APP INVENTOR

OVERVIEW

AppInventor is a special website that enables you to create your own apps for Android devices. The exercises below should take 2-3 hours to complete and will help you to get started. To develop your skills further search online for tutorials and YouTube videos.

GETTING STARTED

WHAT YOU WILL NEED

- A computer with the Google Chrome web browser
- An Android phone or tablet (sorry iPhones are not compatible)
- A Google/Gmail address and password
- Internet Access

Additionally on your Android device you need to open your 'Settings' and under 'Security' you should tick the box for 'Unknown Sources - Allow installation of apps from unknown sources'. After creating your own App you should switch this option off again.

TESTING YOUR APP

At home you may install a special app on your Android device called 'MIT AI Companion' that makes it easy to test your apps as you develop them. However, on the C2K network each time you want to run your app:

- Download your app APK file using the 'Build' menu
- Plug your Android device into the computer using the USB cable
- Copy the APK file from your 'Downloads' folder to the Android device
- Use a file manager (such as ES Explorer or Astro) on the Android device to locate the APK file and tap to install the APK file
- After installing your App you should be able to open it and view it in all its glory

WHAT IS AN APK FILE

An Android Package or APK is a special file that contains all the functionality, images, sounds for an app. You may not have realised it but each time you install an app from the Play Store you are actually downloading an APK file and installing it on your device.

HOW DO I SHARE MY APPS

When you have finished developing your App you can send the APK file to your friends by email or USB. However to install apps that are not in the Play Store they will also have to visit their 'Settings' » 'Security options' to allow 'Installing unsigned from Unknown Sources'.

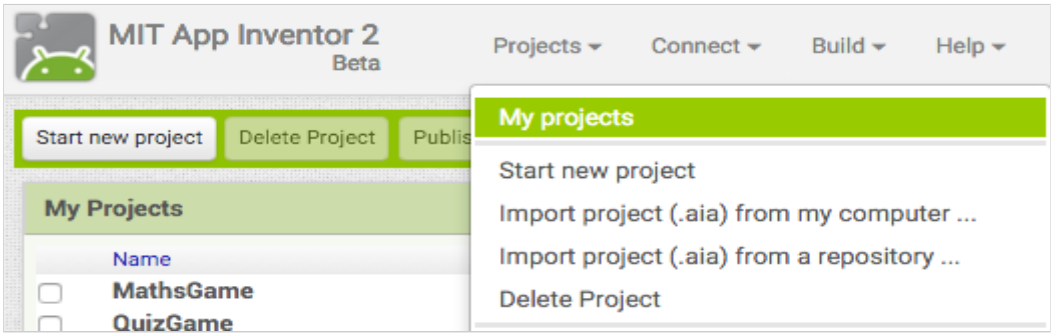
The alternative is to add your App to the Google Play Store so that anyone in the world can install your App easily. To do so you will need to:

- Sign up as a developer at https://play.google.com/apps/publish/
- Pay a one-time fee of $25
- Upload your APK file

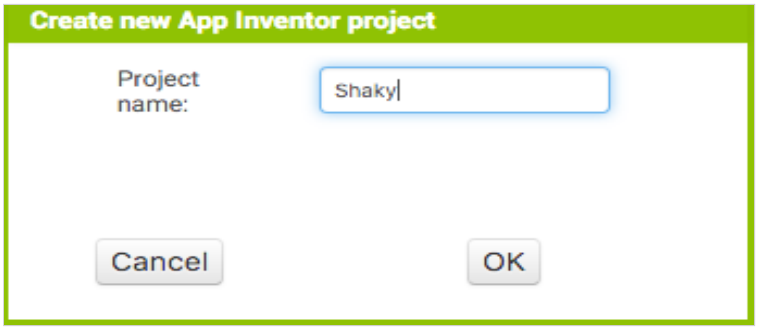
EXERCISE 1

In this exercise we will create an app that will speak when someone shakes the device.

