

TECHNOCode

Teacher Guide

Lessons for Elementary and Middle School Students



Technology Course
using

Scratch

Design a fun Activity Studio for kids.

Spark an interest in computer science! In this STEM project, students become coders that design a fun Activity Studio for kids using Scratch. Through discovery and exploration, they learn how to create a series of hands-on activities that children will enjoy playing. The young programmers apply computational thinking to build algorithms that sequence commands, events, loops, and conditions. They learn how to construct scripts to develop animated scenes, mazes, interactive stories, and games. Additional challenges extend coding skills to create artwork, compose music, produce a diorama, and more! After each project, students complete coding journal logs to help them to think like a programmer.

TECHNOKids®

Copyright © 1993 – 2024 TechnoKids Inc.
All Rights Reserved

Table of Contents

Introduction

Introduction.....	i
How to Use This Guide.....	ii
TechnoCode Overview	iii
Implementation and Technology Integration Ideas.....	v

Session 1 Become a Programmer

Session 1 Become a Programmer	1
Session 1 Getting Started	2
Assignment 1 Computer Programs and You	9
What Does a Programmer Do?	9
Technology and Daily Life	10
Assignment 2 Think Like a Programmer	11
What is a Program?.....	11
Debug Like a Programmer	12
Invent a Game Like a Programmer.....	12
Do You Think Like a Programmer?.....	13
Assignment 3 Join the Scratch Community	14
Join Scratch	14
Assignment 4 Become a Scratch Programmer	16
Open Scratch	16
Get to Know the Scratch Project Editor.....	16
Build Your First Script to Make an Event Happen on the Stage.....	17
Edit the Motion of a Sprite to Change the Number of Steps.....	18
Make the Sprite Look Like it is Saying Something.....	18
Add a Sound to a Sprite	19
Control the Sound to Make It Repeat.....	19
Take the Scratch Challenge	20
Save the Project and Exit Scratch	20
Assignment 5 Create an Animated Scene.....	21
Create a New Project.....	21
Explore the Sprite Library to Add Two Friends to the Stage	21
Delete the Scratch Cat Sprite.....	22
View and Change Information About Each Sprite.....	22
View the Costumes of a Sprite.....	23
Build a Script to Make a Sprite Change Costumes.....	23
Explore the Sound Library to Add an Effect to a Sprite.....	24
Compare Block Placement to Play a Sound	25
Explore the Backdrop Library to Add a Background Image to the Stage.....	25
Take the Scratch Challenge	26
Save the Project and Exit Scratch	27
Session 1 Review: Get to Know Scratch	28
Session 1 Skill Review: Edit an Animated Scene	30
Session 1 Extension Activity: Edit Your Scratch Public Profile	32

Session 2 Build an Aquarium

Session 2 Build an Aquarium.....	34
Session 2 Getting Started	35
Assignment 6 Explore Motion Blocks to Make a Fish Swim	43
Create a New Project.....	43
Add a Fish to the Stage.....	43
Rename a Sprite.....	43
Start the Script	44
Move a Sprite Back and Forth	44

Add a Wait Block to Control the Movement	44
Turn a Sprite to Rotate It	45
Go to a Random Position	45
Glide a Sprite to Move It and Adjust the Speed	46
Loop the Swimming Action	46
Take the Scratch Challenge	46
Save the Project as Fish Tank and then Exit Scratch.....	46
Assignment 7 Plot Where a Fish Goes on the Stage	47
Where is the Dot on the Stage?.....	47
Add Another Fish to the Tank and Move It on the Stage	48
Move the Sprite to a Specific Point on the Stage.....	48
Glide a Sprite from One Point to Another.....	49
Set the Direction and Rotation Style to Control Movement	49
Exit Scratch.....	49
Assignment 8 Use Logic to Have a Fish Switch Directions	50
Add Another Fish to the Tank and Change its Costume	50
Build a Script that Bounces a Fish When at the Edge.....	51
Take the Scratch Challenge	51
Exit Scratch.....	51
Assignment 9 Use If-Then Logic to Control Motion.....	52
Think About If-Then Logic	52
Build a Script that Makes a Fish Swim if Touched by Another	53
Take the Scratch Challenge	53
Exit Scratch.....	53
Assignment 10 Paint an Animated Aquarium	54
Open the Paint Editor to Create a Backdrop	54
Fill the Tank with Water using a Rectangle	54
Paint Coral in the Fish Tank with the Brush	55
Reshape the Points in the Coral	55
Draw the Pebbles at the Bottom of the Tank Using a Line.....	55
Add a Silly Message Using Text	56
Build a Sign for the Text.....	56
Complete the Backdrop	56
Assignment 11 Share the Fish Tank with Viewers	57
Complete the Checklist.....	57
Add Viewing Instructions to the Fish Tank Project Page.....	58
Share the Project with the Scratch Community (Optional)	58
Exit Scratch.....	58
Coding Journal: Aquarium Reflection.....	59
Session 2 Review: About Motion Blocks.....	61
Session 2 Skill Review: Dance to the Beat	63
Code a Sprite to Dance to a Beat	63
Take the Scratch Challenge (Optional)	64
Add Viewing Instructions	64
Exit Scratch.....	64
Session 2 Extension Activity: Draw Artwork with the Pen	65
Part One – Draw a Square	65
Part Two – Stamp a Design.....	66
Part Three – Sketch a Drawing	67
Part Four – Create Artistic Designs.....	68
Session 3 Design a Maze	
Session 3 Design a Maze	69
Session 3 Getting Started.....	70
Assignment 12 Plan a Maze.....	76
Plan the Purpose of the Maze.....	76

Think Like a Programmer to Plan the Maze	76
Assignment 13 Move a Character Using Arrow Keys.....	78
Create a New Project and Save It	78
Add a Sprite and Rename It Character	78
Make the Character Move Using the Up Arrow Key	78
Duplicate a Script to Make the Character Move Using Arrow Keys.....	79
Exit Scratch.....	79
Assignment 14 Paint a Maze Backdrop	80
Open the Paint Editor to Create a Maze.....	80
Fill the Maze with Color.....	80
Draw the Path Using a Brush	80
Add Start and Finish to the Maze	81
Complete the Maze.....	81
Size the Character to Fit Inside the Path.....	81
Exit Scratch.....	81
Assignment 15 Code a Maze.....	82
Place the Character at the Start of the Maze	82
Add a Condition to Keep the Character Inside the Path.....	83
Test the Maze to Debug the Problem	83
Build a Script to End the Game	84
Take the Challenge	84
Exit Scratch.....	84
Assignment 16 Share the Maze Project with Players.....	85
Maze Checklist	85
View the Maze Project Page to Add Playing Instructions.....	86
Share the Project with the Scratch Community (Optional)	86
Play Maze Games (Optional)	86
Read Comments (Optional)	86
Exit Scratch.....	86
Coding Journal: Maze Reflection	87
Session 3 Review: Debug the Script.....	89
Session 3 Skill Review: Catch Me If You Can Game.....	91
Build the Game	91
Answer Questions About the Catch Game	92
Session 3 Extension Activity: Invent an Instrument.....	93
Paint the Instrument	93
Turn the Mouse Pointer into a Sprite	93
Edit Three Sound Clips to Make Them Unique.....	94
Code Each Button to Play a Sound Clip.....	94
Code Each Key to Play an Instrumental Sound.....	95
Prepare the Project Page.....	95
Answer Questions about the Instrument.....	95
Session 4 Broadcast a Story	
Session 4 Broadcast a Story	96
Session 4 Getting Started.....	97
Assignment 17 Explore Looks Blocks	105
Create a New Project and Insert a Moving Sprite.....	105
Explore Say and Think Blocks.....	105
Set or Change Costumes	106
Pick Two Backdrops	106
Select or Switch Backdrops.....	107
Adjust the Size.....	107
Experiment with Graphic Effects	108
Hide or Show a Sprite.....	108
Explore Building Scripts with the Looks Blocks.....	108
Exit Scratch.....	108

