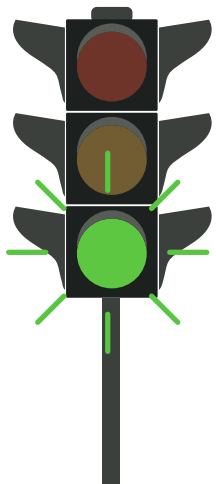
What other games are built on **conditional statements**? As a group, list as many as you can, describing the conditional statements that are key to the game.

Condition: Something we can check to tell if it's true or false.

If statement: Tells us what to do if a condition is true.

If-else statement: Tells us what to do if a condition is true, and what to do if it's not.

Conditional statement: An instruction to do something when certain conditions are true.



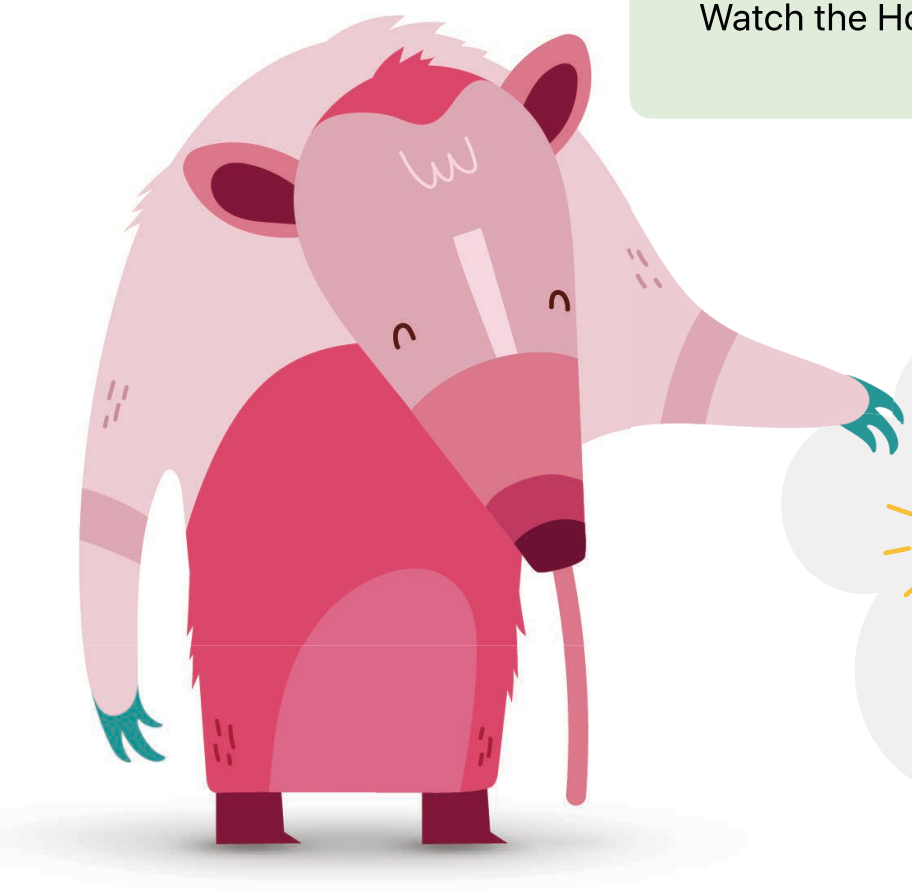
Conditional Statements

7

Practice (40 minutes)

In Tynker, solve the puzzles in Space Cadet Lesson 7 or Dragon Spells Lesson 7 and write algorithms that react to changing situations.

Watch the Hopscotch "If statements" video and create a hatching game.



Think about it

- Why would we want computers to be able to make decisions? Can you think of any examples where this would be helpful?
- List some technologies in your life that use conditional logic. What for?

Take It Further

7

Explain-a-Game (30 minutes)

Most games are built around conditional statements. These statements make up the rules. With a partner, choose a game that you both know well. It could be a sport, a board game, or any game that has rules.

1. Brainstorm all the rules you can think of.
2. Write the rules as if-else statements.
3. Create a short video in Clips that explains the game rules using if-else statements.

Think about it

- Were your game rules using if-else statements shorter than other instructions you've seen?
- How do conditional statements help us write more concise code?

Variables

8

Introduction (20 minutes)

Variables are like boxes for storing data that changes. Imagine if you had a box for each day's clothes. You'd have boxes for tops, bottoms, socks, and shoes. Every day, the contents—or value as it's called in coding—of each box would change. So the box for tops might have a red shirt in it one day, and a blue shirt in it the next.

In coding, we use variables to store values that change, such as player names, locations, game scores, dates, or images.

Try spotting variables in some common nursery rhymes like "Old MacDonald Had a Farm." How would you name the variable? What are some possible values? Now create a new version of the rhyme by making another phrase a variable, for example, "Old MacDonald Had a Zoo."

Variable: A container that stores a value. The container has a name, and the value can change.

Variables

8

Practice (15 minutes)

In Tynker, solve the puzzles in Dragon Spells Lesson 9. You'll program a simple quiz game and use variables to track the number of times an event happens.

Watch the Hopscotch "Variables" video and learn how to use a variable to code a score.

Think about it

- Can you spot variables at work in any of the apps you use?
- What variables do you use every day? Think about things like how many pens are in your backpack, how much sleep you had, or how many candies you ate.

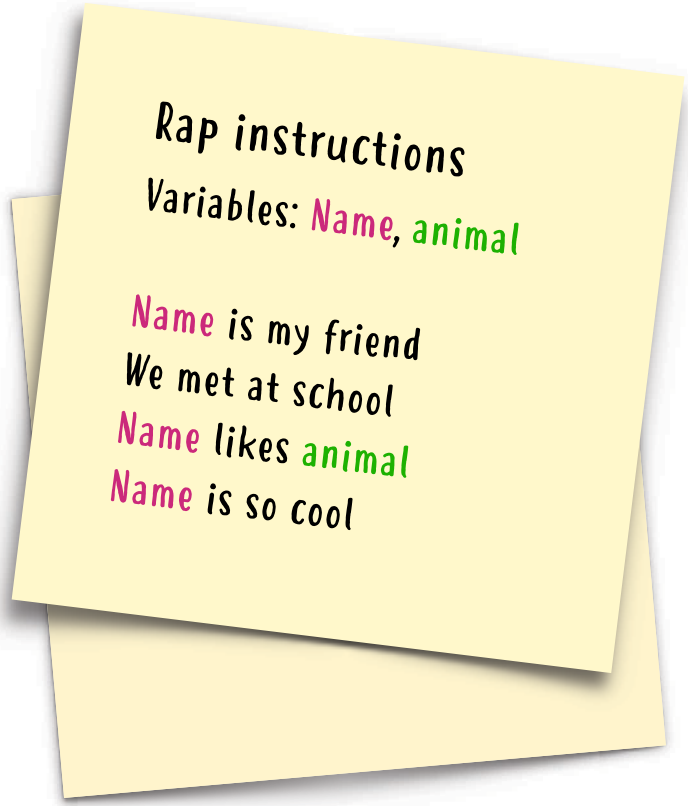
Take It Further

8

Poetry Jam Slam (30 minutes)

Create a short poem or rap about your club members using variables.

1. In small groups, write a four-line poem or rap that describes the club members. The poem or rap should contain at least two variables specific to the club members, such as name, age, favorite food, or where they're from.
2. Create a Keynote slide that includes the poem and lists the variables.
3. Perform several verses of the poem or rap for the whole club, asking different members to supply the values for the variables in each verse.



Think about it

How many variables did the club identify across all the poems and raps?