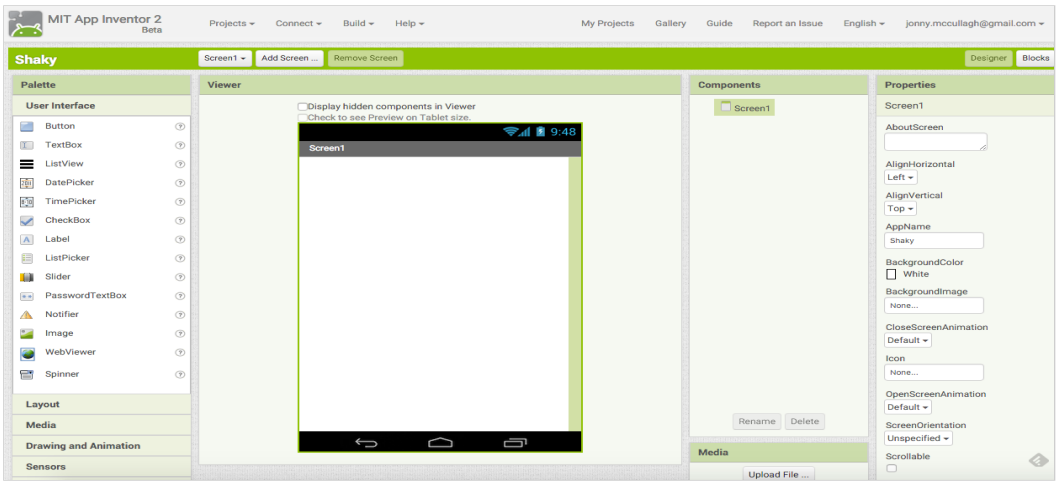
1. Visit the App Inventor website at http://ai2.appinventor.mit.edu
2. Login using your Google email address and password
3. Click the 'Start new project' button as shown in the image below:



4. Type 'Shaky' as the name of your App (as shown below) and click the 'OK' button:

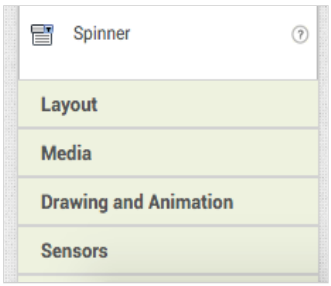


5. A blank app design window opens (as shown below). It is important to spend a few minutes understanding the layout of this window.



There are several things to notice on this screen:

- The '**Palette**' on the left shows a list of widgets you can add to your app. There are many things that can be added to your app including buttons and images. The list is very long and there are extra items under the headings 'Layout', 'Media', Drawing' and 'Sensors'.