Assignment 18 Plan a Scene in a Story.....	109
Organize Story Ideas.....	109
Assignment 19 Introduce the Story.....	110
Open New Project and Save It.....	110
Add a Backdrop to Pick the Setting of the Story.....	110
Select the Main Character and Object in the Story.....	110
Play the Story by Clicking on the Main Character.....	110
What Does the Character Say or Think?.....	111
How Does the Character React to What Is Happening?.....	111
How Does the Character Move?.....	112
What Backdrop Appears to Spotlight the Object?.....	112
How Does the Character React to the Object?.....	113
Exit Scratch.....	113
Assignment 20 Switch Backdrops to Enhance Story Action.....	114
What are the Backdrops in your Story?.....	114
Trigger an Action When the Backdrop Changes.....	114
Take the Challenge to Create a Story Ending (Optional).....	115
Exit Scratch.....	115
Assignment 21 Direct the Timing of Events Using Broadcast.....	116
Plan to Broadcast Messages.....	116
Broadcast a Message to Hide an Object.....	117
Receive a Message that Triggers the Object to Hide.....	117
Show an Object Using Broadcasting.....	118
Broadcast a Message and then Wait for a Friend to Say Something.....	118
Broadcast a Message to More Than One Sprite.....	119
Take the Challenge.....	119
Exit Scratch.....	119
Assignment 22 Complete the Story.....	120
Story Checklist.....	120
View the Story Project Page to Add Viewing Instructions.....	120
Take the Challenge to Enhance the Story (Optional).....	121
Exit Scratch.....	121
Coding Journal: Story Reflection.....	122
Session 4 Review: About Looks Blocks and Timing of Events.....	124
Session 4 Skill Review: Build an Interactive Diorama.....	126
Prepare to Build a Diorama.....	126
Build the Diorama.....	126
Answer Questions About the Diorama.....	127
Session 4 Extension Activity: Edit the Object's Costumes.....	128
Session 4 Extension Activity: Record a Sound Clip.....	129
Session 4 Extension Activity: Organize Scripts with Broadcast.....	130
Session 5 Engineer a Game.....	
Session 5 Engineer a Game.....	131
Session 5 Getting Started.....	132
Assignment 23 Develop a Game.....	139
Prepare to Build a Game.....	139
Plan the Game Design.....	139
Assignment 24 Determine Coding Blocks for Game Design.....	140
What Happens at the Start of the Game?.....	140
What Happens During the Game?.....	140
Keep Score and Track Time.....	141
What Happens at the End of the Game?.....	141
How Will You Direct the Timing of Events?.....	141
Assignment 25 Consult on Game Design.....	142
Assignment 26 Build and Test the Game.....	143

Create a New Project and Save It	143
Add a Backdrop for the Game Setting	143
Have the Player Explain How to Play the Game	143
Control When the Player Turns into the Mouse Pointer	144
Add a Target and Hide It	145
Control When the Target Appears	145
Tips to Improve Game Play	146
Take the Challenge (Optional).....	146
Exit Scratch.....	146
Assignment 27 Keep Score	147
Keep Track of the Score	147
Test the Scoring System.....	148
Set the Score to Zero When a New Game Begins.....	148
Get Creative or Exit Scratch.....	148
Assignment 28 Set a Time Limit.....	149
Create a Timer Variable	149
Test the Timer	150
Set the Timer to Zero When a New Game Begins.....	150
Fixing the Scoring Problem - Looking Ahead.....	150
Exit Scratch.....	150
Assignment 29 Game Over	151
Broadcast When the Game is Over	151
Tell the Player the Game Is Over and Hide the Target.....	152
Tips to Improve the Game.....	152
Exit Scratch.....	152
Assignment 30 Share the Game with Players	153
View the Game Project Page to Add Playing Instructions	154
Share the Project with the Scratch Community (Optional)	154
Exit Scratch.....	154
Coding Journal: Game Reflection	155
Session 5 Review: Operators, Conditions, and Variables	157
Session 5 Skill Review: Develop a Treasure Hunt	159
Prepare to Build a Treasure Hunt	159
Create the Scene.....	159
Introduce the Game.....	159
Keep Score	160
End the Game	160
Session 5 Extension Activity: Chat with a Sprite	161
Session 6 Curate an Activity Studio	
Session 6 Curate an Activity Studio.....	163
Session 6 Getting Started	164
Assignment 31 Build an Activity Studio	168
Plan to Share the Activity Studio.....	168
Open Scratch and Become a Scratcher	168
Build a Studio	168
Add Projects to the Activity Studio	169
Test the Studio.....	169
Share the Activity Studio.....	169
Exit Scratch.....	169
Assignment 32 Gain Player Feedback	170
Record Your Observations	170
Seek the Opinion of Others	170
Ideas to Improve the Activity Studio	171
Session 6 Review: Scratch Quiz	172
Session 6 Extension Activity: Remix a Scratch Project.....	174

Appendices

Appendices176

Appendix A: Assessment Tools177

 TechnoCode Skill Summary177

 Aquarium Marking Sheet181

 Maze Marking Sheet182

 Story Rubric183

 Game Marking Sheet184

Appendix B: Glossary185

Appendix C: Contact Information186



Introduction

This section provides valuable information about teaching TechnoCode. It includes a description of the Teacher Guide, as well as an overview of the course. In addition, there are ideas for implementation and technology integration.

For additional guidance, open the course in TechnoHub and select Get Started to access preparatory steps, resource list, and scheduling timetable.

[How to Use this Guide](#)

[TechnoCode Overview](#)

[Implementation and Technology Integration Ideas](#)

How to Use This Guide

This Teacher Guide contains the following:

Getting Started – this section contains a course description, as well as ideas for implementation.

Course Instructions: The course is comprised of six sessions, each focused on a problem-solving task that aligns with the project theme. Each session includes assignments that break down the task into manageable steps. The components of each session are as follows:

- Overview – an explanation of the session activities and their purpose.
- Materials – a list of handouts, sample files, templates, and teacher resource materials needed to teach the session.
- Teaching Strategies – instructional methods recommended for teaching the activities.
- Lesson Plan – a detailed list of each step in the session.
- Learning Objectives – a summary of the content knowledge and technical skills taught throughout the session. Content knowledge is information about the topic area. Students learn about a particular topic or subject area. A technical skill is the ability to use the computer to complete a given task. Students acquire knowledge of software tools and program features to use the computer effectively to solve a problem.
- Assignments – a session consists of assignments completed by students. Actions to be performed on the computer by the student are indicated with a triangle (▷). Background information and instructions are indicated with a dash (–).
- Review – a session review that contains a list of fill-in-the-blank, multiple choice, or short-answer questions intended to review both concept and technical knowledge (answers included).
- Skill Review – an additional assignment intended to review technical skills (includes completed sample).
- Extension Activity – an additional activity that relates to the problem-solving task presented in the session.

Appendices – this section contains additional information or materials including the following resources.

- Assessment Tools – skill summary and marking sheets for evaluation.
- Glossary – a definition of Scratch or coding terminology.
- Contact Information – how to contact TechnoKids Inc. for curriculum support.

TechnoCode Overview

Introduction to TechnoCode

Spark an interest in computer science! In this STEM project, students become coders that design a fun Activity Studio for kids using Scratch. Through discovery and exploration, they learn how to create a series of hands-on activities that children will enjoy playing.



The young programmers apply computational thinking to build algorithms that sequence commands, events, loops, and conditions. They learn how to construct scripts to develop animated scenes, mazes, interactive stories, and games. Additional challenges extend coding skills to create artwork, compose music, produce a diorama, and more! After each project, students complete coding journal logs to help them to think like a programmer.

