

Find the Gold

1. Synopsis

In this unit, students create a maze game app. The maze app will feature a Ball sprite that moves through the maze based on the user interaction of tilting the smartphone or tablet. Students will learn how to use the Canvas, Ball, and ImageSprite components in App Inventor to create an animated game. Students will also learn to use the Accelerometer sensor to guide the Ball's movement. Collision detection between Ball and ImageSprite components will help create an interactive game. Students will collaborate with each other to implement new features in the app.

2. Learning Objectives

After completing this unit, students will be able to:

1. Demonstrate how to use Canvas, Ball, and ImageSprite components in App Inventor to create an interesting game app.
2. Demonstrate understanding of placement of ImageSprite and Ball components based on an X,Y coordinate system.
3. Use the Accelerometer Sensor component to navigate a ball through the maze.
4. Apply the Computational Thinking practices of being incremental and iterative, and testing and debugging.
5. Demonstrate understanding of the use of the Notifier component in an app.
6. Use conditionals correctly in a program.
7. Work collaboratively to design, develop, and test new features in an app.

3. Mapping with the CSTA Standards

This table shows the alignment of this unit with the intended learning outcomes to the CSTA CS Standards:

2-CS-02	Design projects that combine hardware and software components to collect and exchange data. [C] CS: Hardware & Software [P] Creating (5.1)	The Accelerometer sensor is used in the maze game.
2-AP-10	Use flowcharts and/or pseudocode to address complex problems as algorithms. [C] AP: Algorithms [P] Abstraction (4.4, 4.1)	Students design and implement new features with planning tools.
2-AP-13	Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. [C] AP: Modularity [P] Computational Problems (3.2)	Students design new features, decomposing into different parts - UI components, and actions for those components.
2-AP-14	Create procedures with parameters to organize code and make it easier to reuse. [C] AP: Modularity [P] Abstraction (4.1, 4.3)	Students use SetUpMaze procedure to position maze components.
2-AP-17	Incorporate existing code, media, and libraries into original programs, and give attribution. [C] AP: Program Development [P] Abstraction (4.2), Creating (5.2), Communicating (7.3)	Students finish partially coded procedure and use library sprites from template.

4. Learning Prerequisites

Students should have experience with App Inventor, and be comfortable in the Designer and using the Blocks Editor to code a basic app.

5. Lesson Plan (45 minutes x 6)

This unit consists of six 45 minute lessons.

The Lesson Plan is an abbreviated outline. A longer, more detailed Teacher Guide can be found in the Appendix.

Lesson 1

Time	Activity
15 min	Introduction to Unit <ol style="list-style-type: none">1. Ask students if they have ever played games on their tablets or computers. Is it fun? Ask for examples of favorite apps.2. Explain that they will make a game app called “FindTheGold” in this unit. It works as follows: players navigate a ball through a maze on the mobile device by tilting the device.3. Demonstrate the finished app.4. Students will work in pairs on the project. Introduce pair programming (https://youtu.be/vgkahOzFH2Q). Divide students into their teams.

15 min	Demonstration of Canvas, Ball, and ImageSprite Components in App Inventor <ol style="list-style-type: none"> 1. Canvas is your background for sprites to appears and move. 2. Balls/ImageSprites are the elements on the canvas that can be controlled by user interaction and by coding. 3. Placement of Balls/ImageSprites is based on the Cartesian coordinate system, using X,Y coordinates. 4. Explain Resolution (number of pixels) on a screen and how different phones/tablets have different resolutions. Explain placement and dimensions of sprites should be made using percentages to account for different resolutions.
15 min	Coding Activity <ol style="list-style-type: none"> 1. Ask students to complete the missing blocks in the simplified flowchart in the Student Guide. 2. Using FindtheGold_template.aia, ask students to start coding the app following Student Guide Part 1. <ol style="list-style-type: none"> a. Ask students to discuss with their classmates the function of each component in the Designer (page 2 of the Student Guide). b. Add code blocks to the SetUpMaze procedure to place the remaining Wall ImageSprites on the screen. c. Test the app to make sure the maze fits correctly on the mobile device.

Lesson 2

5 min	Introduction and Review <ol style="list-style-type: none"> 1. Review the basic Find the Gold app and new components used. 2. Let students know that they will use the AccelerometerSensor to control the Ball movement.
15 min	Introduction to the Accelerometer Explain to students how the Accelerometer Sensor works and how they will update the X,Y position of the Ball based on the X and Y acceleration.

25 min	Add Ball Movement to App Students follow <i>Find the Gold Student Guide: Part 2</i> to implement the Ball movement based on the Accelerometer Sensor.
--------	---

Lesson 3

5 min	Introduction and Review <ol style="list-style-type: none"> Review the basic Find the Gold app and new components used. Let students know that they will add to the app by testing for collision, and adding the Notifier component to let the player know when they win the game.
10 min	Conditionals Explain to students the use of if and if-then-else blocks as a way to direct the flow of a program.
5 min	Notifier Show Notifier component and its blocks. Explain its purpose in an app.
25 min	Add Testing for Collision and Notifier to App Students follow <i>Find the Gold Student Guide: Part 2</i> to implement the following: <ol style="list-style-type: none"> Collision detection with walls and gold. Notify user that game is over and ask if they want to play again or quit.

Lesson 4

10 min	Brainstorm New Features <ol style="list-style-type: none"> Ask students what new features might make the game more fun or interesting to play. List out some of the suggested new features on the board.
15 min	New Features Worksheet

	<ol style="list-style-type: none"> 1. Ask student groups to choose two new features to add to their game. 2. Students fill out the <i>New Features Worksheet</i> with their partners. <ol style="list-style-type: none"> a. what new component(s) might be needed. b. what blocks might you use. c. how will you test your new features?
20 min	Add New Features Following the <i>New Features Worksheet</i> , students implement the first feature. Students can use the <i>Student Guide: Challenge</i> as a resource for implementing new features.

Lesson 5

5 min	Check-in Check in with student groups to see if they've implemented one of their two features.
35 min	Add New Features Following the <i>New Feature Worksheet</i> , student groups implement the second feature (or complete the first). They should work with other groups implementing the same feature if they need help. Students can use the <i>Student Guide: Challenge</i> as a resource for implementing new features.
5 min	Wrapup Check in again with teams to see if they have implemented at least one new feature. In the final lesson, they will share their new features with their classmates.