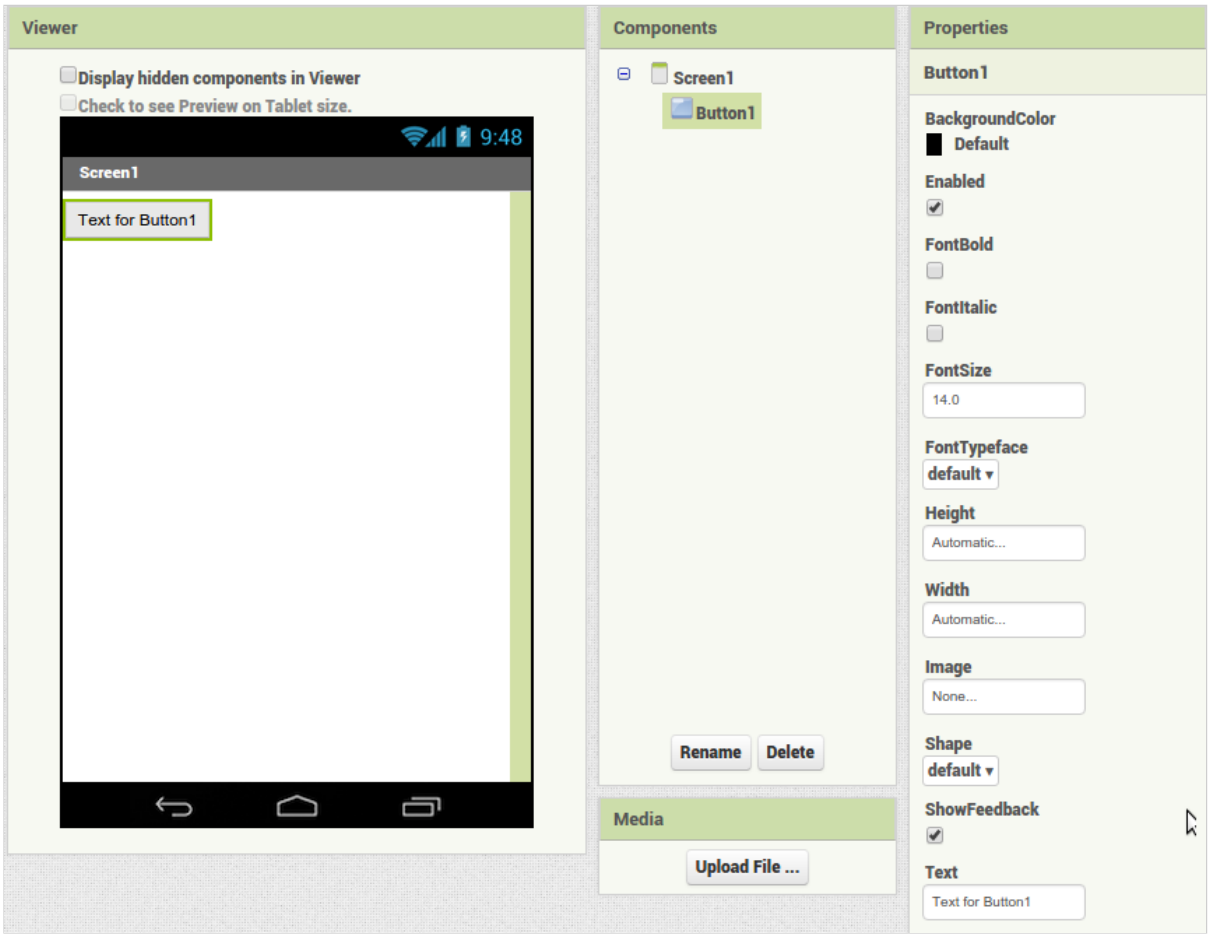


- The '**Viewer**' in the centre shows a preview of how your app will look
- The '**Components**' panel shows a list of items that have already been added to your app
- The '**Media**' panel is below the 'components' panel and allows you to upload files (such as pictures

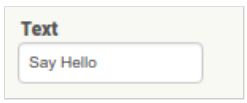
and sounds) to be used within your app.

- The '**Properties**' panel allows you to amend the settings of any items you have already added to your app. Choose an item in the 'components' list and the 'properties' panel will show the settings for that item.

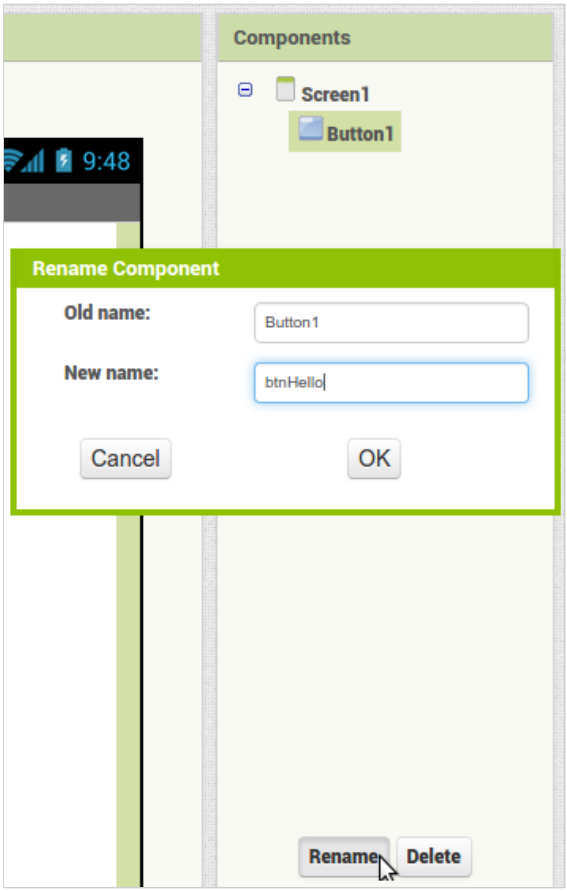
6. From the 'Palette' click-and-drag a button from the palette to the 'Viewer' (as shown below):