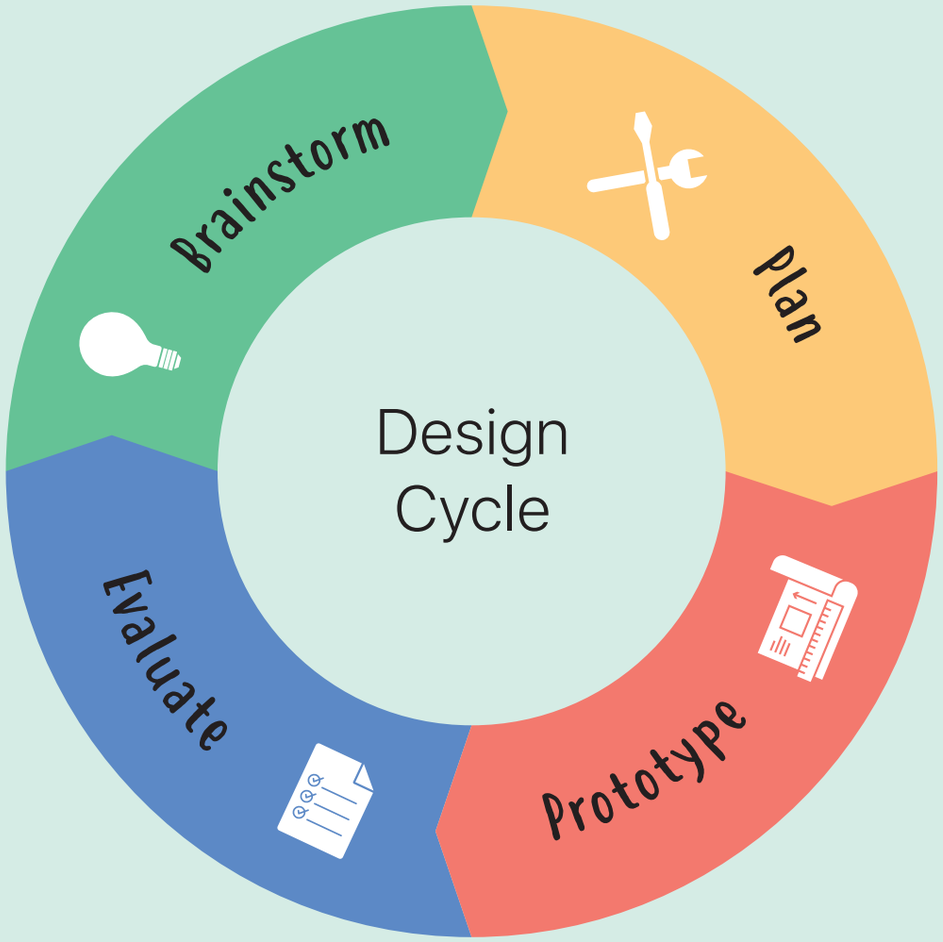
Swift Coding Club

Block-Based

App Design Journal

Welcome to the Swift Code Club!

So you want to be an app designer?
You've come to the right place. This journal will help you learn all you need to know about how apps work, and it will guide you through creating your own app idea, designing a prototype, and pitching your concept.

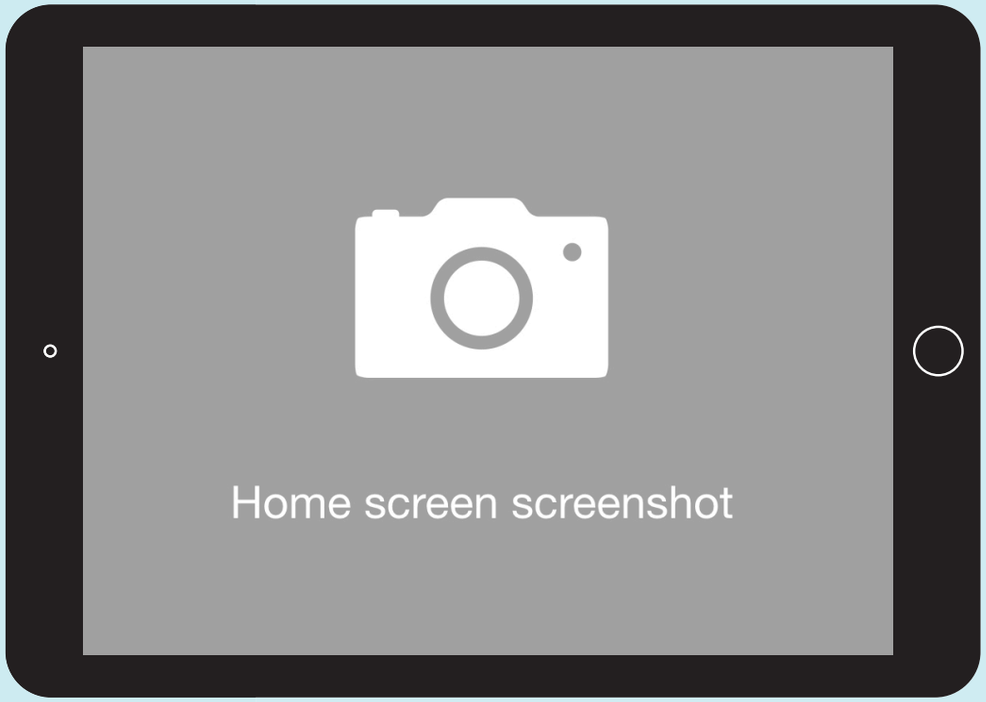


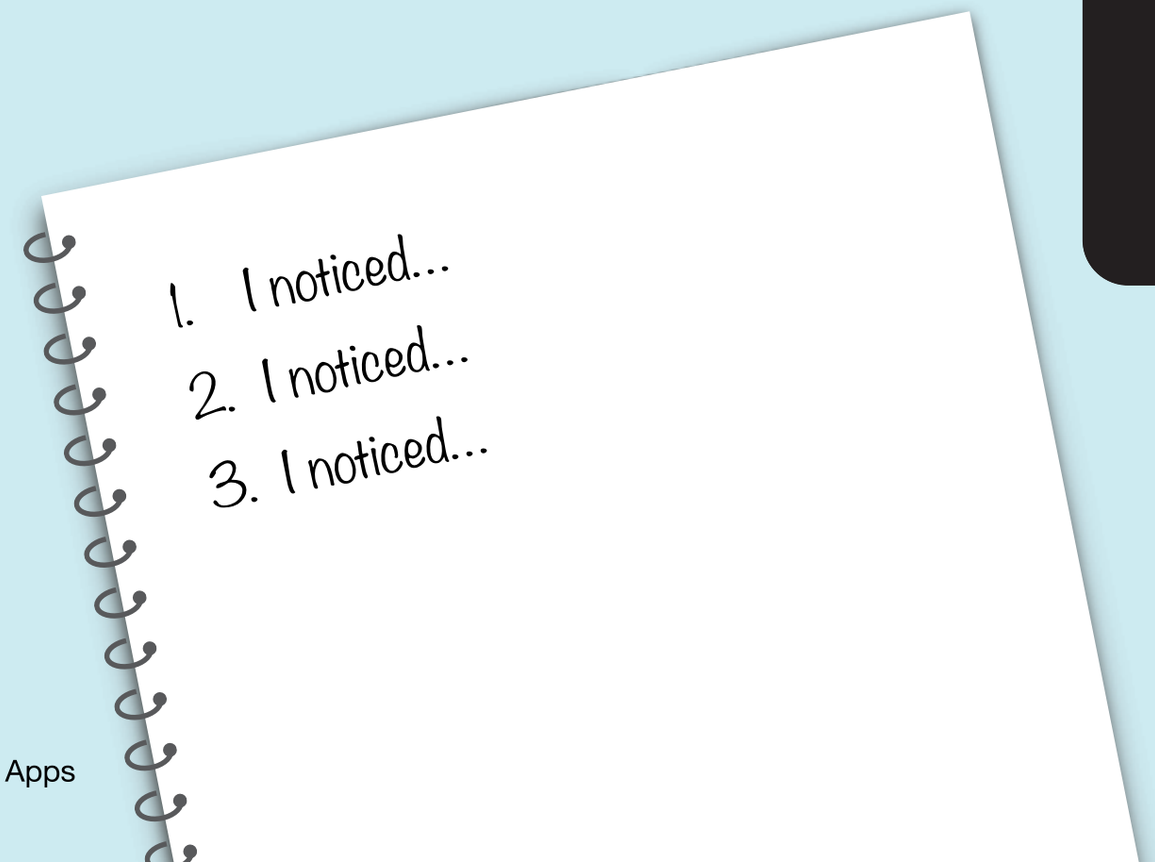
But first you need to make this journal your own.
Go ahead and add your name and a suitably cool photo to the cover slide. Oh, and you'll need a catchy company name. This is just the beginning of great things!