Students complete the following tasks:

- In session 1, students are introduced to programming. They design animated scenes using Scratch. To start, they learn about the importance of computer programs and technology in daily life. By answering a series of questions, they begin to think like programmers. Afterwards, students study the Scratch interface to label the parts. Once familiar with the environment, they discover how to stack blocks of code together to form a script that makes a character talk. Once they have mastered some of the basics, they explore the Scratch libraries to make a scene of two friends having fun.
- In session 2, students create their first project for the Activity Studio. It is an animated aquarium. To start, they explore Scratch Motion blocks to discover how they can be used to make sprites move across the stage. Next, students use forever and if then blocks to control the fish swimming. Afterwards, they learn how to use the Paint Editor to design a fish tank that has a custom backdrop. To practice coding skills, a list of challenges provides a creative spark. Upon completion, the project is prepared for viewers. Students are then given the option to share the file with the Scratch community and classmates.
- In session 3, students create their second project for the Activity Studio. It is a maze game. This activity provides an opportunity for students to practice coding skills from Session 2 to solidify their learning. To start, they complete a planning sheet to organize their ideas. Next, they use Scratch to create a puzzle that has players help a character find a way to the end of a path using arrow keys. To make the project unique, a list of challenges helps to make a one-of-a-kind maze. Upon completion, the game is prepared for players. Students are then given the option to share the file with the Scratch community and classmates.
- In session 4, students create their third project for the Activity Studio. It is an animated story about a magical place. To start, they explore the Looks blocks to discover how they can be used to change the appearance of the main character and setting. Next, they enhance storytelling by triggering actions to occur when there is a switch in the backdrop. Afterwards, they direct the timing of events by sending messages to sprites using the Broadcast blocks. To practice coding skills, a list of challenges provides a creative spark. Upon completion the project is shared with viewers.

- In session 5, students create their final project for the Activity Studio. They apply their coding skills to develop a game. To start, they use planning sheets to determine the objective, scoring system, timing, and coding structure. Next, they discuss their design with a partner to assess if it is suitable for young children. Afterwards, students follow instructions to build and test the code. Challenges are included to foster originality. Upon completion, the project is prepared for players. Students are then given the option to share the file with the Scratch community and classmates.
- In session 6, students build an Activity Studio for kids. It will have a collection of Scratch projects including an animated scene, maze, story, and game. To gain player feedback a link to the studio will be shared. Based on observation and questioning, students will make recommendations upon how they can improve their Activity Studio.

Implementation and Technology Integration Ideas

TechnoCode introduces coding with Scratch to elementary and middle school students. It is an ideal course for Grades 4 and up. Jam-packed with programming activities, TechnoCode sparks an interest in computer science. Step by step instructions explain how to build animations, stories, games, art, music, and simulations.

Empower students with real-world skills they can use in the workplace. The instructional materials in TechnoCode encourage students to think like programmers. Resources include sample videos to inspire imaginations, planning sheets with guiding questions to help design scripts, assessment tools to evaluate student work, and coding journal logs to reflect upon learning.

Ideas for Implementation

The TechnoCode course has students create animated scenes, construct mazes, broadcast stories, engineer games, design artwork, compose music, build a diorama, and more! The activities are suitable for any teaching situation. Select the option that works best for you and your students:

- *Coding Unit with Elementary Students:* Assignments in Sessions 1-3 in TechnoCode are ideal for students new to Scratch. The activities are perfect for Grades 4 and up. Students design animations, create art, develop games, and compose music. The emphasis is on coding basics including how to build scripts, sequence commands, control action with *if then* conditions, and create simple loops. The activities focus upon directing movement, synchronizing sound, and understanding x and y coordinates.
- *Coding Unit with Middle School Students:* Once students understand the basics of coding in Sessions 1-3, they extend their learning in Sessions 4-6. The activities are ideal for students that understand the fundamentals and are ready for a challenge. The critical and computational thinking required is ideal for students in Grades 6-8. They produce a story, engineer a game, develop a treasure hunt, build a diorama, and remix a project. The emphasis is on having students manipulate the appearance of sprites, direct the timing of events with broadcasting, and create original artifacts using conditions, variables, and operators.
- *Computer Science Course:* TechnoCode has 32 assignments designed to ignite an interest in computer science. The focus is on thinking like a programmer. Each coding activity is divided into four parts – exploration, practice, freestyle, and reflection. Using a question-and-answer format, students discover the function of command blocks. Next, they follow guided instructions to build scripts. Afterwards, they apply their skills to complete open-ended challenges. Once a Scratch project is finished, students write a coding journal entry to reflect upon the experience.
- *Hour of Code:* If you only have one class to teach coding there are many assignments in TechnoCode that can be used for this purpose. If your students are beginners, they can develop simple animations. Assignment 5 targets how to build a script, Assignment 6 explores directing movement, and Assignment 17 focuses upon changing the appearance of a sprite. If your students have existing knowledge of Scratch, the skill reviews in Sessions 2-5 are excellent challenges.
- *Coding Workshop Series:* If you are running a workshop series as part of an after-school program or community event, then you will need to select assignments that fit the number of classes offered. Also, consider the age range and coding abilities of students.

Technology Integration Suggestions

The TechnoCode course is primarily a STEM project that teaches coding. However, the activities also integrate into other areas of curriculum including language arts, mathematics, social studies or science, visual arts, and music.

- *Computer Science*: TechnoCode is an introduction to programming. The activities have students build algorithms that sequence commands, events, loops, and conditions. Use the course to target computer science learning outcomes. The course includes a detailed list of skills achieved in each Session, ideal as a teacher checklist for assessment.
- *Language Arts*: The assignments in Session 1 and Session 4 can be integrated into curriculum as a language arts unit. In these assignments, students engage in visual storytelling. They create animated scenes and stories. To extend language arts learning outcomes, the concept of plot, setting, and characters is also applied when engineering games in Session 3 and 5.
- *Mathematics*: Integrate TechnoCode into an existing problem-solving unit in Math class. The assignments are an ideal fit because coding requires mathematical and logical thinking. For example, placing sprites on the stage requires plotting ordered pairs, rotating objects involves knowledge of angles, and setting the size of sprites uses percentages. As well, logic is used to control when or if an action happens.
- *Social Studies or Science*: Include the Session 4 Skill Review in TechnoCode as a creative way to showcase learning from another subject area. In this activity, students build an interactive diorama. It shows a scene from nature or a historical event that engages the viewer to click on objects to learn more. Complete the activity to have students share facts or create a simulation about a topic currently being studied. Samples provided include space exploration, tornado, and farming.
- *Visual Arts*: Target visual arts learning outcomes with TechnoCode. Graphic design is interwoven throughout the activities. Students apply their creativity to paint or edit unique sprites and backdrops. They also apply their skills to engage the audience using visual elements. In addition, the Session 2 Extension Activity specifically has students draw artwork with a pen using code.
- *Music*: Integrate TechnoCode into a music class. In the Session 3 Extension Activity, students invent an instrument. This activity is a fun way for students to express their musical talent.

This is a preview of the teacher guide.
Pages have been omitted.

SAMPLE



Session 3

Design a Maze

In this session, students create their second project for the Activity Studio. It is a maze game. This activity provides an opportunity for students to practice coding skills from Session 2 to solidify their learning. To start, they complete a planning sheet to organize their ideas. Next, they use Scratch to create a puzzle that has players help a character find a way to the end of a path using arrow keys. To make the project unique, a list of challenges helps to make a one-of-a-kind maze. Upon completion, the game is prepared for players. Students are then given the option to share the file with the Scratch community and classmates.

Assignment 12: Plan a Maze

Assignment 13: Move a Character using Arrow Keys

Assignment 14: Paint a Maze Backdrop

Assignment 15: Code a Maze

Assignment 16: Share the Maze Project with Players

Coding Journal: Maze Reflection

Session 3 Review: About Debugging

Session 3 Skill Review: Catch Me if You Can Game

Session 3 Extension Activity: Invent an Instrument

Session 3 Getting Started

Overview

In this session, students create their second project for the Activity Studio. It is a maze game. This activity provides an opportunity for students to practice coding skills from Session 2 to solidify their learning. To start, they complete a planning sheet to organize their ideas. Next, they use Scratch to create a puzzle that has players help a character find a way to the end of a path using arrow keys. To make the project unique, a list of challenges helps to make a one-of-a-kind maze. Upon completion, the game is prepared for players. Students are then given the option to share the file with the Scratch community and classmates.

Materials

- Scratch Offline Editor or Internet access to Scratch <https://scratch.mit.edu/>.
- Scratch Flashcards (optional)
- Assessment:
 - Maze Checklist
 - Maze Marking Sheet
 - Coding Journal: Maze Reflection
- Sample files:
 - *maze* video file
- Session 3 Review
- Session 3 Skill Review
 - *catch* and *catch 2* video files
- Session 3 Extension Activity
 - *music* and *music 2* video file

Teacher Preparation

(Refer to the *Preparing to Teach* section of this guide for instructions)

- Make the files in the Code folder available to students.
- View the sample video *maze* to gain an understanding of the completed project.

Teaching Strategy

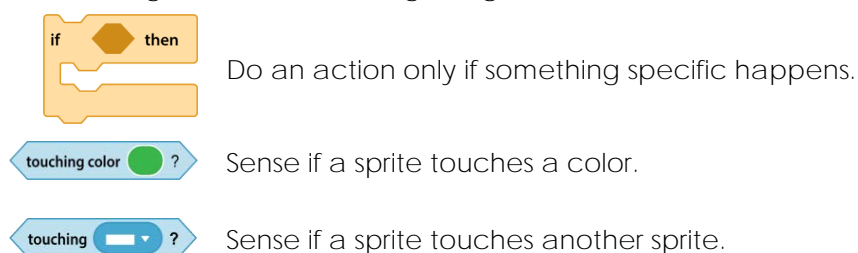
In this session, students design a maze game for kids. Explain scenario to students:

In this session, you will create another fun activity for kids. It will be a game where players use arrow keys to move a character through a maze. Apply your coding knowledge to build a puzzle that is fun for young children to solve.

Assignment 12: Plan a Maze

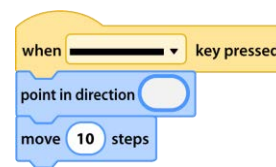
In this assignment, students prepare to create a maze for kids. The game will have players use the arrow keys to help a character find their way through a path. Before starting, have students watch the maze video to gain an understanding of the task. Afterwards, they answer guiding questions to organize design ideas and develop an understanding of the coding blocks required to build the scripts.

The purpose of the maze is to have students apply their knowledge of Scratch in a new way. This will provide an opportunity to solidify their understanding of programming before learning more coding skills in the following session. Prior to beginning, discuss the *if then* block:



Assignment 13: Move a Character using Arrow Keys

In this assignment, students design a maze game based on the plan they developed in the previous assignment. To start, they build the script to control how the character moves when the arrow keys are pressed. Prior to beginning the assignment demonstrate how the coding blocks work in the script:



- Move Up: Walk up an aisle. Explain that this is how the character will move when the up arrow is pressed. Since a character moves in the direction it is pointing the *point in direction* block must be set. What value will make a character move up? 0
- Move Down: Walk down an aisle. Explain that this is how the character will move when the down arrow is pressed. What value does the *point in direction* block need to move a character down? 180
- Move Right: Walk to the right across an aisle. Explain that this is how the character will move when the right arrow is pressed. What value does the *point in direction* block need to move a character right? 90
- Move Left: Walk to the left across an aisle. Explain that this is how the character will move when the left arrow is pressed. What value does the *point in direction* block need to move a character left? -90

Assignment 14: Paint a Maze Backdrop

In this assignment, students apply their painting skills to make a backdrop that has connected paths. The path must be large enough to fit the character. If it is too narrow, the character will get stuck and will be unable to move. Emphasize the need for the path to be surrounded by a solid color.

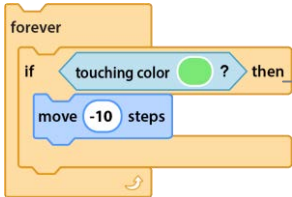
To see the code 'inside' a sample project, follow the instructions in *Preparing to Teach* step 7. Load and view the *maze.sb3* sample from the *Scratch Files* folder.

Assignment 15: Code a Maze

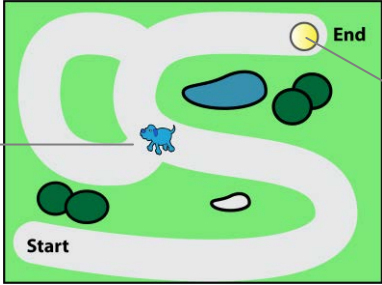
In this assignment, students complete the maze. They write code to keep the character inside the maze using an *if then* block that senses when the background color is touched. To complete the game, an object is added at the end of the path that displays a message when touched by the character.

Examine the script and discuss the coding blocks:


Stay on the path:



If the character touches the background color, then it moves back 10 steps.



Display a message:



If the object touches the character, then it shows a message.

Pose the following questions to develop understanding:

- *What is the purpose of the forever block?* It causes Scratch to keep checking if the condition is true.
- *Would the scripts work without the forever block?* No. Scratch would only check once and then never again.
- *Would the script work if the wrong color was selected?* No. The color must be the backdrop color.
- *Why use -10 steps instead of 10 steps?* The negative sign moves a sprite back. It can't move forward or it would be outside of the maze, instead of inside of it.
- *Will the message "game over" display if the character does not touch the object?* No. it will only show if they touch.

Need more practice? Have students complete *Session 3 Skill Review Catch Me If You Can*. In this review they design a fun game using *if then* blocks.

Assignment 16: Share the Maze Project with Players

In this assignment, students complete their maze. Using a checklist, they review the content and coding to make sure that all the parts are finished. When it is done, they add viewer instructions and notes to the project page.

A digital copy of the *Maze Checklist* and *Maze Marking Sheet* are available if you would like to edit the assessment tools.

Want a fun challenge? Have students complete *Session 3 Extension Activity Invent an Instrument*. In this activity, they explore the Sound Editor tools and Music blocks to create an instrument that plays sounds when buttons and keys are touched by the mouse pointer.

Lesson Plan

Assignment 12: Plan a Maze

- Plan the purpose of the maze game. Answer questions to select the audience, sprite character, goal, instructions, path color, backdrop color, and sprite object.
- Think like a programmer to plan the coding required to build the game. Answer questions to determine the blocks needed to make the character move, keep the character inside the path, and display a message when the character reaches the end of the maze.

Assignment 13: Move a Character using Arrow Keys

- Open Scratch and create a new project.
- Add a sprite and rename it *character*. Delete *Sprite1*.
- Build scripts that control the movement using the arrow keys. Duplicate a script to make the task faster to complete. Organize the scripts in the Code Area.
- Reduce the size of the character to fit inside the maze.
- Save the project as *maze* and exit Scratch.

Assignment 14: Paint a Maze Backdrop

- Open the saved *maze* project in Scratch.
- Apply skills to paint a maze backdrop that includes a background color, path, start point, and finish point. Use the paint tips to produce a unique design.
- Exit Scratch.

Assignment 15: Code a Maze

- Open the saved *maze* project in Scratch.
- Build a script that is triggered by the Go button.
- Set the start point of the character.
- Use an *if then* condition to keep the character inside the path. Use coding blocks to make the character move back ten steps if it touches the backdrop color.
- Apply skills to have a sprite object at the end of the path display a message when touched by the character.
- Select a challenge to practice coding skills and create a unique maze that young children will enjoy playing.
- Exit Scratch.

Assignment 16: Share the Maze Project with Players

- Open the saved *maze* project in Scratch.
- Use a checklist to review the parts of the project to make sure they are complete.
- Based on the assessment, edit the project to improve the code or maze design.
- Access the project page to add viewing instructions and notes.
- (Optional) Share the project with the Scratch Community. Play classmates' mazes.

Learning Objectives

Computer Science

- apply computational thinking
- compare and refine coding blocks to determine their function and select the most appropriate value to complete a task
- edit blocks to improve a script and produce a unique outcome
- create an original idea for a maze that kids will enjoy playing
- develop a plan for a maze that outlines the character, goals, and sequence of events
- consider the needs, wants, abilities, and perspectives of potential users of the maze
- decompose, or break down, the parts of a maze to determine coding blocks
- recognize patterns to build looping sequences and conditions for a maze
- build code to design a maze that is controlled using arrow keys
- run a script or program
- debug errors to find and fix a mistake in a script
- test and refine scripts
- review and edit a project using a checklist
- reflect upon program development using a coding journal to describe steps, explain choices, express feelings, and analyze learning

Coding with Scratch

Manage Projects

- create a project
- save a project
- modify an existing project to add more features
- view a project page
- provide player instructions for a project
- add notes to a project

Working with the Scratch Interface

- drag blocks from the Blocks palette onto the Code Area
- select program options using tools, tabs, menus, or a toolbar
- stack blocks in a sequence to build a script that performs a task
- remove a stack of blocks from the Code Area
- undo an action
- duplicate a script or stack of blocks
- arrange scripts in the Code Area
- create a game using the stage as a canvas

Add Characters and Objects to the Stage

- select a sprite from a Library
- position sprites on the stage using x and y coordinates
- delete a sprite from the sprite pane
- view information about a sprite
- rename a sprite to make it easy to identify
- adjust the size properties
- use multiple sprites to create a game

Apply a Background to the Stage

- create a unique backdrop using paint tools

Sequence Sound with Action

- select an audio clip from a Library
- make a sprite start to play a sound
- increase or decrease the speed of a clip (optional)
- echo a clip to have the sound bounce back (optional)
- apply a robot effect to make the clip sound mechanical (optional)

- increase or decrease the volume of a clip (optional)
- reverse the direction of the clip (optional)
- pick instrument sounds using the music extension blocks (optional)
- set the beats for the instrument (optional)

Trigger a Script

- begin a script with an event block
- trigger a script to run when the Go button is clicked
- trigger a script to run when a keyboard key is pressed

Cause Movement on the Stage

- adjust the value in a block to control movement
- move a sprite forward by a number of steps across the stage
- tell a sprite to go to a specific point on the stage
- set the direction a sprite faces by adjusting it down, up, left, and right using angles
- adjust the rotation style of a sprite to make it move left to right, all around, or not turn
- move a sprite to the mouse pointer (optional)

Modify the Appearance of Characters and Objects on the Stage

- make a sprite display a message to show that a game is over
- customize the text in a block to convey a message to the viewer

Control Action with Conditions

- wait for an amount of time before the next action happens
- loop a sequence forever
- trigger an action if a condition is true

Set Conditions with Sensors

- perform an action when a sprite touches a specific color or another sprite

Graphic Design

- draw shapes such as a rectangle or circle
- set the fill or line color of an object
- adjust the saturation and brightness to customize a color
- paint freestyle lines using a brush to create a maze path
- set the width of a line
- reshape points to bend or move a line segment
- magnify the canvas to zoom in or out
- select an object on the canvas
- add text to the canvas
- format the font of text
- size or rotate a text box
- undo an action
- copy a color
- create a backdrop for a maze
- copy and paste a selection (optional)

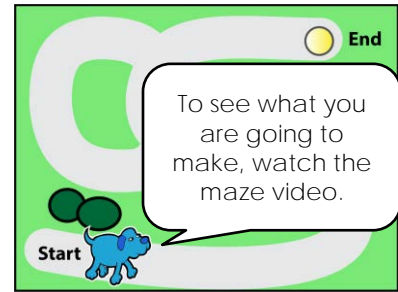
Digital Citizenship (optional)

- sign in and out of a web-based app appropriately
- share a Scratch project with the Scratch community
- view or play another Scratcher's project
- recognize the accomplishments of a peer by posting on a Scratch project
- work respectfully and responsibly with others online

Assignment 12 Plan a Maze

You are going to design a maze for young kids. A maze is a bunch of paths that connect. To play the game, a player must find the correct way to get from start to end.

Look through the Scratch Sprite Library to get ideas.



Plan the Purpose of the Maze

1. The maze is for young kids. Who do you know that would like to play a simple game?

2. Pick a character. What or who needs to find their way through the maze?

3. Every game needs a goal. Why does the character need to get to the end of the path?

- find a lost item
- eat favorite food
- my idea:
- get a treasure
- go home

4. Keyboard arrow keys will be used to move the character. Write player instructions.

Use the arrow keys to

5. What item could be placed at the end of the maze that the character must reach?

Think Like a Programmer to Plan the Maze

6. A player will move the character using arrow keys. A sprite moves in the direction it points. Pick the correct value for each *point in direction* block to set the arrow key.

when **up arrow** key pressed

point in direction

move 10 steps

a. 0

b. 180

when **left arrow** key pressed

point in direction

move 10 steps

a. 0

b. -90

when **down arrow** key pressed

point in direction

move 10 steps

a. 180

b. 90

when **right arrow** key pressed

point in direction

move 10 steps

a. 180

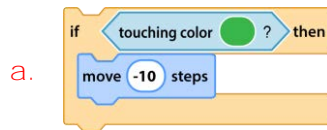
b. 90

7. The game will start when the Go button is clicked. The character must always start in the same spot. Which block should be used?



8. The character must stay in the maze. If it touches the background color that is around the path, then the character will move back 10 steps.

Which stack of blocks has the correct Sensing block?



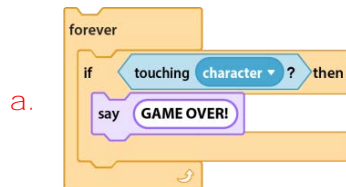
9. The computer must keep checking to make sure the sprite character is in the maze.

What control must go around the *if then* block to make the script run non-stop?



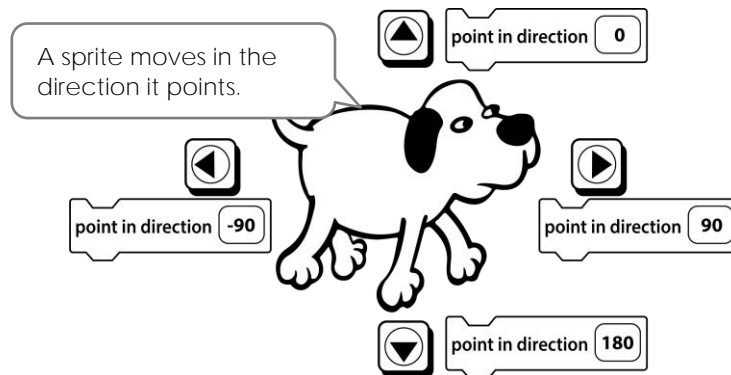
10. The game is over when the character reaches the end of the maze and touches an object. At that time a message will display, such as *GAME OVER* or *You did it!*

Which script should be added to a sprite object to cause this to happen?



Assignment 13 Move a Character Using Arrow Keys


In this assignment, you begin to design your maze. Build scripts to move the character up, down, left, and right using the arrow keys.

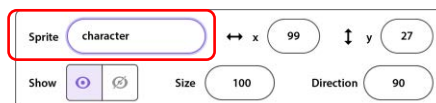


Create a New Project and Save It

- ▷ Sign into *Scratch* and click *Create*.
- ▷ Save the project as *maze*.

Add a Sprite and Rename It Character

- ▷ Find a character for your maze. 
- ▷ Select the sprite. In the Sprite box, type *character*.

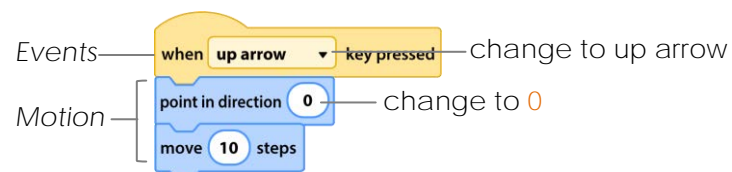


- ▷ Delete *Sprite1*.



Make the Character Move Using the Up Arrow Key

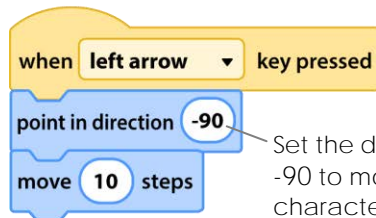
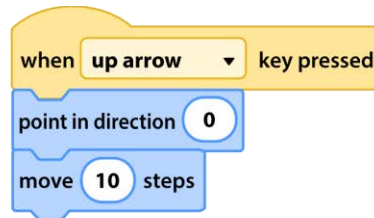
- ▷ Use your skills to build this script. Use the tips to find and edit each block:



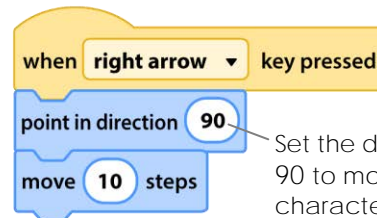
- ▷ Press the *UP* arrow key on the keyboard to move the character.

Duplicate a Script to Make the Character Move Using Arrow Keys

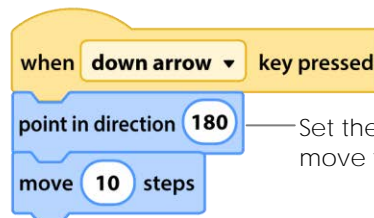
- ▷ Right click on the script. Select *Duplicate*.
- ▷ Edit the script to move left:
 - Change *up arrow* to left arrow.
 - Change *point in direction 0* to *-90*.
- ▷ Repeat the steps to create scripts for the right arrow and down arrow:



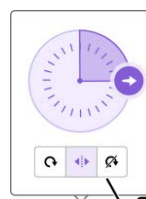
Set the direction to -90 to move the character left.



Set the direction to 90 to move the character right.



Set the direction to 180 to move the character down.



If the character does not face in the correct direction, adjust the rotation style.

- ▷ Press the arrow keys on the keyboard to move the character.

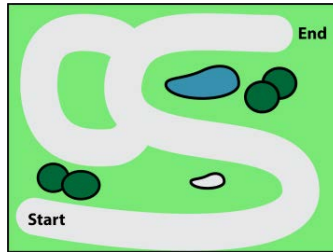


Drag the scripts to organize them in the Script Area.

Exit Scratch

Assignment 14 Paint a Maze Backdrop

Use the Paint Editor to create a maze.

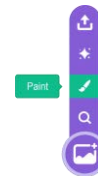


What type of maze will you make? Refer to *Assignment 12* to review your plan.


- road
- path
- tunnel

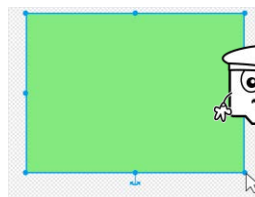
Open the Paint Editor to Create a Maze

- ▷ Open the saved *maze* project in Scratch.
- ▷ Hover over *Choose a Backdrop*. From the menu, click *Paint*.






Fill the Maze with Color




- ▷ Select *Rectangle*. 
- ▷ Click and drag to draw a rectangle. Drag the handles to make it fit the canvas.



The background color will be used to keep the sprite character in the maze. If the character touches the color, then it moves back 10 steps.

- ▷ Fill the canvas with a background color. 
- ▷ Click *Outline*.  Pick *No color*. 


Draw the Path Using a Brush

- ▷ Select *Brush*. 
- ▷ Pick a fill color. 
- ▷ Select a line width.  100
- ▷ Click and drag to paint a path. For example:



The path must be wide enough for the character to move inside.

Add Start and Finish to the Maze

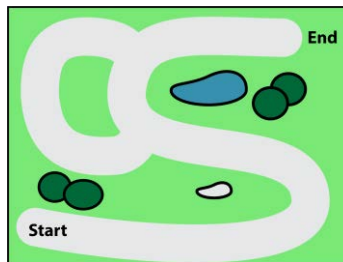
- ▷ Select *Text*. **T**
- ▷ Use your skills to pick a *Fill* color. 
- ▷ Click *font* to pick an option from the menu.




- ▷ Click on the canvas to make a text box. Type *Start*. Place it near the beginning of the maze.
- ▷ Use your skills to add *End*, *Finish*, *Home* or your own idea. Place it near the end of the maze.

Complete the Maze

- ▷ Add details to the maze. Keep a solid color around the maze path.





Design Tips:

- Add details such as trees, rocks, or ponds.
- Use *Undo*  to remove an action.

Size the Character to Fit Inside the Path

You need to resize the character so that it can fit inside the path. The original size is 100%. To make the sprite smaller the size must be less than 100.

- ▷ Click the *Code* tab.   
- ▷ Select the character sprite.
- ▷ Type a *number* into the *Size* box. Press ENTER on the keyboard.



Does the character fit within the edges of the path?

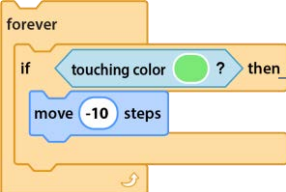


Exit Scratch

Assignment 15 Code a Maze


In this assignment, you will complete the maze using *if-then* logic. Build scripts that will keep the character inside the path. When it reaches the end, an object should display the message *Game Over*.

Stay on the path:

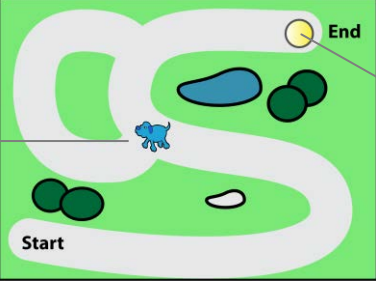


If the character touches the background color, then it moves back 10 steps.

Display a message:



If the object touches the character, then it shows a message.



Place the Character at the Start of the Maze

The game will begin when the *Go* button is clicked. The character must begin at the start of the maze. It should point in the correct direction.

- ▷ Open the saved *maze* project in Scratch.
- ▷ Drag the character to the start of the maze.
- ▷ Use your skills to build this script. Use the tips to find and edit each block:

Events

when clicked

Motion

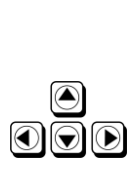
go to x: y:

Where do you want it to start?

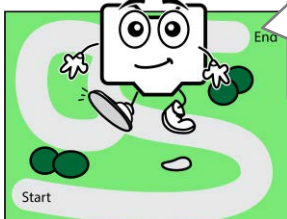
point in direction

Which direction should it face?

- ▷ Click *Go*
- Press the arrow keys on the keyboard to move the character.



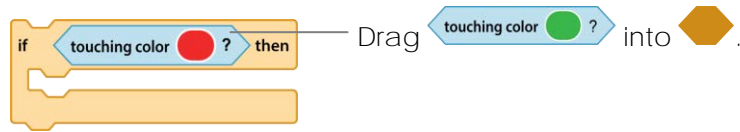
The sprite will move anywhere on the stage. You need to add a condition to keep it inside the path.



- ▷ Click *Stop*.
- ▷ Click *Go*. *Is the character in the correct start position?*

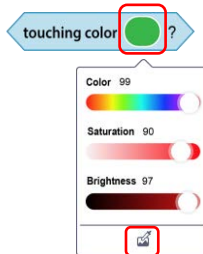
Add a Condition to Keep the Character Inside the Path

- ▶ Add the *if then* and *touching color* blocks.

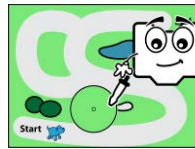


- ▶ Set the color to the background:

- Click the color box.



- Click *Pick Color* at the bottom of box.
 - Select the background on the stage.



Around the maze is a *background color*. If the sprite touches it, an action will happen to keep the sprite inside the path.

- ▶ What happens when the character touches the color? It moves back.

Add *move -10 steps*:



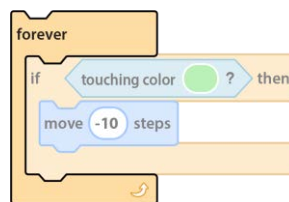
Test the Maze to Debug the Problem

- ▶ Click *Go* (green flag). Move the character. Click *Stop* (red octagon).

The character can still move anywhere. What is wrong?
The *if then* blocks only runs once. The game must keep sensing if the sprite touches the background color. You need to add a *forever* block.



- ▶ Use your skills to add a *forever* block.




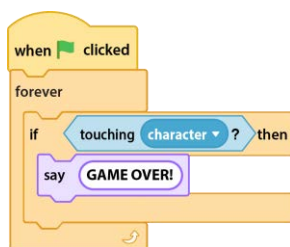
TIP: If the character does not move, you may need to make it smaller to fit inside the maze.

- ▶ Play the game again. (green flag) The character should stay inside the path.

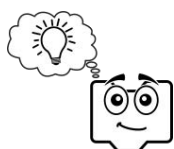
Build a Script to End the Game

The game is over when the character reaches the end of the maze and touches an object. What happens then?

- ▷ Use your skills to add an object to the end of the maze. 
- ▷ Select the *object*.
- ▷ Build the script to end the game. For example:




Take the Challenge



Take the challenge!
Apply what you have learned to
make your maze one-of-a-kind.

Pick one or more challenges:

- Add another sprite to decorate the maze.
- Set the rotation style  for *each* arrow key to control how the character moves. TIP: Set the style to *all round* for up and down.
- Play a sound when the sprite object is touched. You may need to add a *wait* block.

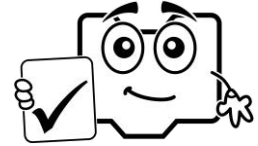
Exit Scratch

Assignment 16 Share the Maze Project with Players

In this assignment, you get ready to share your second activity with players. The maze will be fun for children to solve. Complete the checklist to make sure all the parts of the project are finished. If you missed anything, go back and edit the *maze* project.


Maze Checklist

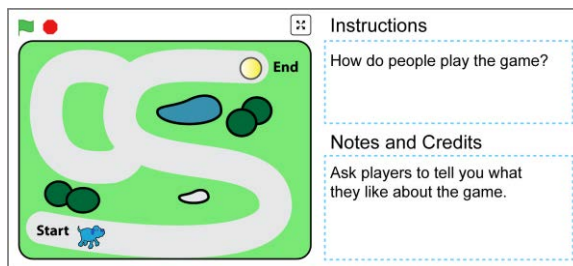
- ▷ Open the saved *maze* project in Scratch.
- ▷ Use the checklist to review the project.
Do you need to make changes?

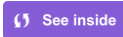


	✓
Design	
The backdrop is a painted maze.	
A solid background color goes around the path.	
<i>Start</i> and <i>Finish</i> labels show the beginning and ending points of the maze.	
The <i>character</i> and <i>object</i> sprites suit the purpose of the game.	
The <i>character</i> sprite has been resized to fit inside the maze path.	
An object is at the <i>End</i> point of the maze.	
Coding	
The <i>character</i> sprite moves up, down, left, and right using the arrow keys.	
The <i>character</i> sprite always begins at the <i>Start</i> point.	
If the <i>character</i> sprite touches the background color, it moves back 10 steps.	
When the <i>character</i> sprite touches the object at the end of the maze, a message tells the player the game is over.	
Additional coding has been added to the maze to make it unique. For example, it may have other sprites or a sound that plays.	


View the Maze Project Page to Add Playing Instructions

- ▷ Click *See Project Page*. 
- ▷ Add instructions:
 - In the Instructions box explain how to play the maze.
For example, *Use the arrow keys to help the character reach the end of the maze.*
 - In the *Notes and Credits* box invite players to add a comment. For example, *This is my first Scratch game. I made it for kids to play. Let me know what you like about it.*



- ▷ If you do not want to share your project, click *See inside*. 

Share the Project with the Scratch Community (Optional)

- ▷ If you want others to play your maze, click *Share*.
- ▷ Click *Copy Link*. 
- ▷ Share the link with others.

Play Maze Games (Optional)

- ▷ Play a friend's maze.
- ▷ If available, in the Comments area, type *one thing you liked about the game*. Click *Post*.



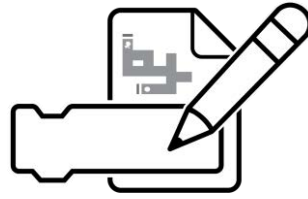
Read Comments (Optional)

- ▷ View your *maze* project page. Read the comments below the project.

Exit Scratch

Coding Journal: Maze Reflection

Answer the questions to write a coding journal entry about the *maze* project.



1. The maze is for young kids. What is one thing you think they will like about the game?

2. How could you make the maze more difficult for older users?

3. There were several *challenges* that you could complete to make a one-of-a-kind maze. What extra items, or code, did you include in your game? Why do you think it made it better?

4. You have learned how to code a game that uses arrow keys to move the character. What other game could you make that uses this type of player control?

5. Debugging is when you need to find and fix problems or *bugs* in the code. Describe a problem you had with your code. What was the *bug* and how did you fix it?

6. When you are learning something new you can feel a mix of emotions. Sometimes you are excited, but other times you can be frustrated. Often you can feel many things at the same time! Circle two words that describe how you felt while making the maze.

excitement

interest

frustration

surprise

pride

confusion

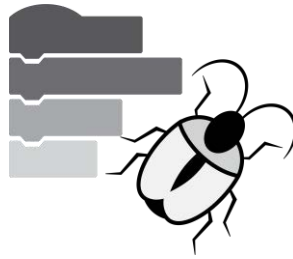
other:

7. Pick one emotion from Question 6. Describe why you felt that way.

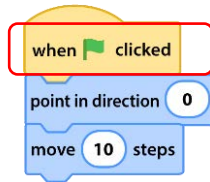
8. You can share a project on the Scratch community. This lets others view or play your creations. Do you like being able to share your work with others? Why or why not?

Session 3 Review: Debug the Script

Debugging is when you need to find and fix problems or *bugs* in the code. Think like a programmer to answer the questions.



1. The script needs to start when the up arrow is pressed. Circle the block in the script that is incorrect.



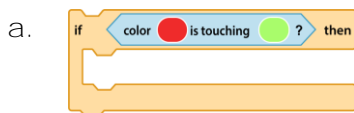
2. Select the correct block to fix the script.



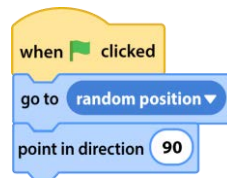
3. The character should move 20 steps when it touches the color red. Circle the block that is incorrect in the script.



4. Select the correct block to fix the script.



5. The sprite needs to start in a specific spot. Find the mistake. How do you fix it?



The *go to random position* is the wrong block. Instead, it should be a *go to* block that includes x and y coordinates that specify a point on the stage.

/2

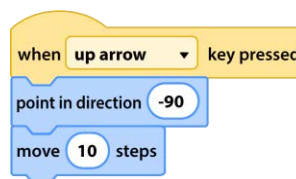
6. The stack of blocks should run non-stop. What block needs to be added to the code?



forever block

/2

7. The script should move the sprite up when the up arrow is pressed. Find the mistake. How do you fix it?



The *point in direction* value needs to change to 0.

/2

TOTAL: /10

Session 3 Skill Review: Catch Me If You Can Game

Design a game.


Can the character catch the item before it moves?

Need ideas? Help the...