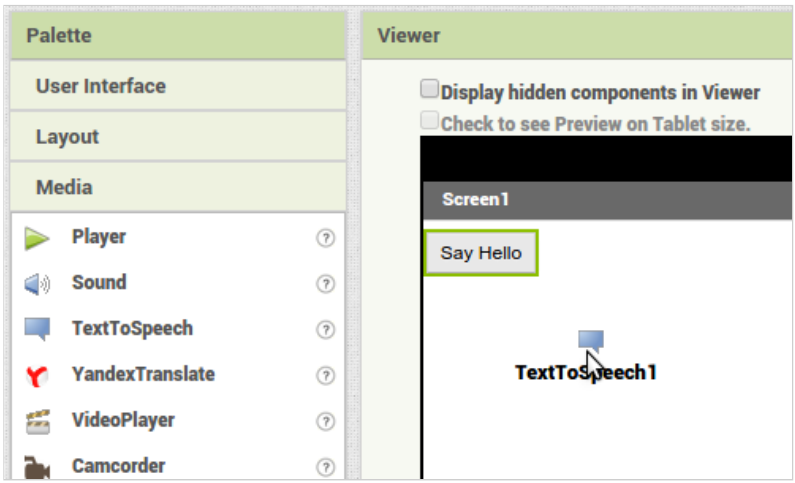
7. There should now be a button on your App. Click the button and notice that the 'Properties' panel on the right changes to show the properties of your button you clicked. In the 'Properties' panel you can change the colour of your button and many other properties but for now let's change the 'Text' attribute. By default it should be "Text for Button1" but change that to "Say Hello".



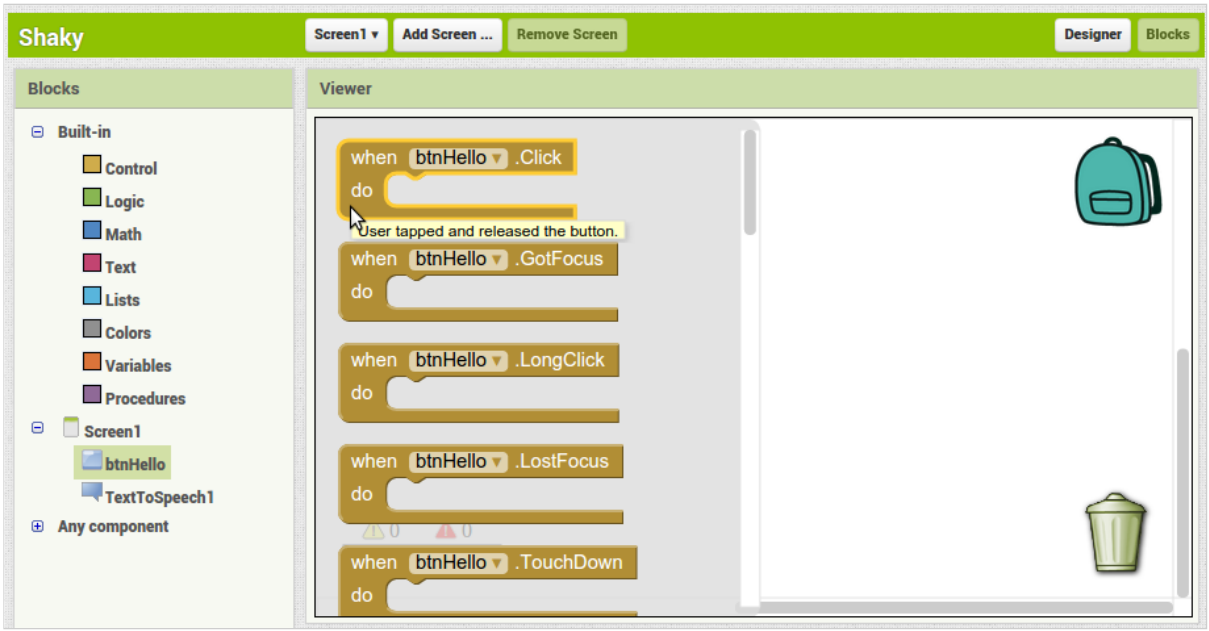
8. Notice that the 'Components' panel now shows your screen object 'Screen1' and on 'Screen1' there is a button named 'Button1'. Click the 'Button1' component in the 'Components' Panel and then click the 'Rename' button below that. Change the name of the button from 'Button1' to 'btnHello' (as shown below), then click 'OK'.



9. Next add another component to the Screen. From the 'Palette' under the 'Media' section click the 'TextToSpeech' component and drag it onto your screen. Because this is not a component the user can see it will be placed underneath the screen in the 'Non-Visible Components' section.