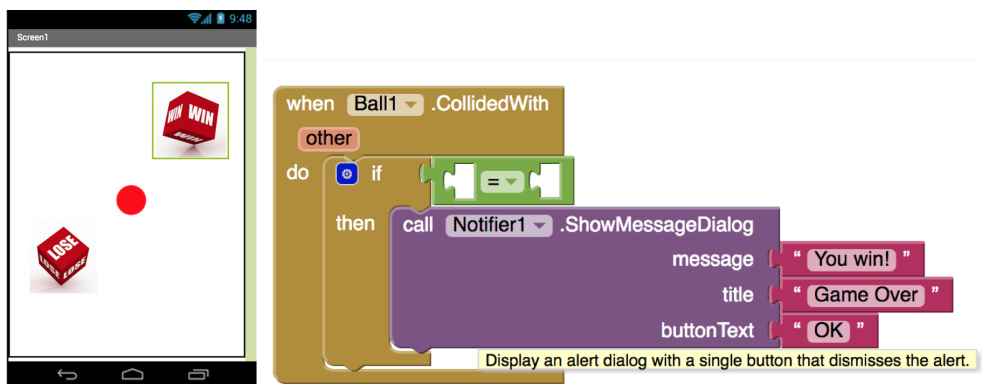
Lesson 6


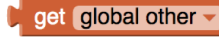
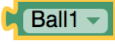
10 min	Install apk on Mobile Device <ol style="list-style-type: none">1. Teacher demonstrates how to create an apk and install the Find the Gold game on the mobile device.2. Students install their apk's on their devices.
25 min	Feedback <ol style="list-style-type: none">1. Show students the <i>Feedback Worksheet</i>, and demonstrate positive, constructive feedback.2. Students exchange their mobile device with another group (that has implemented a different feature than them) and play each other's games.3. Students fill out the <i>Feedback Worksheet</i> about the other group's game. Remind them about being constructive and positive.4. Students do another switch with a different group, and fill out the <i>Feedback Worksheet</i> for that group too.
10 min	Wrap-Up <p>Ask students to reflect on the unit:</p> <ol style="list-style-type: none">1. Answer multiple choice questions.2. Fill out Survey of Learning Attitudes.3. Answer questions on Collaboration.

6. Assessment

Multiple-choice questions

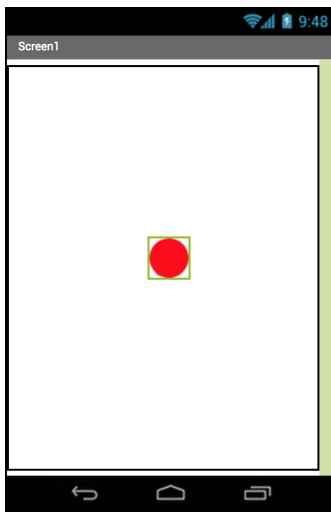
1. The following app is created. There is a Ball, a WinSprite, and a LoseSprite on a Canvas. If the ball collides with the WinSprite, the user wins the game. To test this, the programmer adds the when Ball1.CollidedWith block, and the if block. What goes in the missing inputs for the equals block?



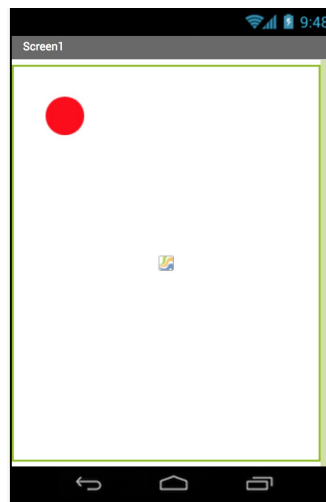
- A.  
- B.  
- C.  
- D.  

2. A student creates an app with a Ball and wants to center it on the screen. She drags the Ball to the center of the screen in the Designer, but when she tests the app, it appears in the top left corner of the screen. What can she do to center the Ball on the screen, and make sure it will be centered on any mobile device the app runs on?

Designer



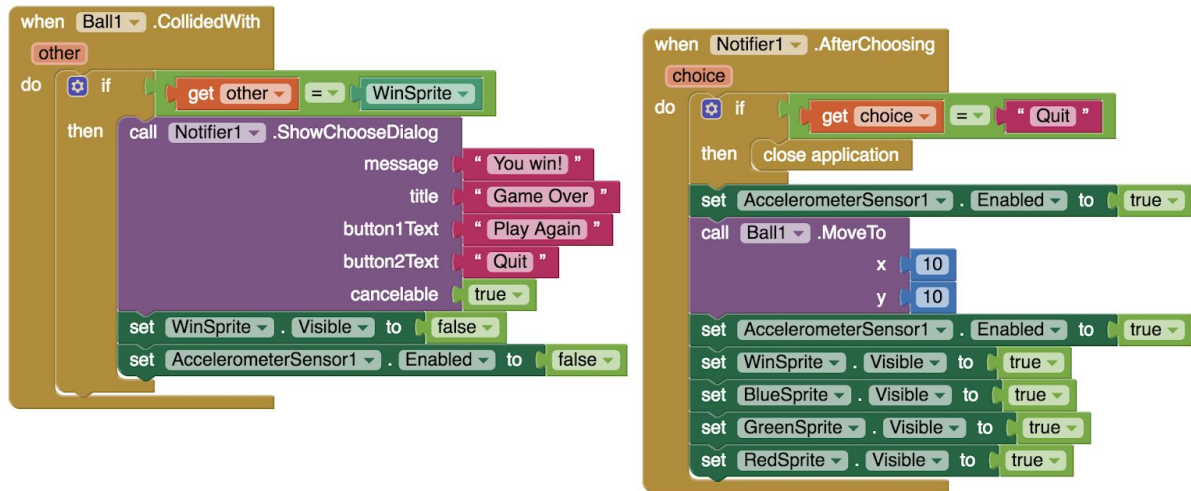
Testing



- A. Make the X,Y of the Ball larger numbers.
- B. Use the Accelerometer to roll it to the center of the screen.
- C. Set the X, Y of the Ball to 50% of the Canvas width and height respectively.
- D. Change the radius of the Ball to a larger number.

Answer: C

3. Look at the following code. What will happen if the user chooses “Play Again” when playing the app?



- A. The app will close.
- B. The Accelerometer will stop.
- C. The ImageSprites all disappear.
- D. The ImageSprites all appear.

(Answer: D)

Survey of learning attitudes

In order to evaluate students' attitude, perception, and understanding towards coding, students are required to finish a 5-point scale survey below by putting a “✓” in the appropriate box.