Exploring Apps

Before you can start creating your own app, it's a good idea to explore what apps are designed to do and how they do it. As you investigate different app features, you'll start to figure out what makes some apps great and others, well, not so great!

Take a screenshot of your iPad Home screen, then insert it in the iPad frame to the right. What do you notice about the apps on your Home screen? Are they similar or different? Do they do the same things? Describe some things you notice about the apps on your Home screen.





App purpose

You'll have noticed that apps on your Home screen have different purposes. Add each app to a category on this page. If one doesn't fit easily, you can put it in the "Other" category.

Creation

App name

Entertainment

App name

Information or Learning

App name

Communication

App name

App name

Other

Looking at the apps in your "Other" category, what new app purposes could you add?

New app purposes

Knowing the purpose of your app is a really important first step in the design process.

App audience

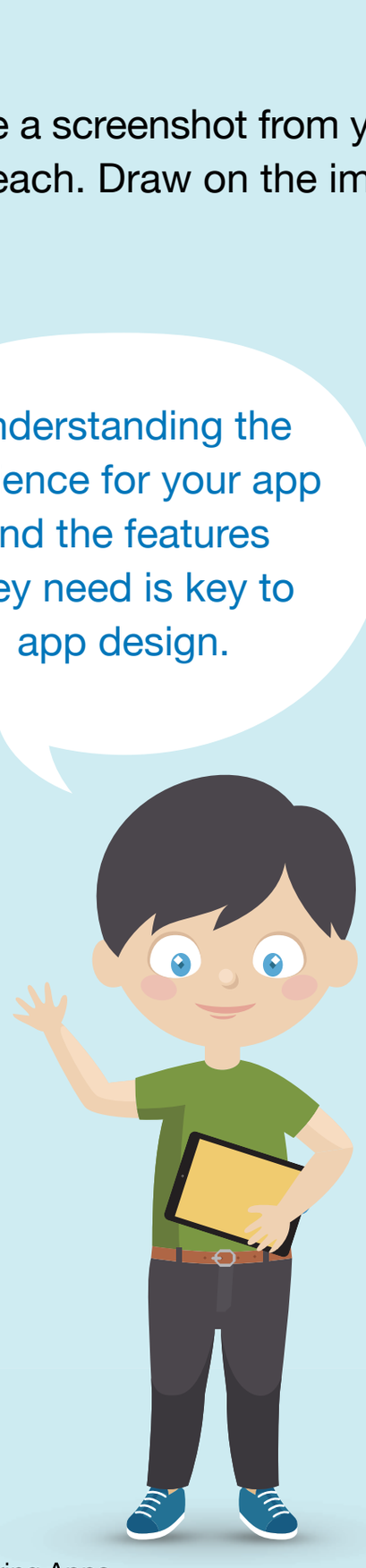
Apps are designed for specific audiences. Look at the example below from Tynker.

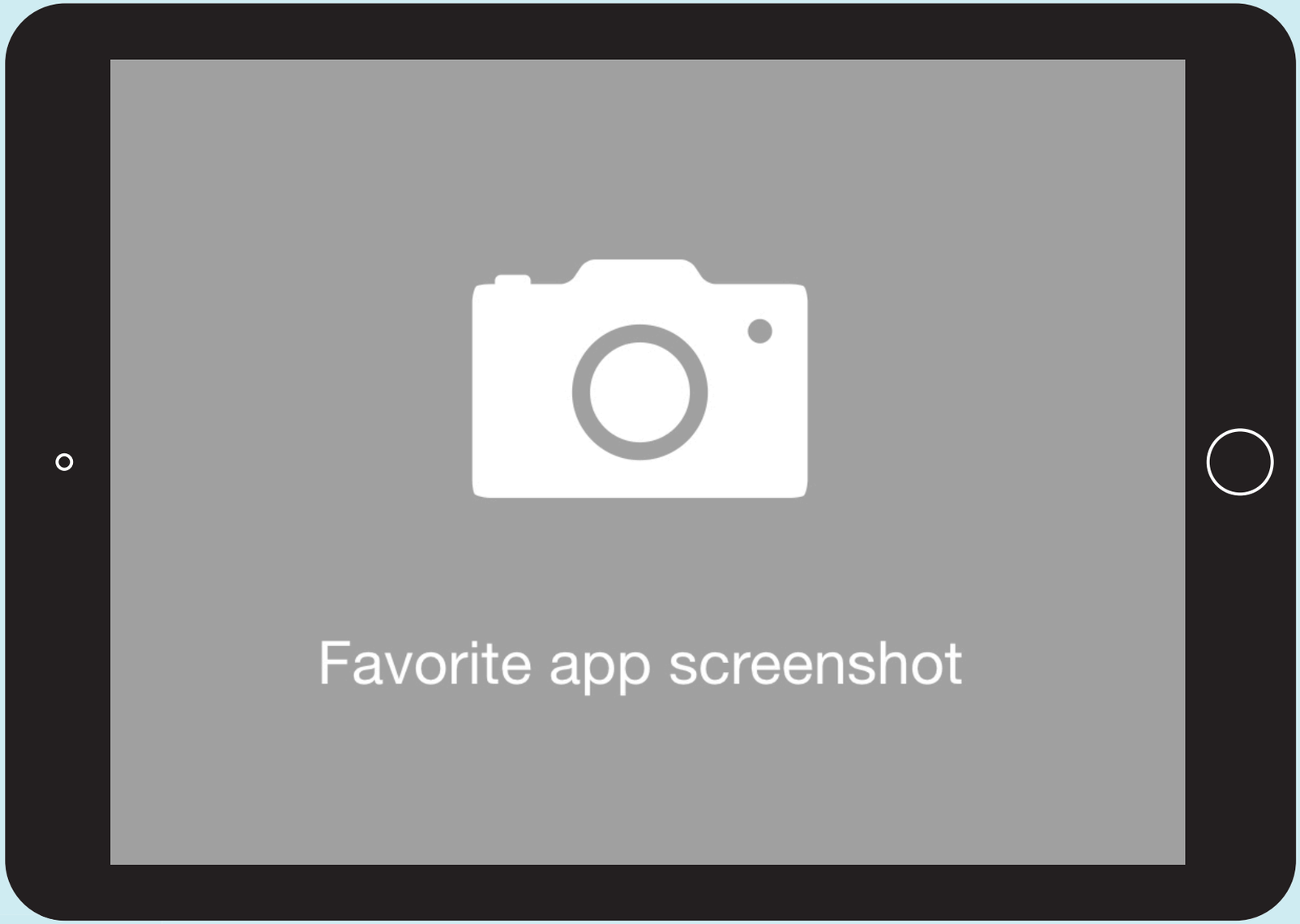
Who is the main audience
for Tynker? How can you tell?

Answers

Take a screenshot from your favorite app, and see if you can spot some clues about the audience it's trying to reach. Draw on the image to identify key features.

Understanding the audience for your app and the features they need is key to app design.





App design toolkit

There are tons of iOS features you can use to design great apps. You'll explore each one, then learn how it works and what's possible. Each feature lets you collect and use data in interesting ways.

Keyboard

Have you noticed how a keyboard automatically pops up when you need to type in an app? Apps use keyboard data—letters, numbers, or even emoji—to do a range of things, such as translating a phrase into another language, locating specific information, or finding an address on a map.

Just for fun: Use your emoji keyboard to write your favorite film title. Can anyone decipher it?

Favorite film emoji name

Do you know any apps that use the keyboard? What do they do with keyboard data?

App name

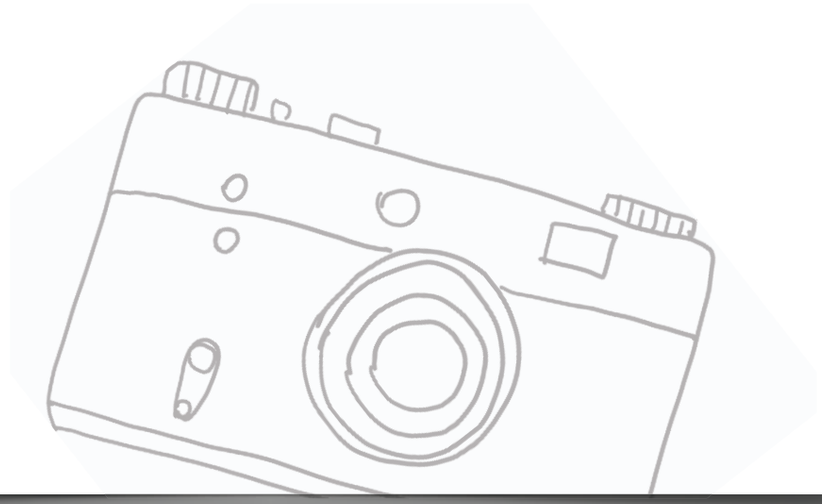
Camera

The camera collects image data as pixels. Each pixel has a particular color and brightness. Apps can store that data as an image. But you can change the data, like when you use effects in Photos.

Try it yourself! Choose an image from Photos and insert it in the first box. Now apply different effects. Can you explain how the Photos app is working with the pixels in each image?

Can you think of any other apps that use Camera data in creative or useful ways?

1. App name and description
2. App name and description
3. App name and description
4. App name and description
5. App name and description



Do any of your apps change background color when it's dark?

Isn't that a clever use of camera data?

Microphone

Sound is another type of data you can collect. Apps like Messages let you send audio messages. And Siri collects audio data to understand your requests.

Find an app that uses microphone in an interesting way and describe what it does. Tap the + menu in Keynote, then add an audio file to describe your app.

Just for fun: Try saying this to Siri, “Hey Siri, read me a haiku.”

Have you discovered any other funny responses from Siri?

Now turn on Enable Dictation for your iPad. Go to **Settings** > **General** > **Keyboards** and turn on **Enable Dictation**. Tap the text box below, then tap the microphone icon on the keyboard to dictate your response.

How is using microphone data to add audio files different from dictation?

Response

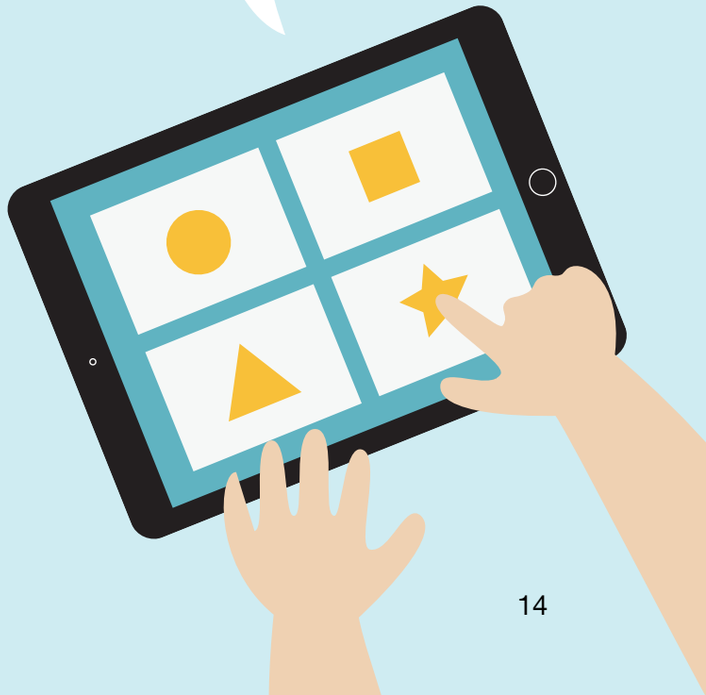
Touchscreen

One of the most important features of iOS apps is the touchscreen, which lets users tap, swipe, and drag to interact with the app.

Open an app like Tynker or Hopscotch, and explore the different ways you can use touch to interact with the app. Describe the gesture and action below.

Response

In code, you can program a touch event to trigger something to happen.



Gyroscope and accelerometer

The gyroscope measures how a device is being rotated. The accelerometer measures how quickly a device is moving. Together, they collect data on how a device is being moved around.

Explore an app like Star Walk or an augmented reality app. How does the app use gyroscope data?

Answer

Could you create an app that recognizes if the user is falling? What would it use that data to do?

GPS

GPS stands for Global Positioning System. You can use GPS data—latitude and longitude—to describe precisely where a device is located.

Open the Maps app. How does the app use GPS data? What other data is included in the Maps app?

Answer

When would it be handy to have an app share your location? Are there online safety or privacy considerations?

Bluetooth

Bluetooth enables an app to connect to a nearby device, such as a digital thermometer, robot, speaker, or light switch. The app sends data to the device and receives data back from it.

Explore a connected device like Sphero, Meebot, Lego® MINDSTORMS, or a Parrot drone.

What data is being sent **to** the connected device?

Answer

CD

What data is being received **from** the connected device?

Answer

CD

In your world, what devices would you like to control with an app? Share your idea with a friend.

User Interface

The user interface, or UI, is what users see when they use an app. The UI includes colors, images, fonts, buttons, and navigation tools.

Compare three learning or information apps to see how common UI elements are used for navigation. Draw icons from the apps in the table below.

App	Menu	Home	Replay	Next	Favorite
Tynker					NA

Another key feature of an app is its icon. A well-designed app icon helps the app stand out on the App Store and on the device. A great app icon is simple, easily recognizable, and captures the purpose of the app.

Add images of your favorite app icons in the boxes below and explain why they're great.

App name

Description

App name

Description

App name

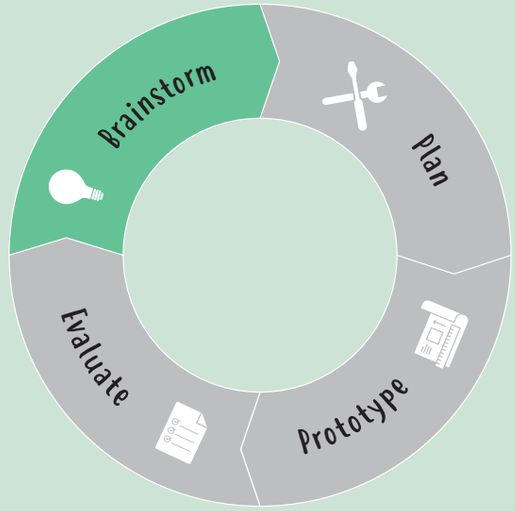
Description

Does the app icon match the app UI? How?

My App Idea

Apps can be purely entertaining, and they can also help people learn new ideas, solve problems, connect with others, or create something amazing.

Now it's time to put your developer hat on and design your own app. This section will guide you through how to brainstorm app ideas and plan your app.



Who’s on the team

App developers rarely work alone. Someone might come up with the initial idea, but then they get help from others to bring that idea to life. This image shows five main roles in an app design team.

I come up with great new **IDEAS** for apps!

I work on the **DESIGN** of the app.

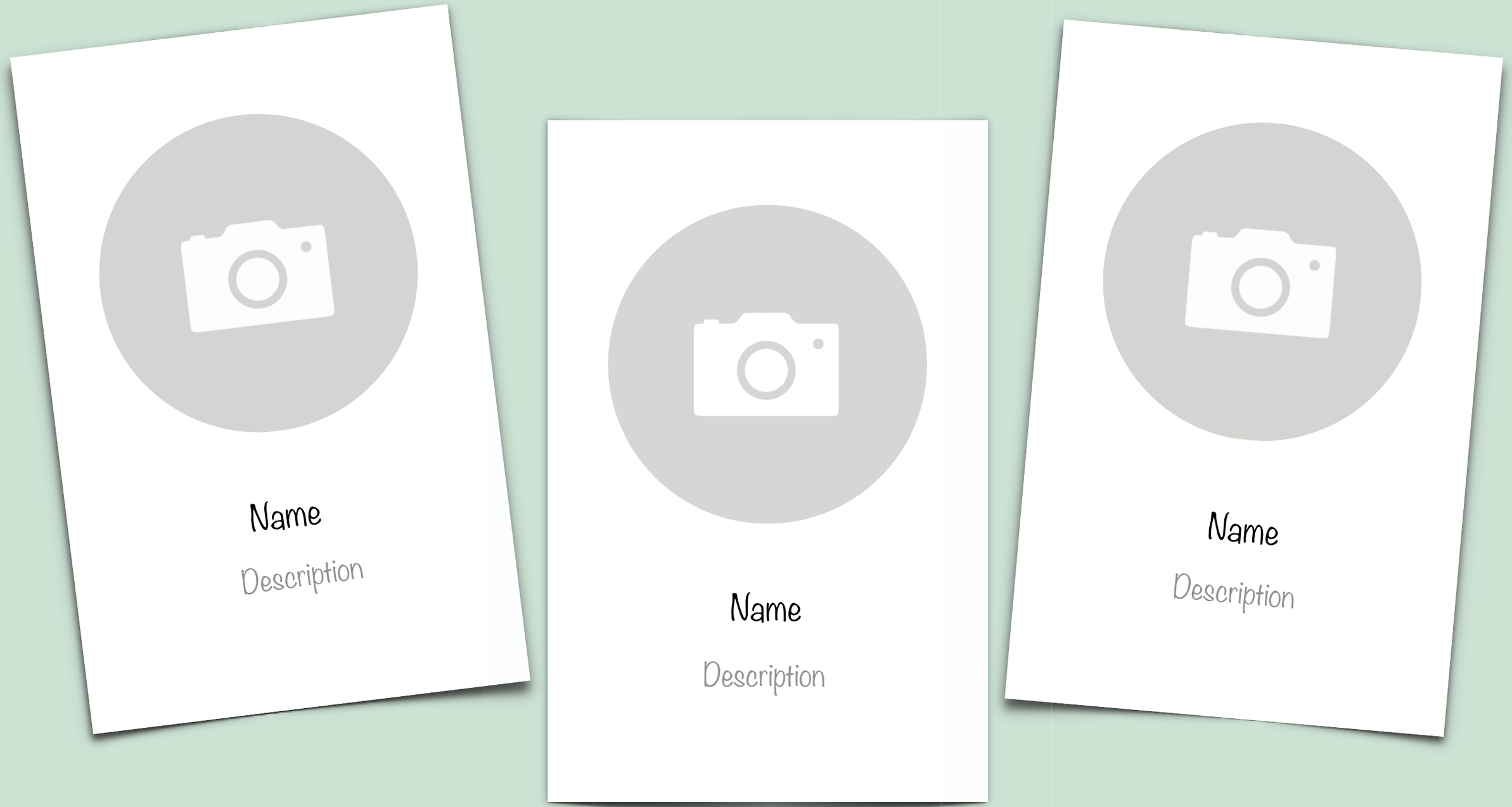
I do the **RESEARCH** to see how people will use the app.

I do the **CODING** for the app.

I **TELL** people about the app.

Who’s on your team?

Form a team of 3–5 club members. Add their photos to the circles below, and describe the role each person will have and what their strengths are. (Just copy this slide if you need more roles).



What's the problem?

Before we start designing an app, we need to understand the problem we're trying to solve.

As a team, brainstorm some problems at school, at home, or in your community.

Problem ideas

Now choose three problems to explore more. Add text, video, or audio to explain your ideas.

What's the problem?

Answer

How could an app help?

Answer

What's the problem?

Answer

How could an app help?

Answer

What's the problem?

Answer

How could an app help?

Answer

Stuck for ideas? Ask others for their thoughts, or check the App Store for apps that solve a similar problem.

Understanding the problem

As a team, choose the problem you're going to focus on solving. Copy it into the middle box. Now do some research and complete each of the cloud questions.

Why is this a problem?

Answer

Answer

What other solutions are already available?

Problem

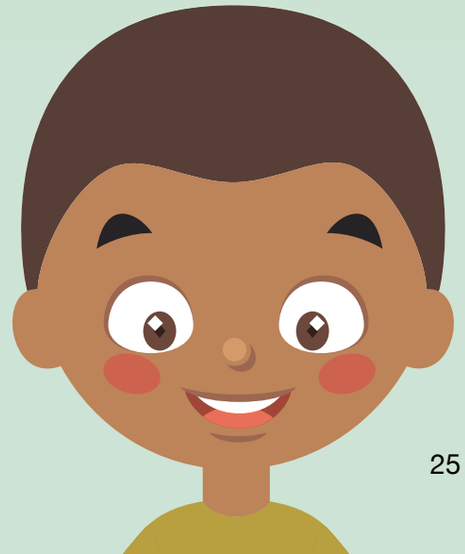
Answer

Who is affected by this problem?

Answer

Who cares about this problem?

The more you understand your problem, the better your app will be.



Think about the audience

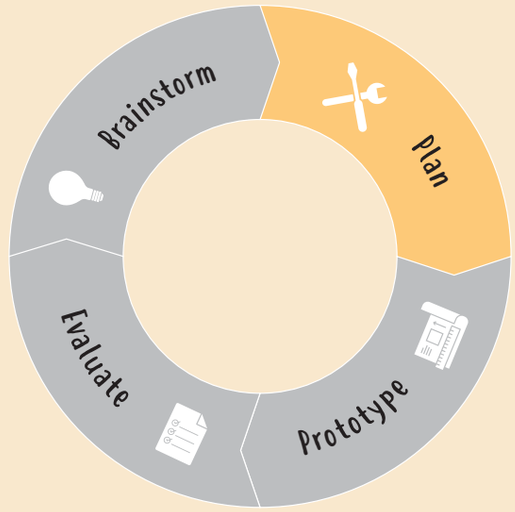
Apps are designed with specific audiences in mind. It's important to be clear on who your audience is and what they need.

Draw a picture of the type of person who would use your app. Add comments to explain what you know about them, and the type of features they'll want.

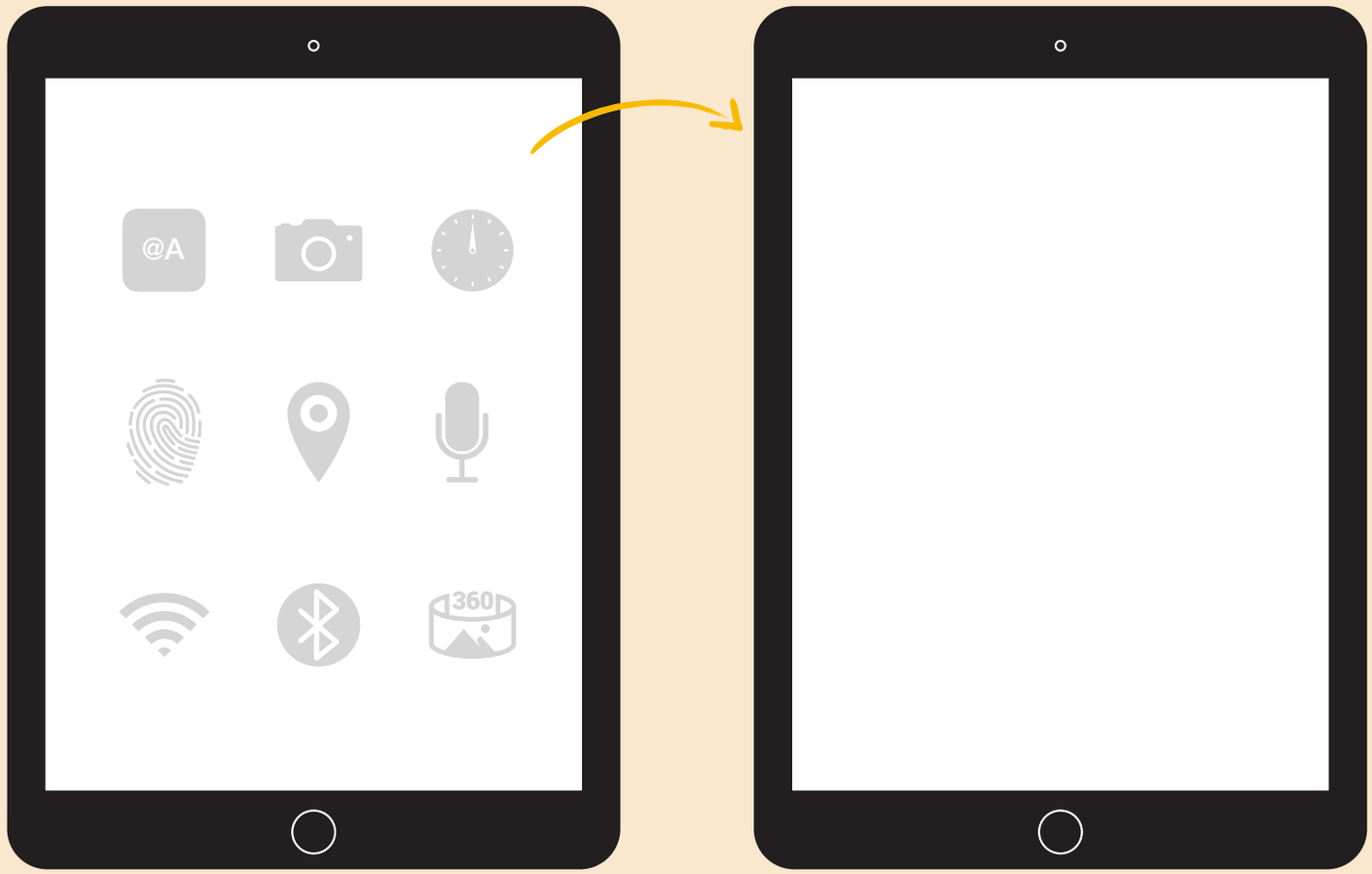
My Audience

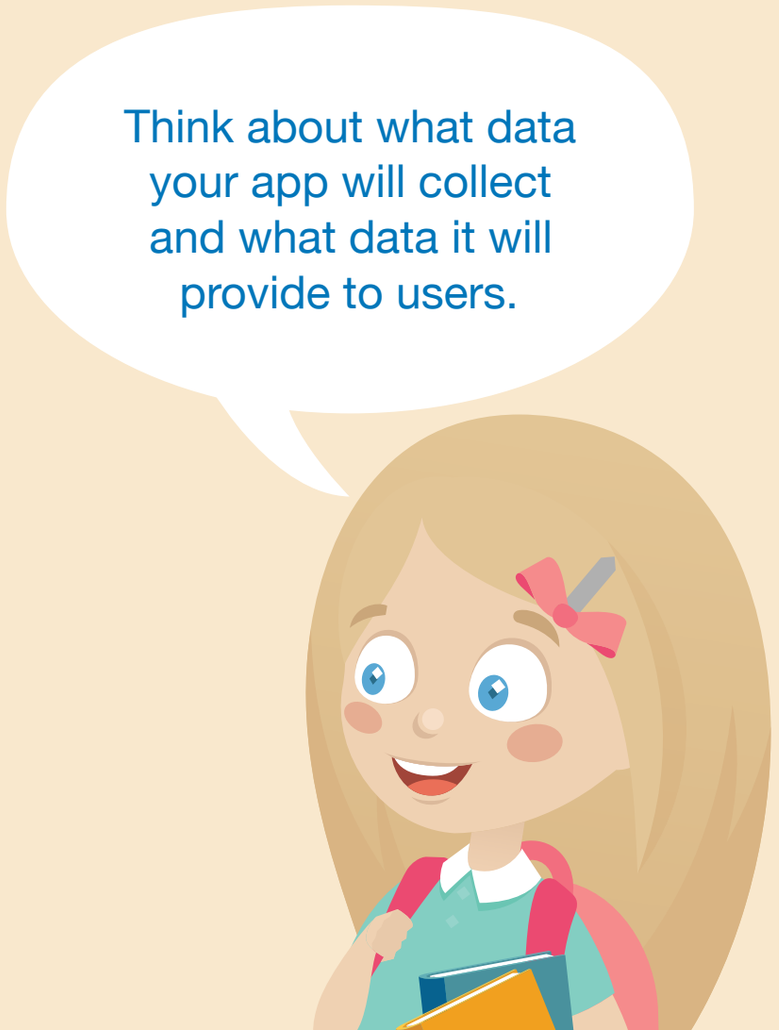
App Features

Remember the app design toolbox from slide 8? Now it's time to get creative with those features to solve your problem AND meet the needs of your audience.



Choose the app features your app will use and drag them into the iPad on the right. Add a brief description of what your app will do with that feature.

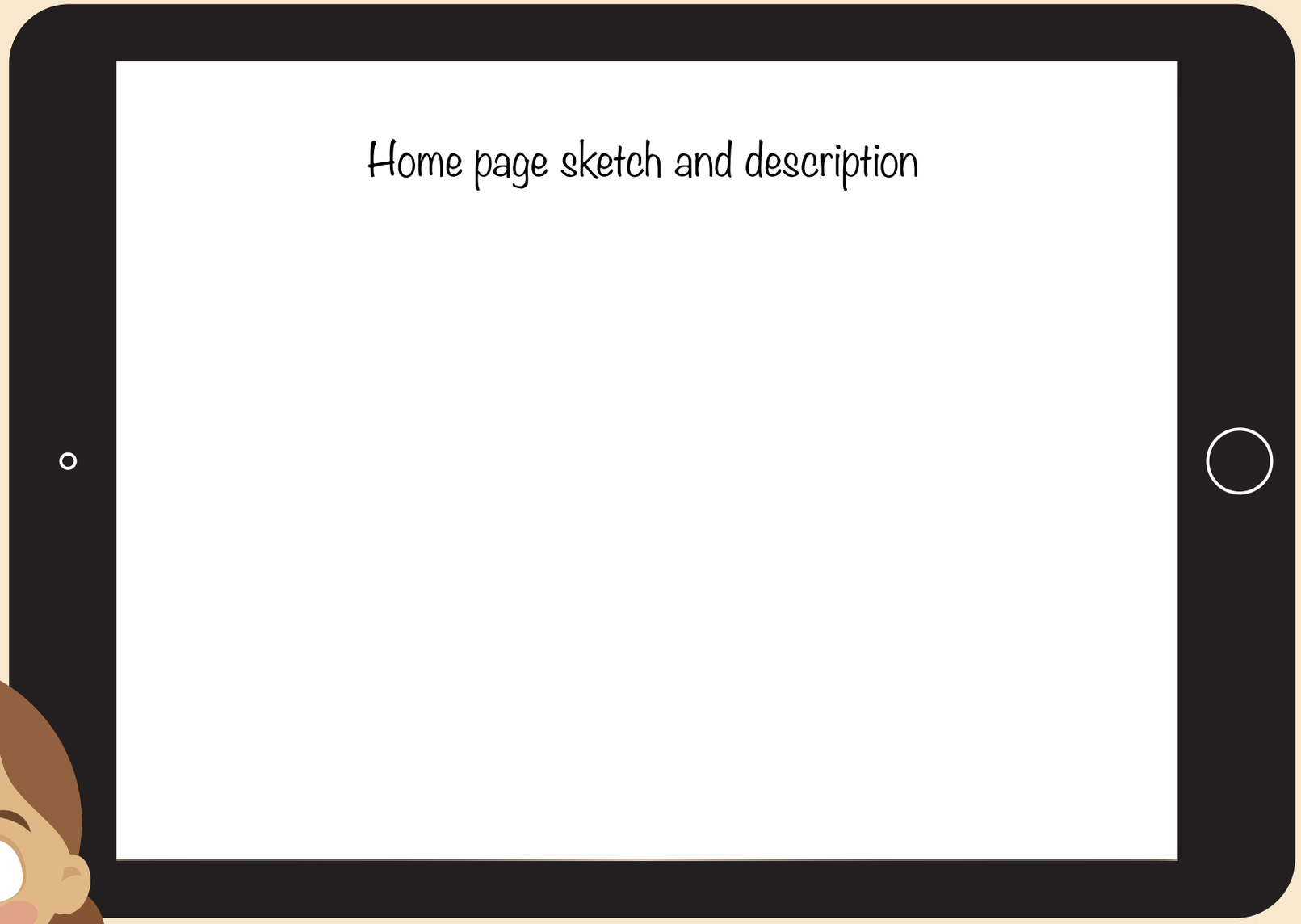




Design the app UI

How do you want your app to look? Think about the colors you'll use, the font, any images, and navigation symbols. Each team member should sketch an idea for the home page.

Add your sketch to the iPad, then share the sketch with your team. As a team, decide on a final sketch. It might be one person's idea, or a new idea that combines the best parts of everyone's sketches. Add that final sketch to the next slide.



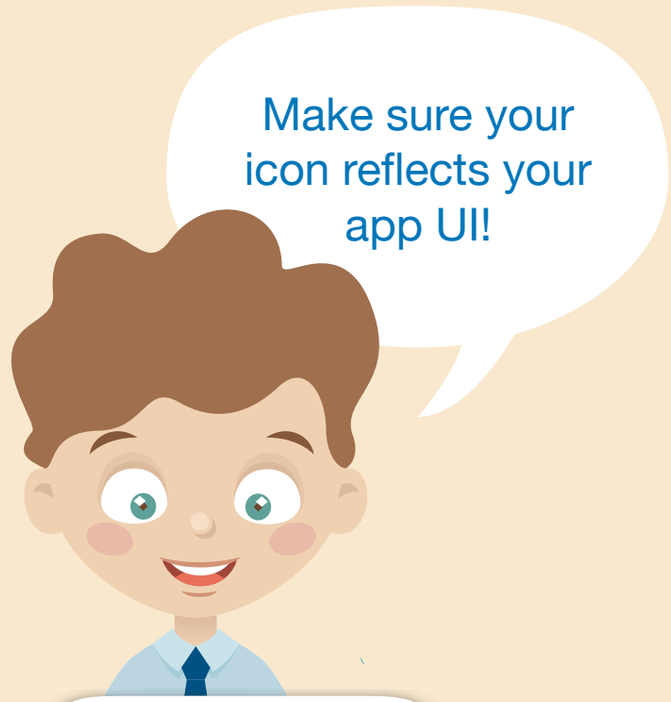
Add your final UI sketch inside the iPad frame. Then add any notes about the look and feel of the app, ideas for sounds, or how users might input or access data. (Copy this slide to add more ideas).

Home page sketch and description

Try to maintain a consistent look and feel across your app.

It’s all in a name. And an icon!

Your app is going to need a catchy name and an icon that will look great on users’ screens and in the App Store. Don’t forget that icons on screen can be quite small.



As a group, brainstorm app names and decide on one.

App names

Sketch icon ideas to share with your group.

As a group, decide on a final icon idea and have one person design the final version. Add it below. Then ask some other club members, family, or community members for feedback and record their response.

App name

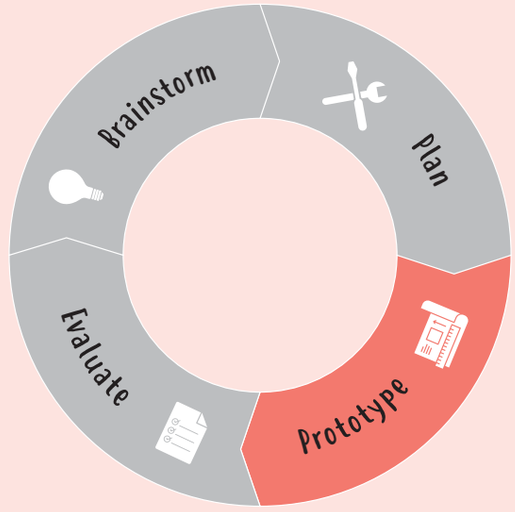
Feedback

Feedback

Feedback

Building a Prototype

Building a prototype helps you figure out how your app will work and what the user experience (UX) will be.



This section will guide you through creating a storyboard for your app, then building a working prototype in Keynote.

Download the Bug Buzz prototype to see how it was created. Tap the buttons as you would any app.

Building the storyboard

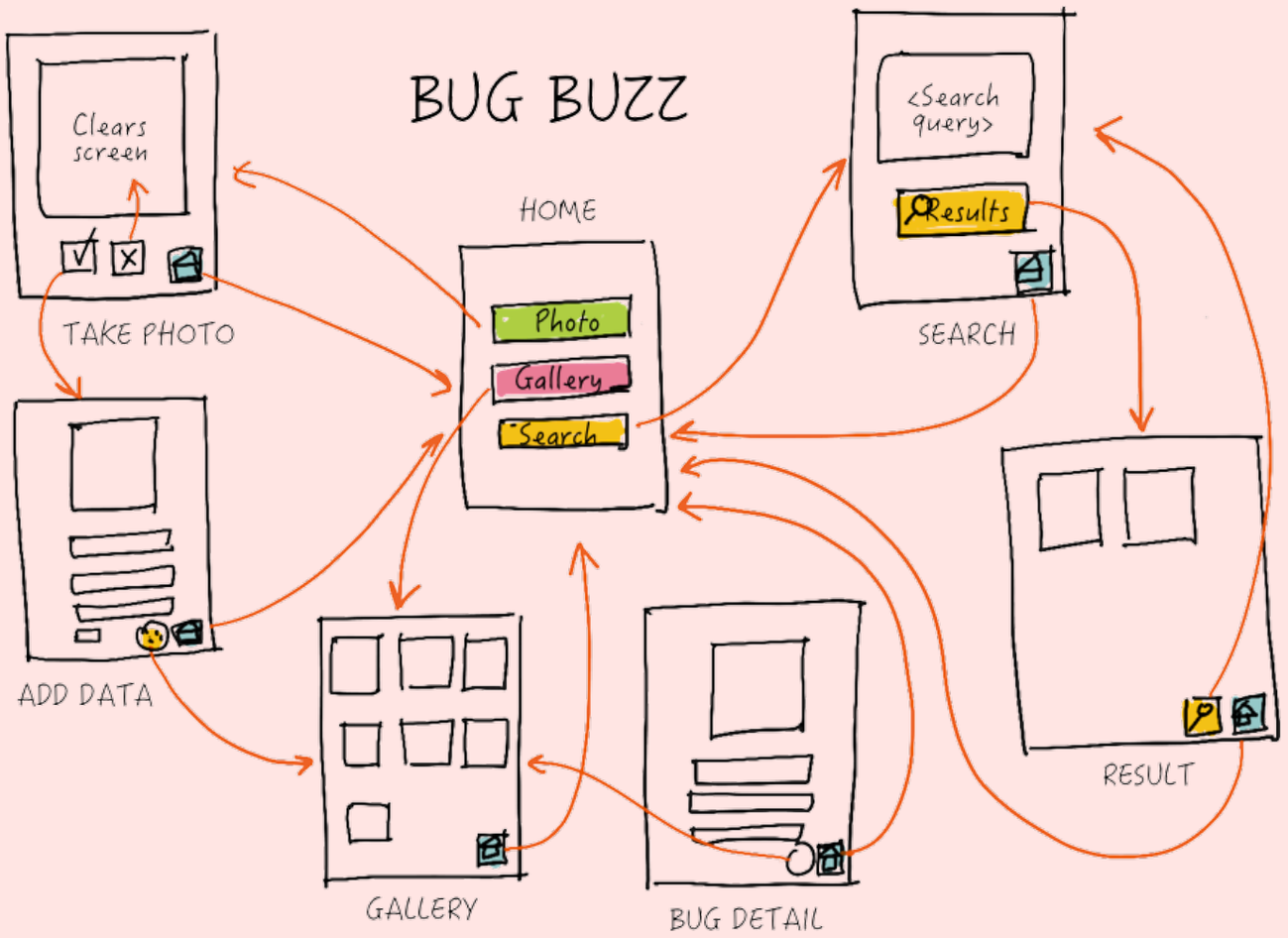
To build the storyboard, you'll need to think through all the screens your app will use and how the user will move from screen to screen. An easy way to do this as a group is to use small pieces of card or paper.

1 Sketch each screen your app needs on a card. Give the screen a clear, descriptive name for easy reference.

2 Now stick the cards to a big piece of paper or a whiteboard, and add arrows to show how users can move from screen to screen.

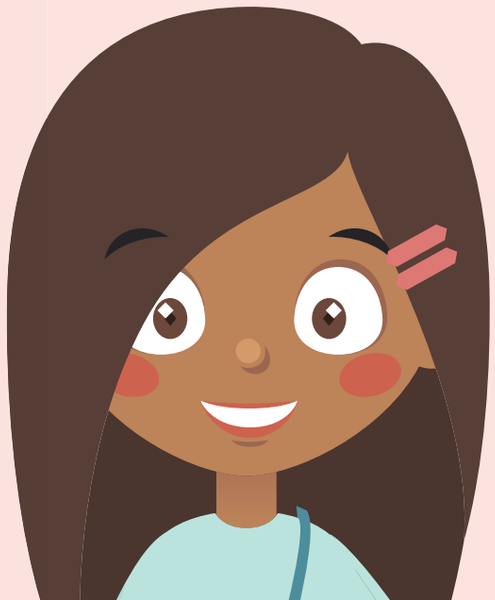
3 As you work out the connections, add navigation button sketches to your screens.

4 Take a photo of the final storyboard flowchart and add it to the next slide.



Add a photo of your storyboard flowchart here.

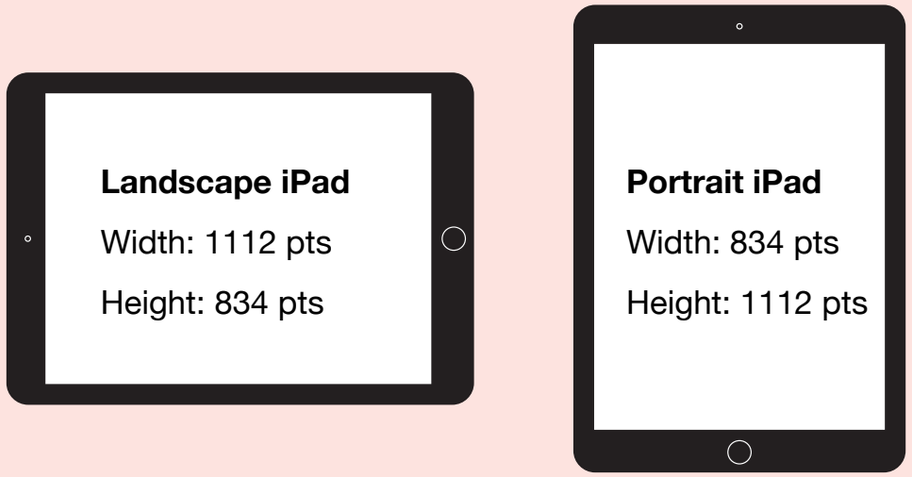
Check that every screen connects to at least two other screens!



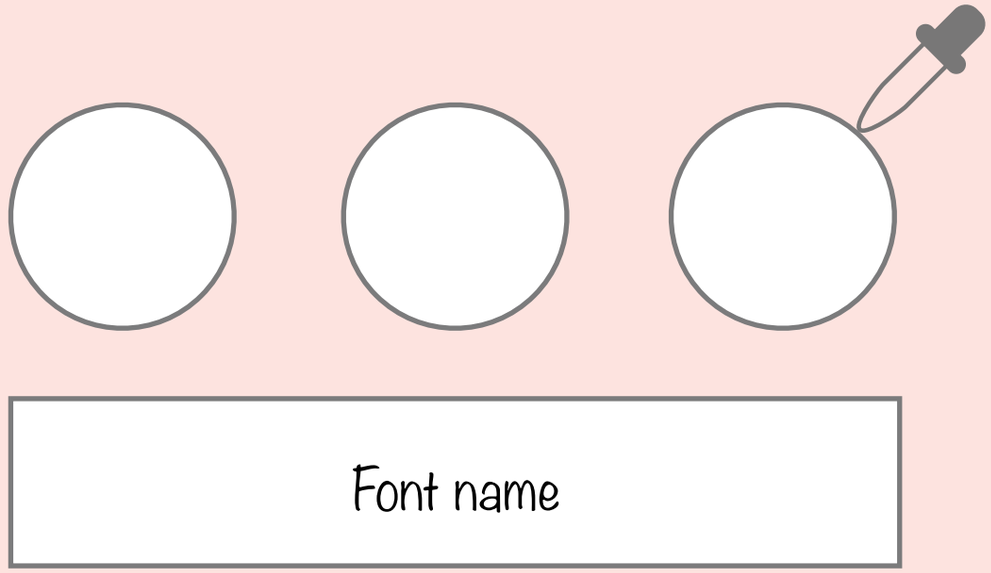
Creating a Keynote prototype

Follow the steps below to create an interactive Keynote prototype.

1 Set up your Keynote file to be the right size for your app prototype to run on iPad. Tap the More button •••, then tap Document Setup > Slide Size > Custom. Enter the dimensions below.



2 Decide on the colors and fonts you'll use, and design the navigation buttons. Put these design tools in working slides that you can delete later.



3 Give each group member screens to build. Build one screen per slide.

4 Combine the completed slides in a single presentation.

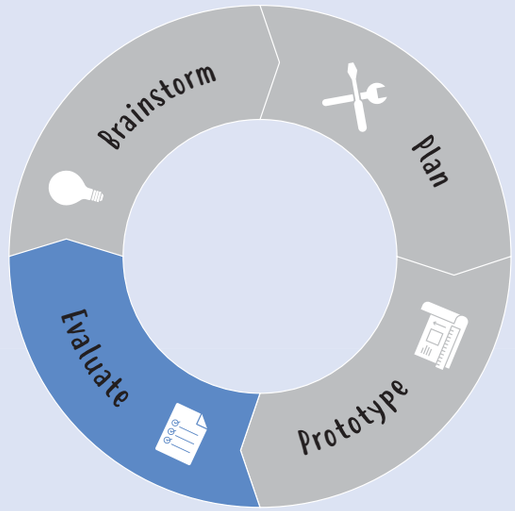
5 Create interactive links between the slides so that their buttons actually trigger touch events. Tap the object you want to link, tap Link, then Link to Slide.

6 To make sure that the presentation changes slides only when the viewer clicks the navigation buttons, tap the More button •••, then tap Settings > Presentation Type > Links Only.

7 Add animation to show how screens transition from one to another.

Evaluate Your App Idea

Woohoo! You have a working prototype! Now it's time to get some feedback. Share your app prototype with your friends and family, and if possible, people who fit your target audience. Describe the problem you're trying to solve, then observe how they use your app, ask them questions, and record their feedback.



Questions you ask could include:

- Do you think this is a good solution?
- Would you use the app?
- What would you change about the app?

Record your observations for each user, and add an audio or video file with their feedback.
Copy this slide to add more user feedback.

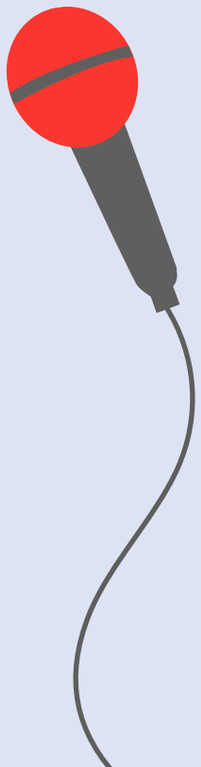
Name

Name

Observation

Observation

User feedback



As a group, discuss the feedback you received. Now create a list of the issues you need to address.

1. Issue
2. Issue

Build Your App Pitch

You've tested and improved your app idea. Now it's time to share it with the world! Your app pitch should cover the why, who, what, and how of your app.

Why The problem your app is trying to solve

Who The audience for your app

What An overview of what the app does

How Details about the UX and UI

Include your app prototype and explain:

- Your design approach
- How the app collects and uses data, including user input
- How the app uses coding concepts like conditional logic and events
- The user feedback you received and improvements you made

Make a three-to-five minute presentation or video that introduces your team and presents your app pitch.

Reflection

Well done. You've not only learned all about how apps work, you've actually designed one! And now you're probably bursting with amazing ideas about what you want to build next. But first, just take a moment to reflect on the skills you've developed through the club and your strengths as an app designer. What's your app design super power?

Record your reflections here, and just for fun, draw yourself as an app design super hero. Because you are!