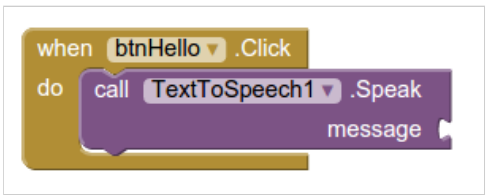


10. Now it is time to program your components. At the top right of the AppInventor page you should see buttons for 'Designer' and 'Blocks'. A the moment you are in 'Designer' mode. Click the 'Blocks' button. You will notice that the design of your App is hidden and you can now start adding programming blocks.

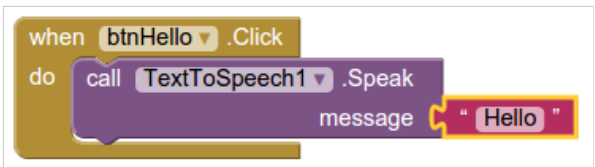


11. On the left panel you should see the components you currently have in your App (Screen1, btnHello, TextToSpeech1). Click on 'btnHello' and you will notice that all the blocks available for your button appear. The first block is the one we need. Click and drag it onto the workspace area.

12. Next in the left Blocks panel click the 'TextToSpeech1' component and choose the third (purple) block. Drag it onto the workspace and drop it inside the "When btnHello.Click" block. The blocks stick together like jigsaw pieces. Your blocks should look like the ones shown below:



13. Next from the left Blocks panel choose 'Text'. The available blocks should appear. Click-and-drag the first block onto the workspace and join it to the 'Message' part of the speak block. Click the space inside the quotes and type "Hello". Your blocks should look like the ones below:

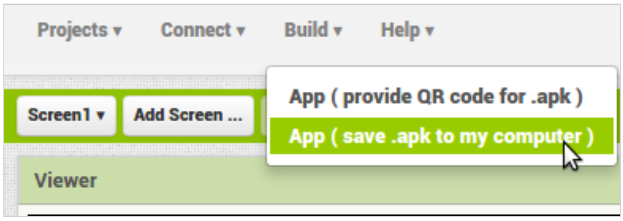


14. You can now test your App. When you press the button on the screen the phone should say 'Hello'.

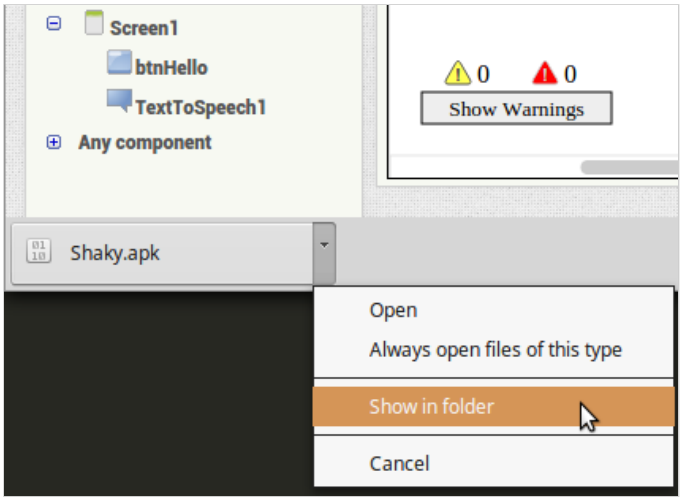
HOW DO I SHARE MY APPS

If you are on the C2K network you should use the APK file:

- From the top menu choose 'Build' > 'App (save .apk to my computer)'



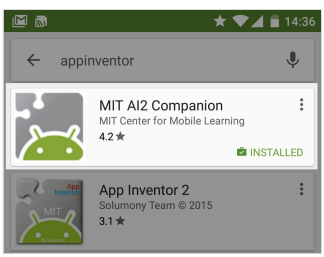
- The .apk file should start building and then download
- Notice that the downloaded APK file appears at the bottom of the Chrome Web Browser (see below). Click the arrow and choose 'Show in folder'.



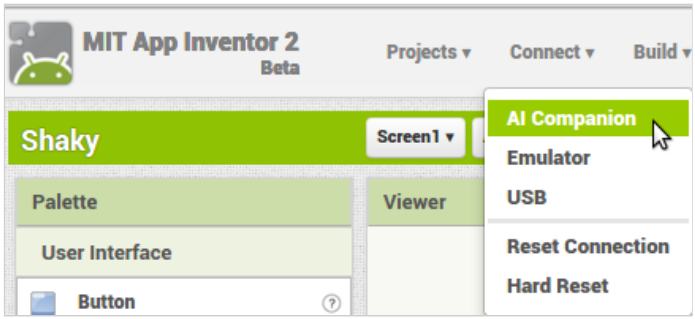
- Your computer will then show the 'Shaky.apk' file. You should connect your Android tablet or phone to your computer using the USB cable and then Copy/Paste the 'Shaky.apk' file from your computer onto the device.
- When the file has been copied onto your Android device use a file manager such as 'Astro' or 'ES Explorer' to find and then open the file. You should choose to "Open with Package Installer". Once the APK has been installed you can 'Open' it and test the button.
- If the APK will not install you may need to amend your Security Settings for 'Unknown Sources' as described on the first page.