After completion of this unit, I think...	Disagree	Somewhat disagree	Neutral	Somewhat agree	Agree
Learning how to make apps makes me want to learn more about coding.					
I feel more connected to the technology around me when I make apps.					
I am excited to share this app with friends and family.					

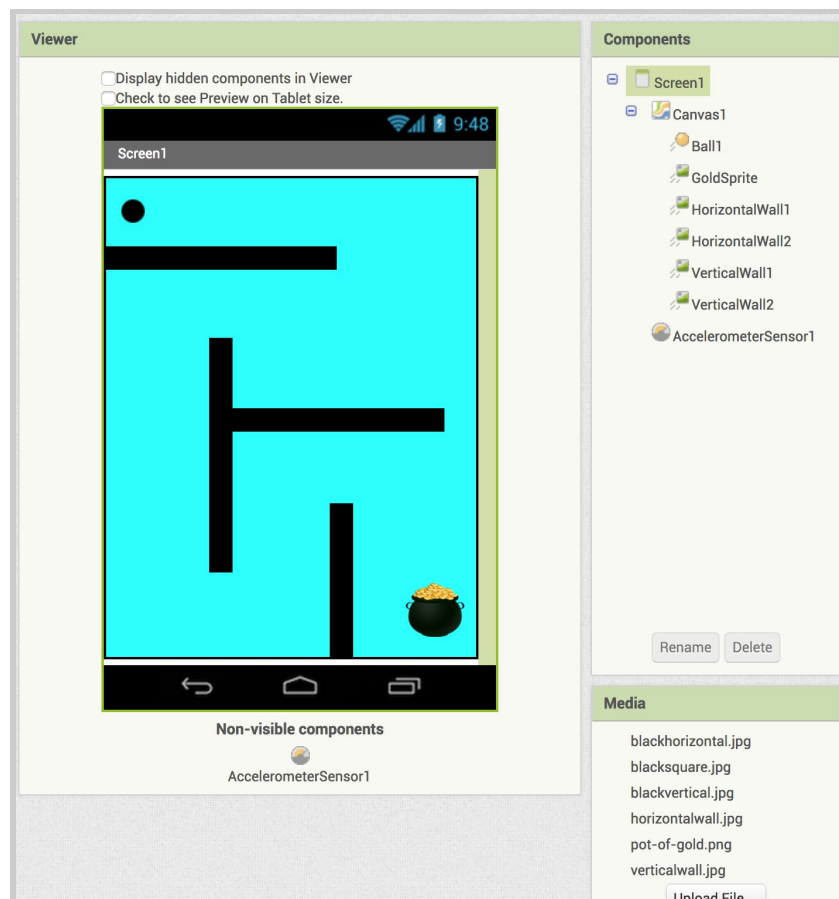
Self Assessment on Collaboration

Ask students to reflect on how well they worked with their partner, and to answer the following questions honestly.

1. Did you like working with another person? Do you feel you were a good partner, and respected and encouraged your partner in the project?
2. What was your role as a partner in this project? What did you do and what did your partner do?

7. Screen Design and Code

Designer



Blocks

```

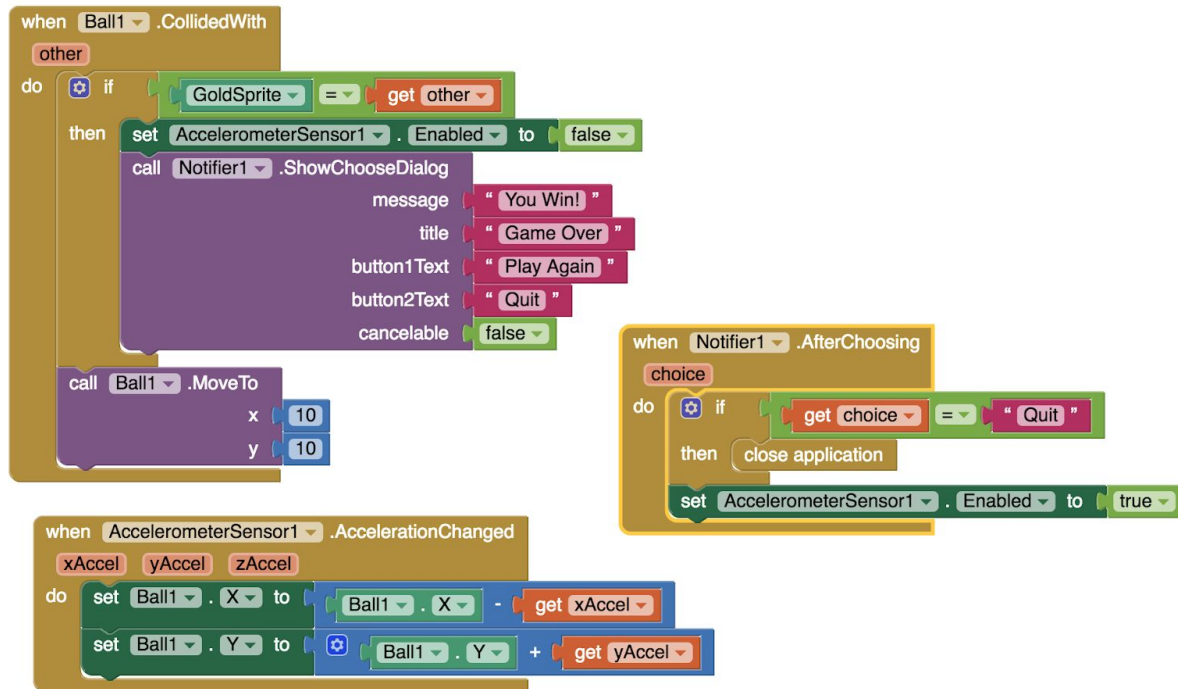
when Screen1.Initialize
do call SetUpMaze

```

```

to SetUpMaze
do
  set GoldSprite.X to Canvas1.Width - 70
  set GoldSprite.Y to Canvas1.Height - 70
  set HorizontalWall1.X to 0
  set HorizontalWall1.Y to Canvas1.Height * 0.2
  set HorizontalWall1.Width to Canvas1.Width * 0.6
  set VerticalWall1.X to Canvas1.Width * 0.3
  set VerticalWall1.Y to Canvas1.Height * 0.4
  set VerticalWall1.Height to Canvas1.Height * 0.4
  set HorizontalWall2.Width to Canvas1.Width * 0.7
  set HorizontalWall2.X to Canvas1.Width * 0.3
  set HorizontalWall2.Y to Canvas1.Height * 0.5
  set VerticalWall2.X to Canvas1.Width * 0.7
  set VerticalWall2.Y to Canvas1.Height * 0.8
  set VerticalWall2.Height to Canvas1.Height * 0.3

```



Appendix 1

Find the Gold Teacher's Guide

Lesson 1

Learning Objectives

At the end of this lesson, students should be able to:

1. Demonstrate understanding of properties of Canvas, Ball, and ImageSprite components in App Inventor.
2. Use percentages to place components on the screen in an app.

Lesson Outline

Introduction to Unit (15 minutes)

Introduce the unit by relating it to apps students may have used on their own mobile devices.

1. Ask students if they have ever played games on their tablets or computers. Is it fun? Ask for examples of favorite apps.
2. Explain that they will make a game app called “FindTheGold” in this unit. It works as follows. Players navigate a ball through a maze on the mobile device by tilting the device.
3. Demonstrate the finished app (FindTheGold.aia).
4. Students will work in pairs on this project. The teacher should determine the appropriate pairings based on student ability and skill level. Explain that students will be working together with a partner to code their apps and implement the new features. They will use

the pair programming model to code their project. Show the Code.org video (<https://youtu.be/vgkahOzFH2Q>) to show how it works.

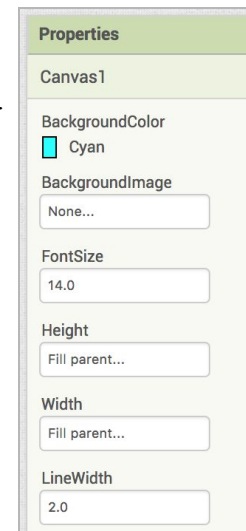
Here are a set of rules for working using a Pair Programming model:

DO	DON'T
<ul style="list-style-type: none">• Be respectful• Talk to one another about the work• Explain what you are doing• Think ahead and make suggestions• Switch roles often	<ul style="list-style-type: none">• Be a bossy navigator• Grab the driver's mouse or keyboard

Demonstration of Canvas, Ball, and ImageSprite Components in App Inventor (15 minutes)

Demonstrate the components in the **Drawing and Animation** drawer in the Designer - **Canvas**, **Ball**, and **ImageSprite**.

- **Canvas** is your background for sprites to appear and move on.
 - Generally, you will set its *Width* and *Height* to **“Fill Parent”**.

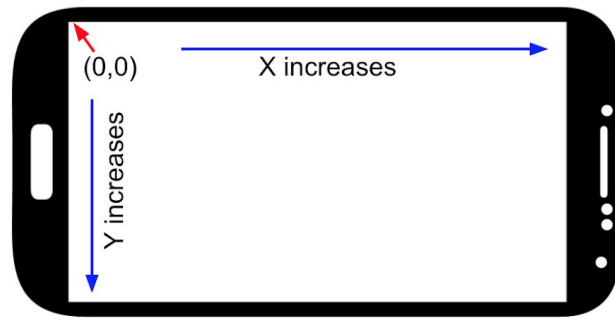
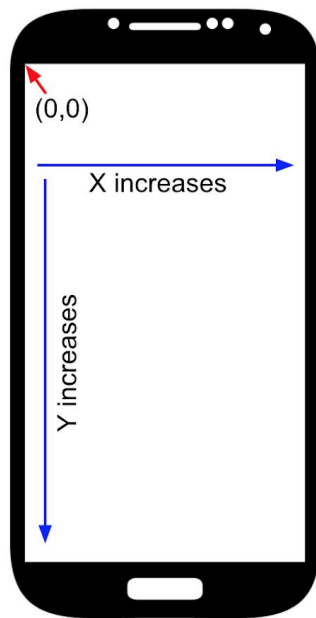


- **Balls/ImageSprites** are the elements on the Canvas that can be controlled by the user interaction and by coding. Below are the properties used for these components in this app.

ImageSprite Properties

Ball Properties

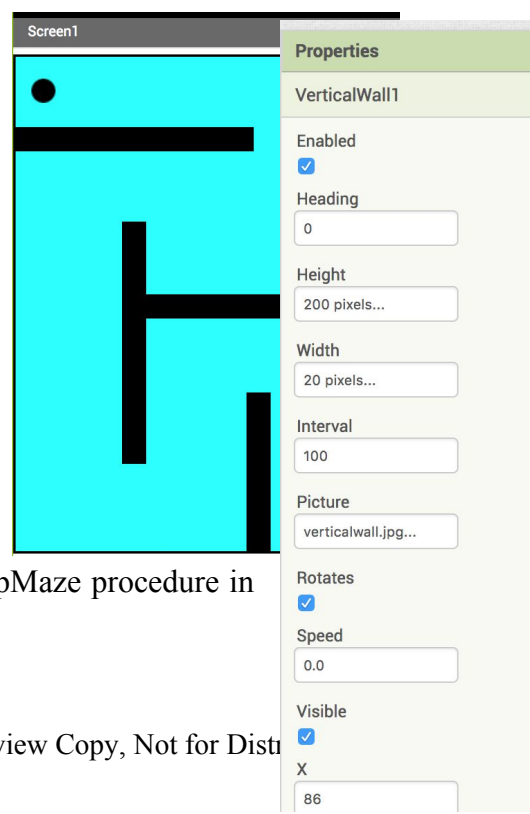
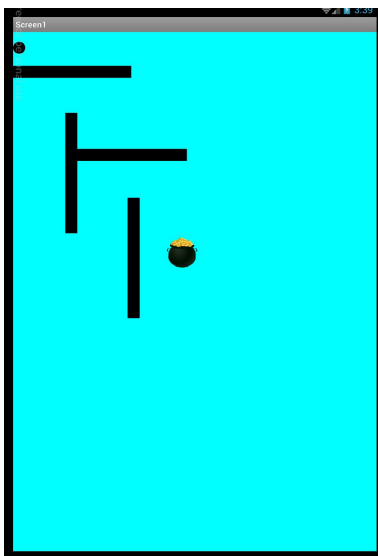
- Note that the Canvas uses the Cartesian coordinate system, with X as the horizontal axis and Y as the vertical access. Each ImageSprite or Ball is located by its X,Y coordinates in this system. However, the origin (0,0) is located in the upper left corner of the Canvas. X increases as you move to the right. Y increases as you move down (note this is different in a normal Cartesian coordinate system, where Y increases as you move up). This app works in portrait mode. However, the X and Y axes remain the same for an app in landscape mode. X is horizontal, Y is vertical.



- Mobile devices come in different sizes and resolutions (generally measured in pixels - horizontally by vertically. For example 768x1024). Because of differing sizes, placing ImageSprites and Ball components based solely on pixels may change the appearance.

High Resolution device

Low Resolution device

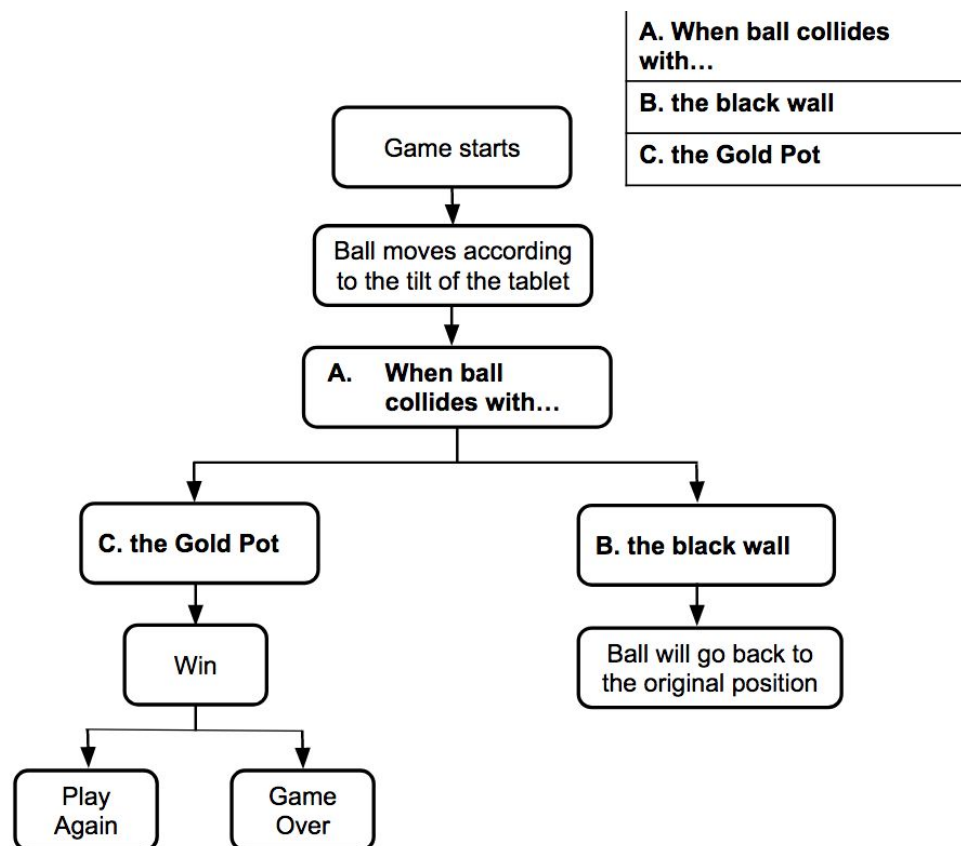


- Students will be asked to complete the SetUpMaze procedure in

the Coding Activity that follows. Explain placement of the Wall ImageSprites are initially set in the Properties panel by their X and Y at certain pixels on the screen. That changes depending on the resolution, so they will be asked to set the X and Y coordinates in code blocks based on percentages. See the Coding Activity below for more details.

Coding Activity (15 minutes)

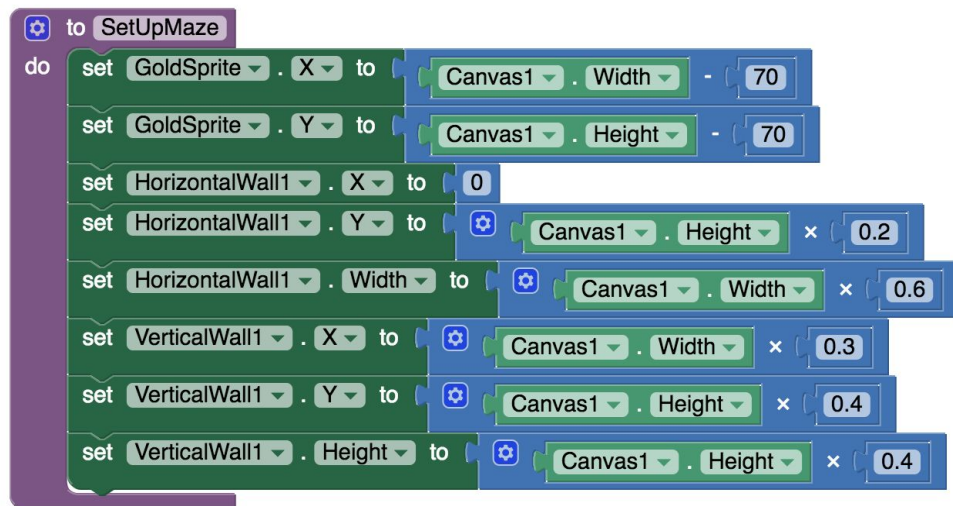
1. Ask students to complete the missing blocks in the simplified flowchart in the Student Guide.



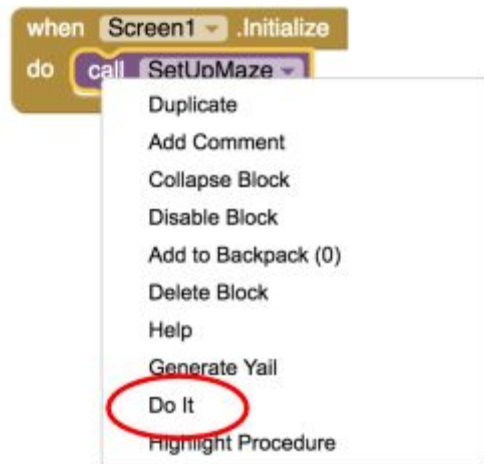
2. Distribute the project template, FindtheGold_template.aia, to students. Ask them to start coding the app following Student Guide Part 1.
 - a. Ask students to discuss with their classmates the function of each component in the Designer (page 2 of the Student Guide). They should be able to identify walls and Gold as ImageSprites, and the Ball and Canvas.

Components	Naming	Functions
Ball	Ball1	Moves according to the tilt direction of the tablet using Accelerometer Sensor.
ImageSprite	GoldSprite	<ul style="list-style-type: none"> • The target • When ball hits GoldSprite, win game and call Notifier
ImageSprite	WallSprite1 WallSprite2 WallSprite3 WallSprite4	<ul style="list-style-type: none"> • Obstacles (black walls) • When ball hits these obstacles, the ball moves back to its original position
AccelerometerSensor	AccelerometerSensor1	Gets the values of of XAccel and YAccel while the tablet tilts.
Clock	Clock1	Moves the ball by the values of XAccel and YAccel at a certain time interval.
Notifier*	Notifier1	Allows users to choose what to do when ball hits the target.

- b. The template contains a partially complete procedure, called SetUpMaze, that sets the X,Y, and Width (or Height) of the Wall ImageSprites and GoldSprite to account for differing resolutions. Students are to add code blocks to the SetUpMaze procedure to place the remaining Wall ImageSprites on the screen. They can experiment with different percentages (decimal numbers) to create a maze they are happy with. Note that students can decide the placement - there is no right or wrong placement for the components. Students will add code blocks for HorizontalWall2 and VerticalWall2.



- c. Students will test the app to make sure the maze fits correctly on the mobile device. This may take some trial and error. A quick way to run through the code after changes are made is with the “Do It” command. If students right-click on the call SetUpMaze block and select “Do It” they can immediately see the effects of changes to their code.



Appendix 2

Find the Gold Teacher's Guide

Lesson 2

Learning Objectives

At the end of this lesson, students should be able to:

1. Use the AccelerometerSensor in App Inventor to control movement of a Ball sprite in a game app.
2. Demonstrate understanding that animation can be achieved by changing the X and Y coordinates of a Ball.

Introduction and Review (5 minutes)

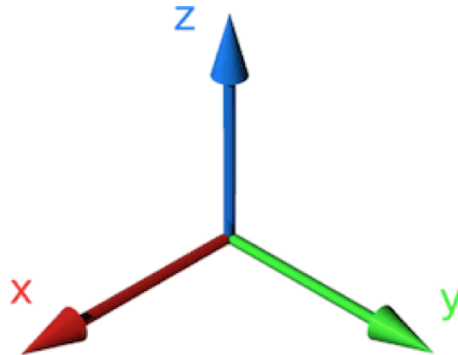
1. Review the basic Find the Gold app and new components used. Go over the Canvas, ImageSprite, and Ball components and check that all students are happy with the placement of the walls in their maze apps.
2. Explain to students that the focus of this lesson is the Accelerometer Sensor. The Accelerometer will be used to control the Ball movement in the maze app.

Introduction to the Accelerometer (15 minutes)

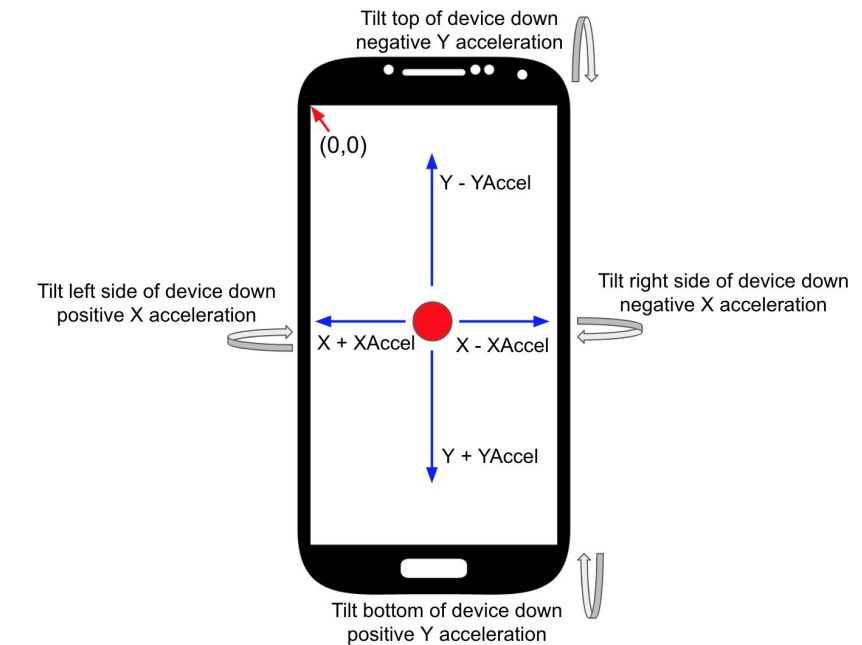
It is not expected that students, especially at the middle school age, will understand acceleration or the physics behind the Accelerometer. Students should understand that the X and Y coordinates of the Ball will be updated according to the acceleration values returned by the AccelerometerSensor in App Inventor.

1. Explain the Accelerometer Sensor in a smartphone/tablet and explain that it measures movement in X, Y, and Z directions.

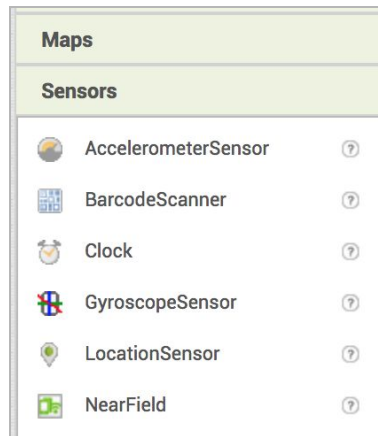
- a. Accelerometer Sensor: Non-visible component that can detect shaking and measure acceleration approximately in three dimensions.



- b. For the maze game, students will deal only with the X acceleration and Y acceleration, how much is it tilted in either direction. When a mobile device is in portrait mode, the tilting the device side to side changes the X Acceleration. Tilt left is positive X acceleration. Tilt right is negative X acceleration. Tilting the phone top to bottom changes the Y Acceleration. Tilting the top of the device downwards is negative Y acceleration. Tilting the bottom of the device downwards is positive Y acceleration. As the device is tilted in a direction, the X and Y acceleration changes, and therefore, the Ball will “move” in that direction. To move the ball according to the tilt, you add YAccel to Y and subtract XAccel to X.

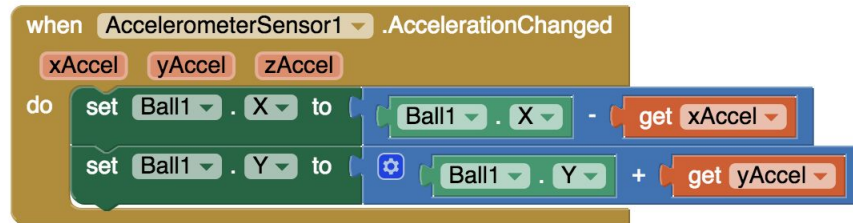


- c. The AccelerometerSensor is found under the Sensors drawer.



- d. This app will use the **AccelerometerChanged** event to trigger movement of the Ball. The current X and Y acceleration values will be added to the X and Y coordinates of the Ball. As the device is tilted in a direction, the X and Y

acceleration changes, and therefore, the Ball will “move” in that direction. Note that XAccel is subtracted from X, while YAccel is added to Y.



Add Ball Movement to App (25 minutes)

Students follow *Find the Gold Student Guide: Part 2* to implement the Ball movement based on the Accelerometer Sensor.

Appendix 3

Find the Gold Teacher's Guide

Lesson 3

Learning Objectives

At the end of this lesson, students should be able to:

1. Demonstrate understanding of Ball/ImageSprite movement and collision detection in the App Inventor environment.
2. Use conditionals correctly in a program.
3. Demonstrate understanding of the use of the Notifier component in an app.

Lesson Outline

Introduction and Review (5 minutes)

1. Review with students the basic Find the Gold app and new components used (Accelerometer, Clock, Canvas, Ball, ImageSprite).
2. Check that students have completed Part 1 and 2 of the app and that they can control ball movement with the Accelerometer.
3. Let students know that they will add to the app by testing for collision, and adding the Notifier component to let the player know when they win the game. They will test if the Ball collides with any of the Wall ImageSprites and if so, move the Ball back to its starting position. If it collides with the GoldSprite, the game is over, so they will notify the user the game is over.

Conditionals (10 minutes)

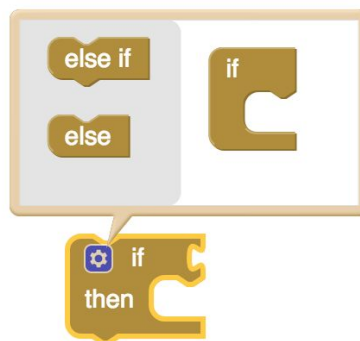
In a computer program or mobile app, there are times when you don't want to execute all of your code. It will depend on what the "conditions" are.

All conditional blocks are found in the Control drawer. The **if-then** block will test *if* a condition is true, and will execute the code in the *then* part of the block. If the condition is false, the code is skipped.

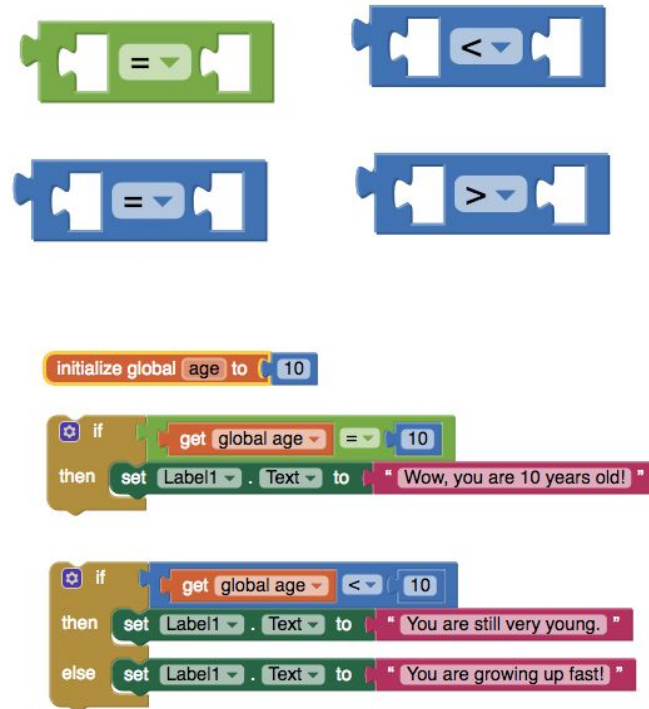


Explain that **if** blocks test for conditions. If the condition is true, the blocks inside the **if** block are executed. If the condition is false, those blocks are skipped.

1. Show students the Control drawer, and how to drag out and use an **if** block. Also demonstrate how to make **if-then-else** with the blue mutator icon. Students will not use this block in Parts 1 or 2, but some may need to use it for their new feature, so all students should be familiar with both types of blocks.



2. Most blocks that attach to the if block are found in the Logic (green) drawer, although you can also find similar blocks in the Math (blue) drawer (testing greater than and less than, for example).

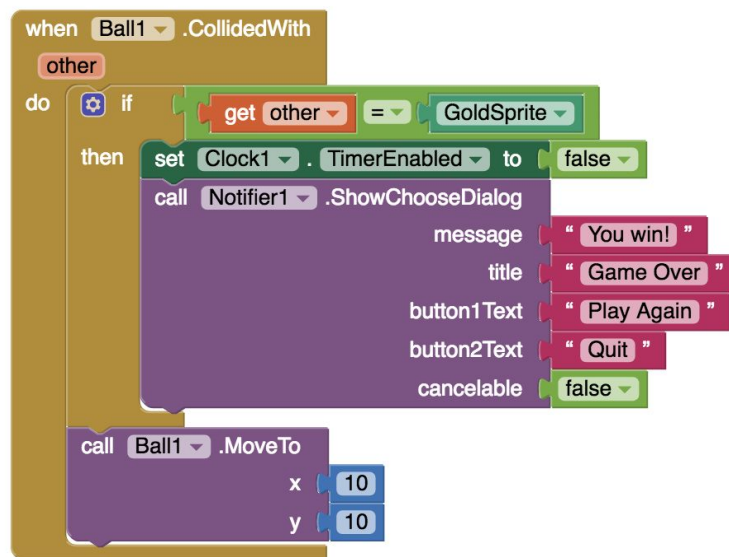


In the above examples, if $\text{age} = 10$, the label is set to “Wow, you are 10 years old!”. $\text{age} = 10$ is the *condition* that is checked. If it is true, the **Label1.Text** is set. Otherwise, it is not set.

In the if-then-else example, **Label1.Text** is set in both cases, but to different text. If the condition is true ($\text{age} < 10$), it is set to “You are still very young”. If the condition is false ($\text{age} \geq 10$), the label is set to “You are growing very fast”. So, if-then-else blocks cover both the true and false cases for the condition.

Teachers should cover the **if** statement (conditional) used in the **Ball1.CollidedWith** blocks. If the condition is true, the blocks inside the if block are executed. The app detects collision between the **Ball** and the **ImageSprites** (walls and Gold Pot) to change outcomes of the game. The *other* parameter tells which **ImageSprite** collided with **Ball1**. You can test using an **if** block

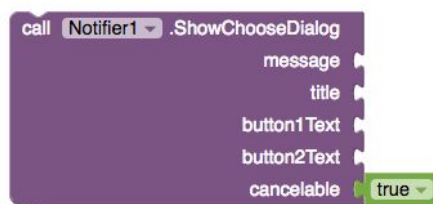
to check which other **ImageSprite** collided with **Ball1**.



In this case, if the **GoldSprite** collides with **Ball1**, the game is over. A Notifier message is displayed to tell the that the game is over and gives them the option to play again. The code block at the bottom, **Ball1.MoveTo**, will be executed in any case. Just the code inside the if block is affected by whether the condition is true or not.

Introduction to Notifier (5 minutes)

1. Students will add the **Notifier** component in this lesson, to let the player know when they win the game. The **Notifier** display alerts and messages to the user. This app uses **Notifier.ShowChooseDialog** to tell the user the game is over and asks whether they want to play again or not.
2. Teacher demonstrates to students how to add the **Notifier** component in the Designer, and shows the **ShowChooseDialog** block in the Blocks Editor. Teacher should also explain that once the user presses a button in the dialog box, the **Notifier.AfterChoosing** event is triggered.



3. For this lesson, students will write code to notify user that game is over and given them

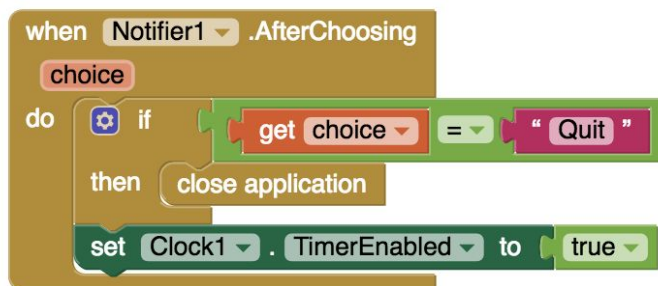


two button options - “Play Again” or “Quit”.

- a. If user chooses quit, close application.
- b. If user chooses play again, go back to start of the maze.



4. The **Notifier.AfterChoosing** event is triggered once the user selects a choice from above. The code in this block should test for which response was given.



Add Testing for Collision and Notifier to App (25 minutes)

Students follow *Find the Gold Student Guide: Part 2* to implement the following:

1. Add **Ball1.CollidedWith** block to test for collision.
2. If the ball collides with a wall, go back to start of maze.
3. If the ball collides with the pot of gold, stop ball moving.
 - a. Notify user that game is over and ask if they want to play again or quit.
 - b. If user chooses quit, close application.
 - c. If user chooses play again, go back to start of the maze.

Appendix 4

Find the Gold Teacher's Guide

Lesson 4

Learning Objectives

At the end of this lesson, students should be able to:

1. Plan and implement a new feature in an app.
2. Follow a design plan to implement it.
3. Test and debug an app to ensure it works as expected.

Lesson Outline

Brainstorm New Features (10 minutes)

In this lesson, students will extend the existing Find the Gold app to add something of their own. Ask students what new features might make the game more fun or interesting to play. List out some of the suggested new features on the board. Explain that students will choose two new features and add it to the Find the Gold game.

New Features Worksheet (15 minutes)

Ask student groups to choose two new features from the brainstorming list to add to the app. Students work with their partners to fill out the *New Features Worksheet*.

1. what new component(s) might be needed
2. what blocks you might use
3. what steps you might take to implement the new feature

Add New Feature (20 minutes)

Following the *New Features Worksheet*, student groups implement their first new feature. Students can use the *Student Guide: Challenge* as a resource for implementing new features. If students implement one feature and have additional time, they may add their second feature to their app.

Appendix 5

Find the Gold Teacher's Guide

Lesson 5

Learning Objectives

At the end of this lesson, students should be able to:

1. Follow a design plan to implement a new feature in an app.
2. Test and debug an app to ensure it works as expected.

Lesson Outline

Check-in (5 minutes)

Check in with student groups to see if they've implemented one of their two features. If students are struggling, their goal should be to implement just the first feature from their *New Features Worksheet*.

Add New Features (35 minutes)

Following the *New Features Worksheet*, students implement the second feature (or complete the first). They should work with other groups implementing the same feature to help each other. Students can use the *Student Guide: Challenge* as a resource for implementing new features.

Wrapup (5 minutes)

Check in again with teams to see if they have implemented at least one new feature. In the final lesson, they will share their new features with their classmates.

Appendix 6

Find the Gold Teacher's Guide

Lesson 6

Learning Objectives

At the end of this lesson, students should be able to:

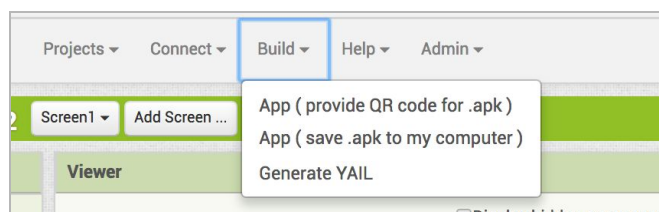
1. Create an apk file from an App Inventor project, and install it on a device.
2. Provide and accept constructive feedback to and from peers.

Lesson Outline

Install apk on Mobile Device (10 minutes)

Demonstrate to students how to create an apk and install their Find the Gold game on the mobile device.

1. Under the Build menu, choose “App (provide QR code for apk)”.



2. Wait for the apk to build, and the QR code appears.
3. Scan the QR code with a BarcodeScanner app on the mobile device. Note that MIT AI2 Companion does work as a scanner app if another one is not installed.

4. Follow the prompts to install the app.

Explain to students that by creating an apk, they are actually installing a completed app on their mobile device. Unlike the live testing with MIT AI2 Companion, this results in a permanently installed app.

Have students install their apk on their device, and make sure that it can open and run.

Feedback (25 minutes)

1. Show students the *Feedback Worksheet*, and demonstrate positive, constructive feedback.
2. Student groups exchange their mobile devices with another group (who has implemented a different feature than them) and they will play each other's games.
3. Students fill out the *Feedback Worksheet* about the other group's games. Remind them about being constructive and positive.
4. Students do another switch with a different group, and fill out *Feedback Worksheet* for that group too.
5. If there is time, start a class discussion and ask students to describe a feature that another group has added that they are impressed with. Why?

Wrap-Up (10 minutes)

Ask students to reflect on the unit:

1. Answer multiple choice questions.
2. Fill out Survey of Learning Attitudes.
3. Answer questions on Collaboration.