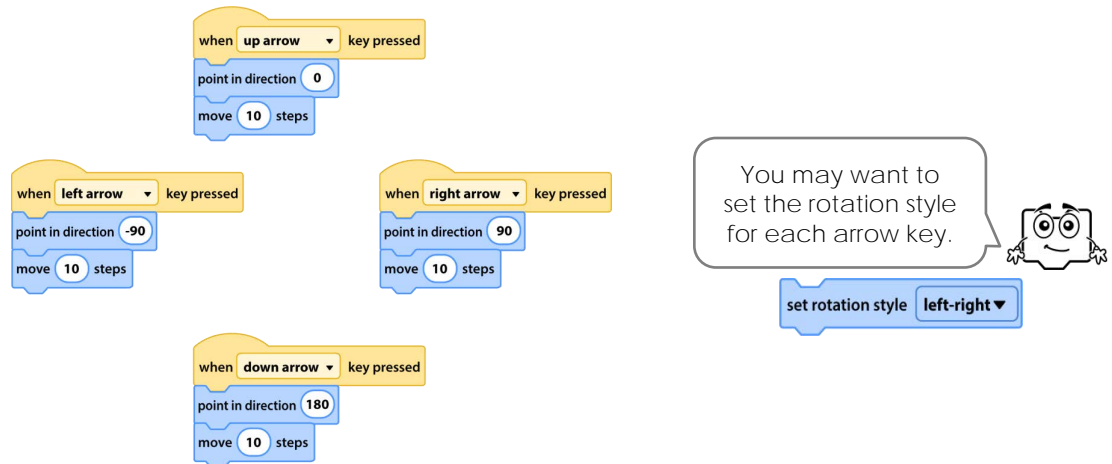
- knight find the disappearing dragon
- girl catch the butterfly
- shark eat dinner




Build the Game

1. Open Scratch and start a new project. Save it as **catch**.
2. Insert a *catcher*:
 - a. Insert a character that must catch an item. 
 - b. Rename the sprite **catcher**. Resize it.
 - c. Delete *Sprite1*.

3. Build the scripts to move the *catcher* using arrow keys:



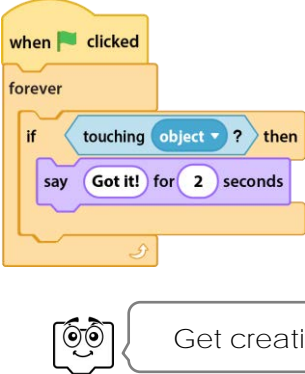
4. Insert an *object*:
 - a. Insert an item the character is trying to catch. 
 - b. Rename the sprite **object**. Resize it.
5. Build a script to have the *object* show up in random places for a few seconds. How long should it stay in one spot?



What happens when the object appears?

- play a sound
- turn back and forth
- move 20 steps




6. Build a script that displays a message when the *catcher* touches the object. What should the character say?



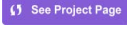
The image shows a Scratch script starting with a 'when clicked' block, followed by a 'forever' loop. Inside the loop is an 'if touching object?' block with a 'then' branch containing a 'say Got it! for 2 seconds' block. Below the script is a character with a speech bubble that says 'Get creative!'.

What happens when the character catches the item?

- go to a specific location
- display the next costume

- Set where the character starts the game 
- Play a sound when the game starts. 
- Insert or paint a backdrop 

7. Add playing instructions:

- Click *See project page*. 
- In the *Instructions* box explain how to play the catch game. For example, *Use the arrow keys to help the character catch the object.*
- In the *Notes and Credits* box describe the age range of the game. For example, *This game is for children ages 3-5.*

8. (Optional) Share the game. Play classmates games.

Answer Questions About the Catch Game

1. Why will young children like playing the game?

2. Pick two things that will make the game more interesting for older children:

- 30 second timer
- score points each time catch item
- background music
- obstacle that if touched holds character in place for a few seconds
- your own idea:

3. Could the game work without a *forever* block? Why or why not?

No. The forever block makes it so that the object shows up in different spots over and over again. The forever block also makes the character say something each time an object is touched. Without it, the character would only do it once.

Session 3 Extension Activity: Invent an Instrument

Invent an instrument and use it to compose music.

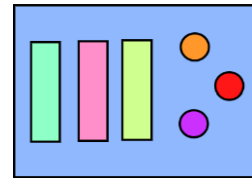
It will have at least three buttons and three keys. Each one will be a different color. When the musician moves the mouse pointer over each color, a sound will play.



To see what you are going to make watch the music video.

Paint the Instrument

1.
 - a. Open Scratch and start a new project. Save it as **music**.
 - b. Delete *Sprite1*.
 - c. Hover over *Choose a Backdrop*. Click *Paint*.
 - d. Draw an instrument with at least three buttons and three keys:

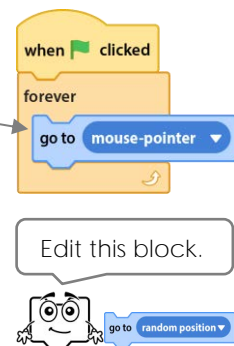


Design Tips:



- Draw the instrument outline with the *Brush*. Fill with color.
- Copy and paste buttons or keys to make them quickly.
- The instrument should fill the canvas.
- Each button and key must be a different color.

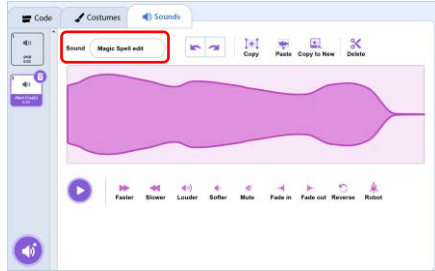
Turn the Mouse Pointer into a Sprite

2.
 - a. Insert a simple symbol or shape.
 - b. Make the sprite tiny so it will fit over a button or key. Size **40**
 - c. Click the *Code* tab.
 - d. Build a script to make the sprite move with the mouse-pointer:
 - e. Click *Go*.
 - Is the mouse pointer the sprite?
 - Does the mouse pointer fit over the buttons and keys?





Edit Three Sound Clips to Make Them Unique

3. a. Click the **Sounds** tab. 
- b. Click *Choose a Sound*.  Pick a short clip.
- c. Type **edit** beside the sound name.



Tips for Picking Sounds:

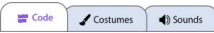
- Pick a clip with a single beat.
- Select sounds from the *Effects*, *Notes*, or *Percussion* categories.
- Trim  a sound to shorten it.

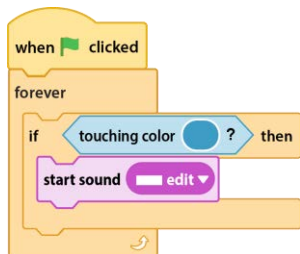
- d. Edit the clip using the Sound Editor tools:
 - Click the tool many times to increase the effect.
 - Click *Undo*  many times to remove effects.





- e. Apply your skills to insert and edit sound clips for each button. 


Code Each Button to Play a Sound Clip

4. a. Click the **Code** tab. 
- b. Build the script:




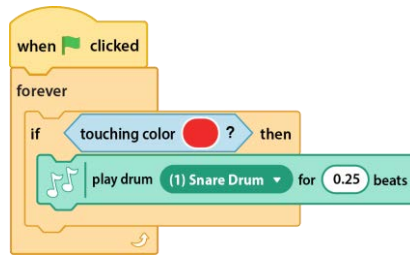
Coding Tips:

- Make a script quickly. Right click and select *Duplicate*. Edit the color and sound.
- Copy the color of the button. 
- If you cannot copy the color, make sure the code is not running. Click *Stop*. 
- Use *start sound* instead of *play sound* to have many sounds play at the same time.


- c. Click **Go** .
 - Does the sound play when the mouse pointer touches the button?
- d. Use your skills to build scripts for the remaining buttons.


Code Each Key to Play an Instrumental Sound

5.
 - a. Click *Add Extension*.  Select *Music*.
 - b. Build the script:

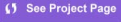


Coding Tips:

- Copy the color of the key. 
- Pick from over 15 instruments.
- Adjust the beat. 1 beat is very slow, 0.75 is slow, 0.5 is medium, 0.25 is fast, and 0.15 is very fast.

- c. Click Go .
 - Does the sound play when the mouse pointer touches the key?
 - d. Use your skills to build scripts for the remaining keys.
-

Prepare the Project Page

6.
 - a. Click the *project page*. 
 - b. In the *Instructions* box explain how to play the instrument. For example, *Move the mouse over the buttons and keys to make music.*
 - c. In the *Notes and Credits* box, give your invention *a name*.
 7. (optional) Share the instrument with others.
-

Answer Questions about the Instrument

1. What is the name of your invention?
2. Why will children like playing the instrument?
3. A sprite was turned into a mouse pointer. What sprite did you pick? Why?

This is a preview of the teacher guide.
Pages have been omitted.

SAMPLE