If you are at home you can test your app using the AI Companion:

- On your Android Phone/Tablet visit the 'Play Store'
- Search for the AI Companion App and install it

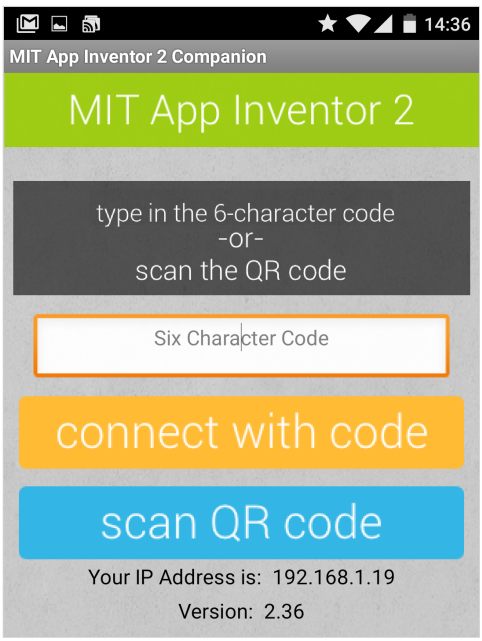


- On the AppInventor website choose the 'AI Companion' from the 'Connect' menu:



- You will receive a code to use with the AI Companion App on your mobile device (see example below):

- Open the AI Companion App and enter the code obtained from the step above



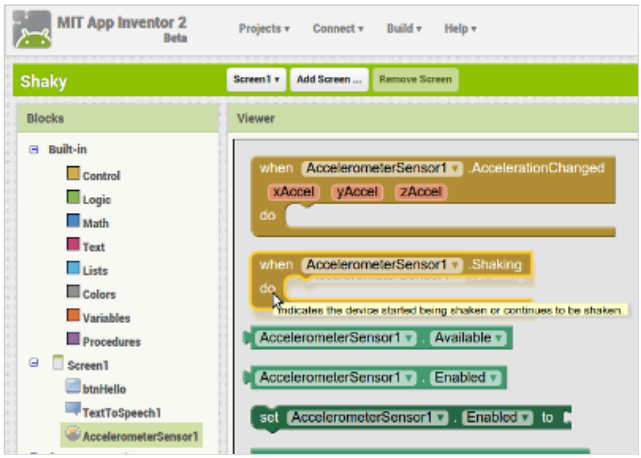
- If all goes well you should now be able to test your own App on your own Android device

EXERCISE 2

Let's add a little more functionality to your 'Shaky' App.

1. From the `'Palette'` click the `'Sensors'` option. Then click-and-drag an `'AccelerometerSensor'` from the palette to the `'Viewer'`. It will be added under the screen to `'Non-visible components'` as `'AccelerometerSensor1'`.

2. Return to the 'Blocks' editor and click the 'AccelerometerSensor1' block.



3. Click-and-drag the 'When AccelerometerSensor1 shaking, do' block and drag it onto the workspace.

4. Click the 'TextToSpeech1' object and drag the third (purple) block for 'call TextToSpeech1.Speak message' onto the 'AccelerometerSensor1 shaking' block.

5. Next from the left Blocks panel choose 'Text'. Click-and-drag the first block onto the workspace and join it to the 'Message' part of the speak block. Click the space inside the quotes and type "Stop shaking me". Your blocks should look like the ones below:

6. Test your App again and shake your device to see if if speaks to you.

EXERCISE 3

In this exercise we will create a Space Invaders style game app.

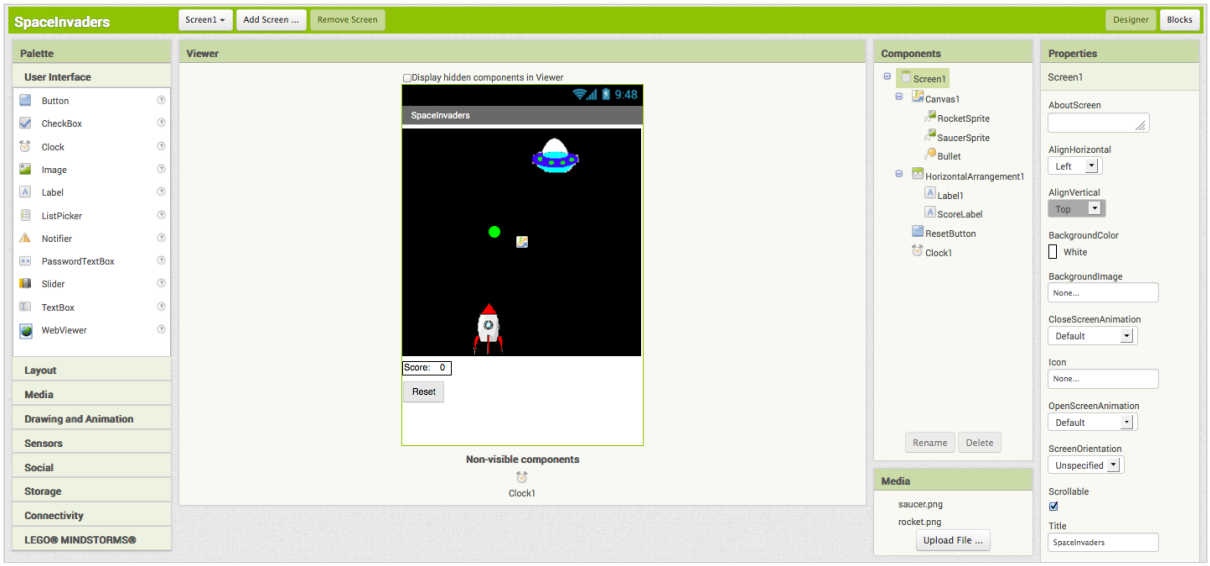
INTRODUCTION

By building the Space Invaders App you will get practice with using 'Clock' components and 'Timers', using Animation components such as 'Image Sprites' and the 'Canvas', setting visibility, and detecting collisions in App Inventor. You'll program an application that has a shooter ship whose goal is to shoot all the flying saucers on the screen.

1. Visit the App Inventor website at http://ai2.appinventor.mit.edu

2. Login using your Google email address and password
3. Click the 'Start new project' button.
4. Name your project 'SpaceInvaders' (without spaces!)
5. Use the palette to create the components shown below for your SpaceInvaders interface. When you finish, it should look something like the screenshot below.

Component Type	Palette Group	What you'll name it	Purpose of Component	Action
Canvas	Drawing and Animation	Canvas1	The background that we will be putting our sprites on	Change Width property to "Fill parent" and Height property to 250. Set the BackgroundColor property to Black.
ImageSprite	Drawing and Animation	RocketSprite	The rocket ship in our game	Upload the rocketship image and set the Picture property to "rocket.png". Set the Y property to 180. This will place the rocket at the bottom of the canvas.
ImageSprite	Drawing and Animation	SaucerSprite	The flying saucer in our game	Upload the saucer image and set the Picture property to "saucer.png". Y to 5
BallSprite	Drawing and Animation	Bullet	The bullet from the rocket ship.	Change PaintColor to Green and set the Radius property to 8.
Clock	User Interface	Clock1	We use the Clock for its Timer method to move the the saucer	Change TimerInterval property to 3000.
Horizontal Arrangement	Layout	HorizontalArrangement1	To contain Label1 and ScoreLabel	
Label	User Interface	Label1	To contain the word "Score: "	Change Text property to "Score: ".
Label	User Interface	ScoreLabel	To contain the current numerical score	Change Text property to "0".
Button	User Interface	ResetButton	To reset the game so the player can play again	Change Text property to "Reset".



Now that you have all the essential properties configured, feel free to change the colours of any components that you want to.

MOVING THE ROCKET

In this game, the user will move the rocket from side to side. This means we will only be changing the X-direction of the rocket sprite. To do this we will use the RocketSprite.Dragged event handler. When the rocket is dragged, we will adjust its X property to be the currentX that we dragged the sprite to.

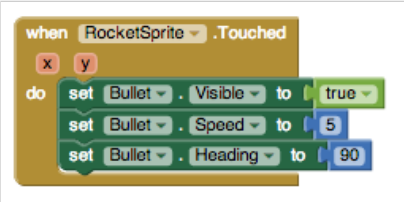
Once you put these blocks together, connect your phone and test this feature out!

PROGRAMMING THE BULLET'S BEHAVIOUR

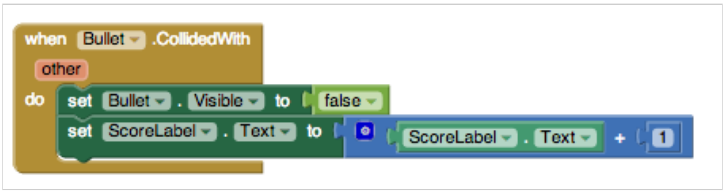
There are several features we want our bullet to have in this game. We want it to shoot from the rocket, collide with the saucer and be invisible after the collision and before being shot.

Let's start by using the Screen1.initialize block. When the screen is initialized, we will program the bullet to be invisible. We do this by setting the bullet's visibility property to False.

Next, we want to make sure that the bullet appears again when we shoot from the rocket. When we touch the rocket, we want the bullet to start heading towards the saucer. We will do this by using the RocketSprite.Touched event handler. When the rocket is touched, we not only want to set the bullet to be visible, but we also want to set the speed and heading of the bullet. Heading is a value from 0 to 360 that indicates what direction the sprite should be moving towards. 0/360 is to the left, 90 is up, 180 is right, and 270 is down. The speed is measured in pixels/sec.

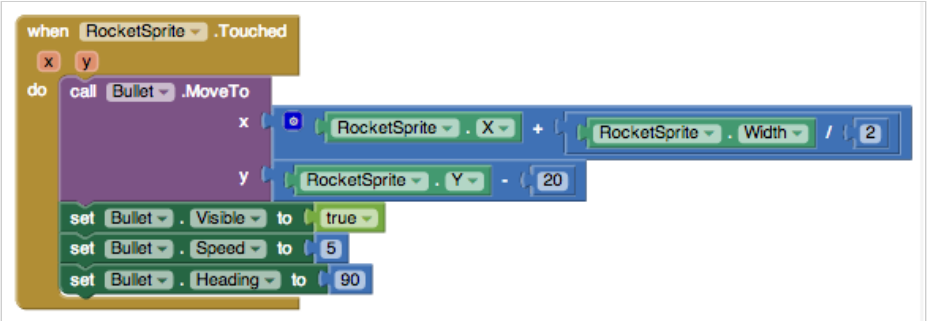


The last thing we need to program is what happens when the bullet hits the saucer. We will use the Bullet.CollidedWith event handler. This event is called whenever the bullet collides with another sprite. Since our rocket sprite is locked into a Y at the bottom of the screen, the bullet will never collide with the rocket and only with the saucer. On collision we want two things to happen. 1. The score should increase by 1. 2. The bullet should become invisible.



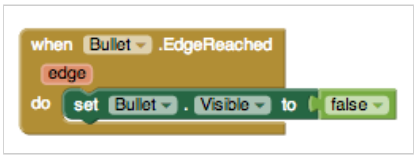
TEST IT OUT!

You may have noticed that once you shoot the bullet, it doesn't appear to let you shoot it again. We need to program the bullet to return to the place in front of the rocket when we shoot it. We can do this using the Bullet.MoveTo block.



Now, test it out!

You may have noticed that if you miss the saucer, the bullet moves to the top of the screen and gets stuck there until you try shooting again. To make the bullet disappear when it hits the top edge of our canvas, we need to use the Bullet.EdgeReached event handler.



PROGRAMMING THE RESET BUTTON

Sometimes, users might want to restart the game and reset their score. When this happens, we need to set the score back to 0.

INCREASING THE DIFFICULTY -- CHANGING THE POSITION OF THE SAUCER

Let's make the game a little more challenging! Now, when the bullet collides with the saucer, let's change the location of the saucer. The saucer will keep the same Y value so we'll only have to change the X. We can do this by using the random block.

To make it even more difficult, we'll also change the position of the saucer when the Timer goes off.

CAN YOU THINK OF ANY IMPROVEMENTS YOU COULD MAKE?

Add sound – e.g. create a sound component – link it to a sound (upload a sound file) then link it to an event. E.g. when the bullet if fired or it collides with the ship. Add an icon to the app.

COMPLETE PROGRAM

Here's the complete SpaceInvaders program: