



Build a balloon pop game!

Part 1

An MIT App Inventor tutorial

Feat. Tim the beaver



App overview: Build a balloon pop game!

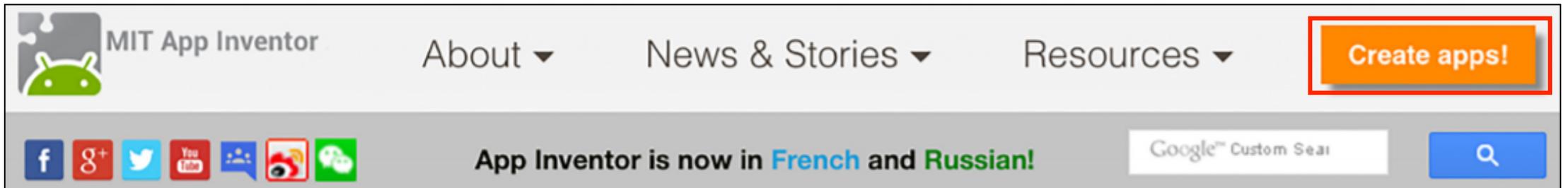
When you are done you and your friends will be able to use this app to play a fun game of pop the balloon!

- We will use *Image Sprite* components—programmable moving images—to create balloons that drop from the sky
- The player will have to “pop”—that is, click on—the balloons before they reach the bottom of the screen
- In this simple version of the app, nothing will happen if the balloons reach the bottom. In “Build a balloon pop game! Part 2”, we will add a “Game over” message and restart button that appear when this happens.



Step 1: Signing in to App Inventor

Click the “Create apps!” button in the menu bar at the top of the MIT App Inventor Hour of Code page.



Step 1 continued

Welcome to MIT App Inventor!

You can either Continue with an Account, and you will be given a Revisit Code to return to the site if you wish.



Continue Without An Account

or

Your Revisit Code: - - -

Enter with Revisit Code

Or you can sign in if you have a Google account. Your projects will be saved with your account id.

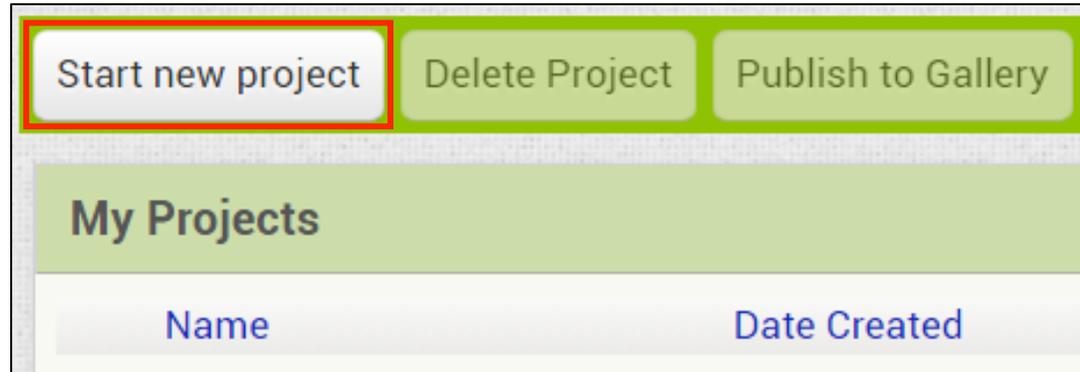


Or sign in with:



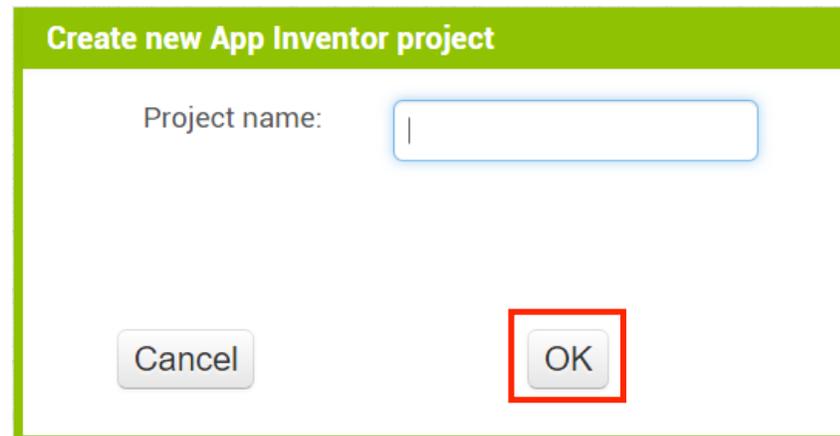
Step 2: Creating a new project

Click “Start a new project” in the upper left corner...

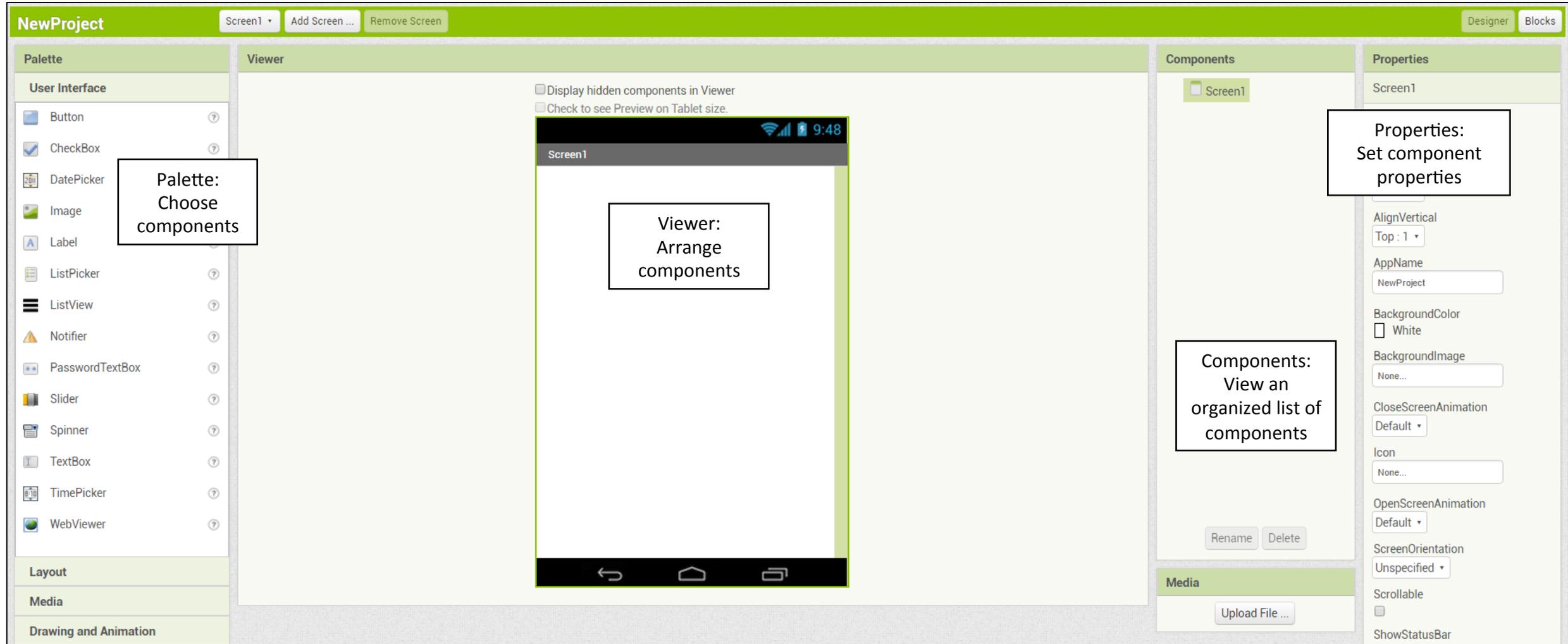


For this tutorial
you can call
your app
“BalloonPop”

...give it a name and click “Ok” to get started!



Step 3: Familiarize yourself with the designer window



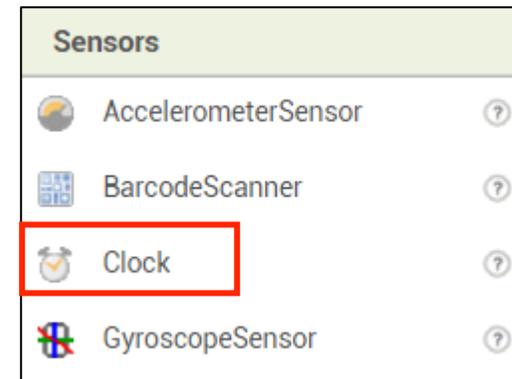
The screenshot shows the MIT App Inventor Designer interface. The top bar includes 'NewProject', 'Screen1', 'Add Screen...', 'Remove Screen', and 'Designer'/'Blocks' tabs. The interface is divided into four main panels:

- Palette:** A list of user interface components such as Button, CheckBox, DatePicker, Image, Label, ListPicker, ListView, Notifier, PasswordTextBox, Slider, Spinner, TextBox, TimePicker, and WebView. A callout box states: "Palette: Choose components".
- Viewer:** A central workspace for arranging components on a mobile device screen. It includes a status bar at the top showing 'Screen1' and a time of 9:48, and an Android navigation bar at the bottom. A callout box states: "Viewer: Arrange components".
- Components:** A panel showing an organized list of components currently on the screen, including 'Screen1'. It has 'Rename' and 'Delete' buttons. A callout box states: "Components: View an organized list of components".
- Properties:** A panel for setting the properties of the selected component. It shows properties for 'Screen1', such as 'AlignVertical' (Top: 1), 'AppName' (NewProject), 'BackgroundColor' (White), 'BackgroundImage' (None...), 'CloseScreenAnimation' (Default), 'Icon' (None...), 'OpenScreenAnimation' (Default), 'ScreenOrientation' (Unspecified), 'Scrollable' (unchecked), and 'ShowStatusBar'. A callout box states: "Properties: Set component properties".

Additional callouts include 'Display hidden components in Viewer' and 'Check to see Preview on Tablet size' in the Viewer panel, and 'Layout', 'Media', and 'Drawing and Animation' sections in the bottom left.

Step 4: Add components!

To build this app you will need five components—a canvas, three image sprites, and a clock. Find these components in the Palette and drag and drop them onto the Viewer.

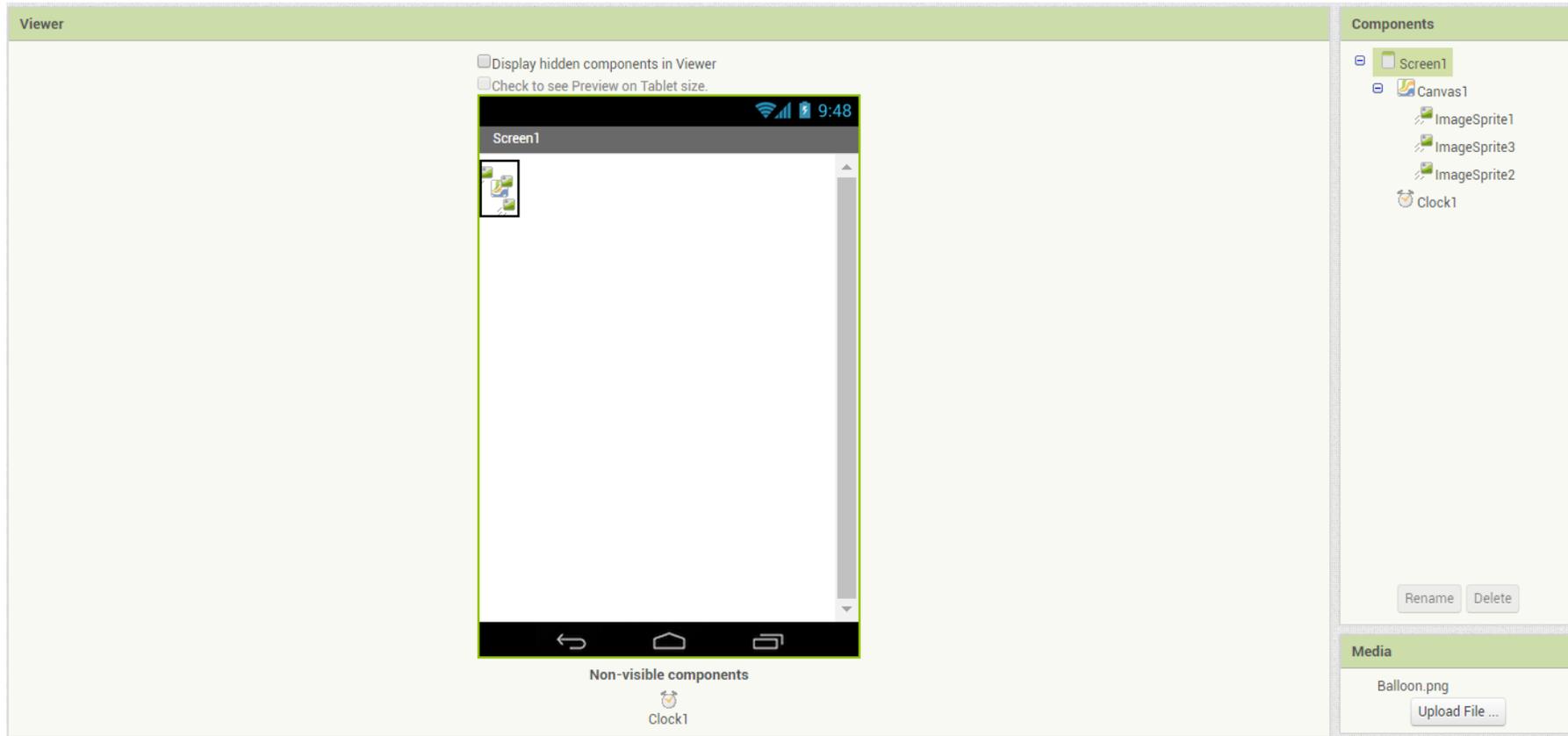


Note that ImageSprites must be placed ON the canvas.

Components
are the
building blocks
of App Inventor
apps!



Your screen should now look like this:



The screenshot displays the MIT App Inventor interface. The central 'Viewer' pane shows a mobile app preview for 'Screen1' with a status bar at the top (9:48) and an Android navigation bar at the bottom. A small icon is visible in the top-left corner of the canvas. Above the preview are two checkboxes: 'Display hidden components in Viewer' and 'Check to see Preview on Tablet size.'. Below the preview is a 'Non-visible components' section containing a 'Clock1' component. To the right, the 'Components' pane lists 'Screen1' containing 'Canvas1', 'ImageSprite1', 'ImageSprite3', 'ImageSprite2', and 'Clock1'. At the bottom of this pane are 'Rename' and 'Delete' buttons. Below the components list is a 'Media' section with 'Balloon.png' and an 'Upload File ...' button.

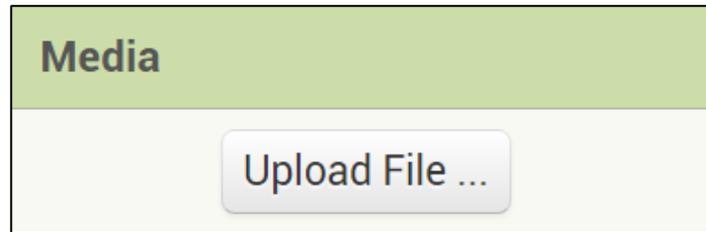
Make sure to drop
the Image Sprites
inside the Canvas!



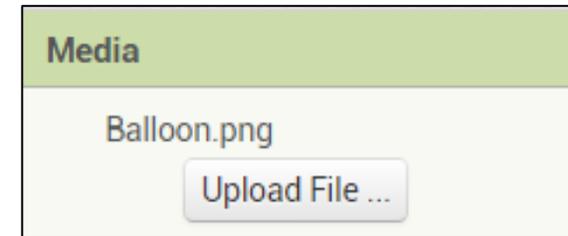


Step 5: Upload media files

To complete this app you will need to download a picture of a balloon for your sprites from [here](#). Then you will need to upload it to the App Inventor server by clicking the upload file button under “Media”



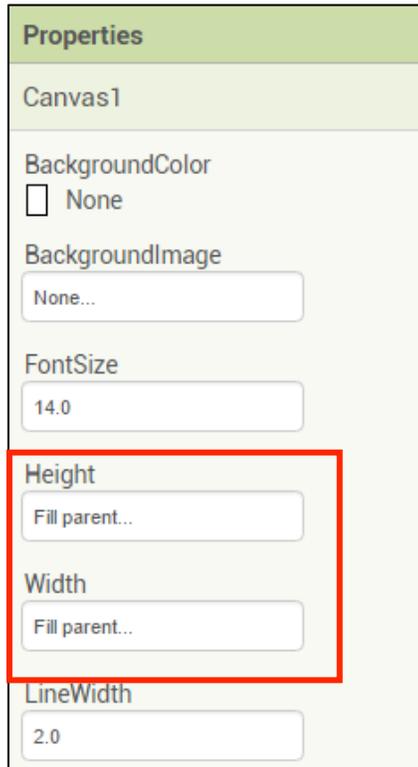
Before upload



After upload

Step 6: Set properties

Now we will change some component properties to start truly building our app!
To view and change a component's properties, find it in the "Components" list and click on it.



Properties

Canvas1

BackgroundColor
 None

BackgroundImage
None...

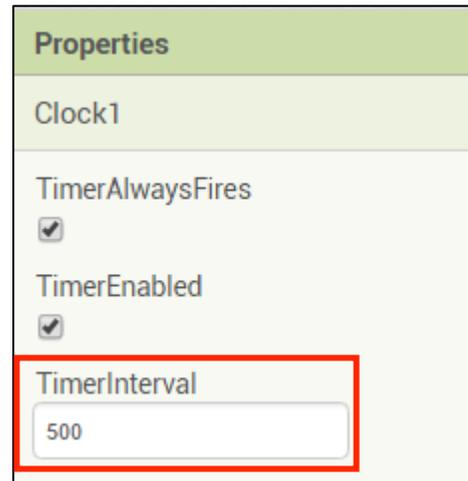
FontSize
14.0

Height
Fill parent...

Width
Fill parent...

LineWidth
2.0

1. Click on Canvas1 and set both the Height and Width to "Fill parent..."



Properties

Clock1

TimerAlwaysFires

TimerEnabled

TimerInterval
500

2. Click on Clock1 and set TimerInterval to 500

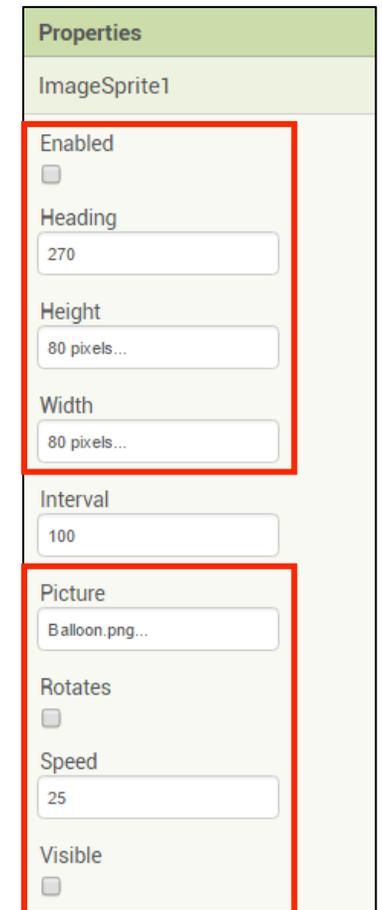
Step 6 continued

Now let's set the properties of our Image Sprites! First, we want to set their X properties so that they spread out across the canvas. Set the X value of ImageSprite1 to 10; the X value of ImageSprite2 to 100; and the X value of ImageSprite3 to 190



Now we need to set some properties that are the same for all three! The example to the right shows ImageSprite1, but all three Image Sprites should have the same properties set.

- Set Heading to 270
- Set Height to 80 pixels
- Set Width to 80 pixels
- Set Picture to Balloon.png
- Set Speed to 25
- Uncheck the boxes for Enabled, Rotates, and Visible



Properties

ImageSprite1

Enabled

Heading 270

Height 80 pixels...

Width 80 pixels...

Interval 100

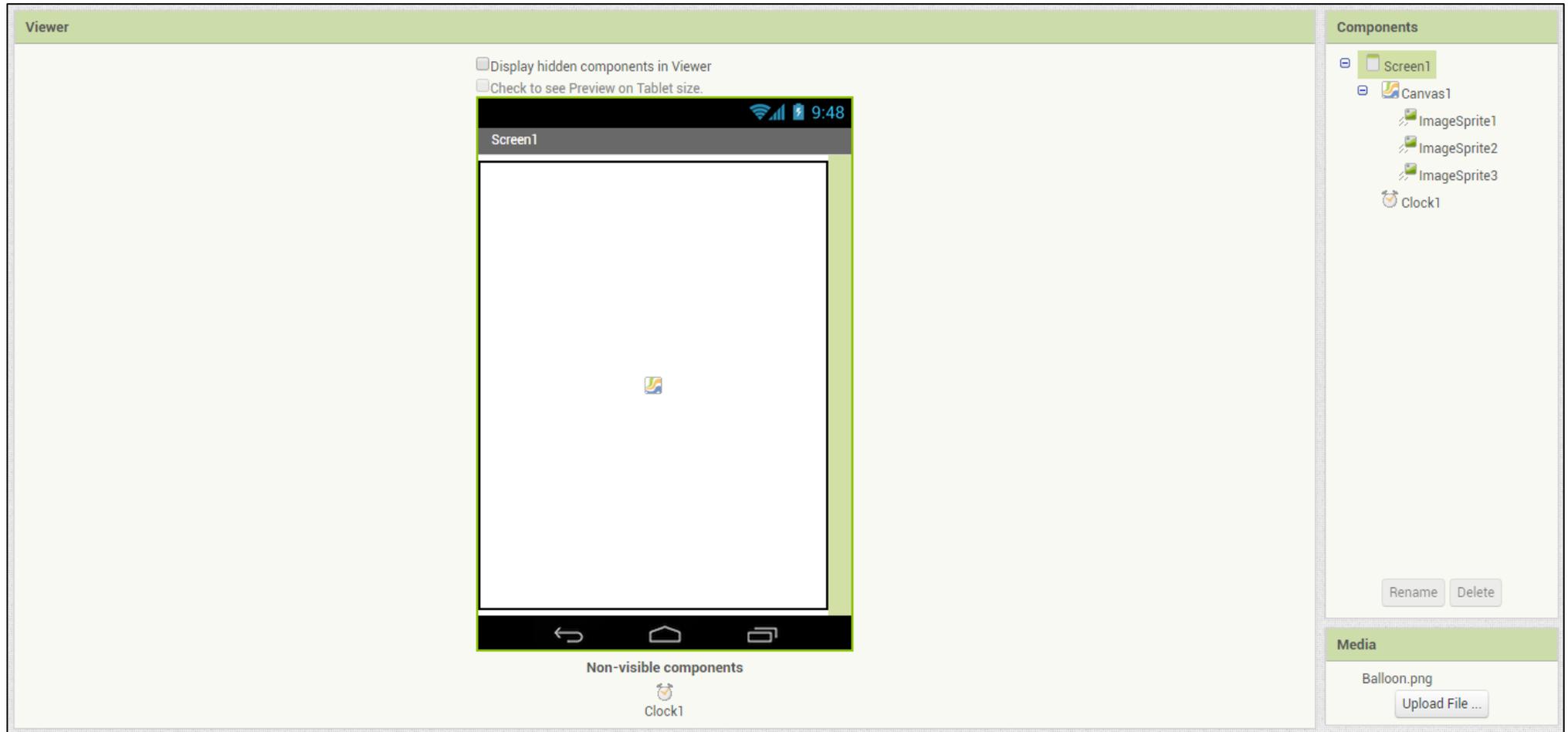
Picture Balloon.png...

Rotates

Speed 25

Visible

Your screen should now look like this:



The screenshot displays the MIT App Inventor interface. The central area is the **Viewer**, which shows a mobile app preview. Above the preview are two checkboxes: Display hidden components in Viewer and Check to see Preview on Tablet size. The preview shows a screen with a black header bar containing the text "Screen1", a status bar with signal, battery, and time (9:48) indicators, a large white content area with a small icon in the center, and a black footer bar with three navigation icons. Below the preview is a section for **Non-visible components**, which includes a clock icon labeled "Clock1".

On the right side, the **Components** panel lists the following components:

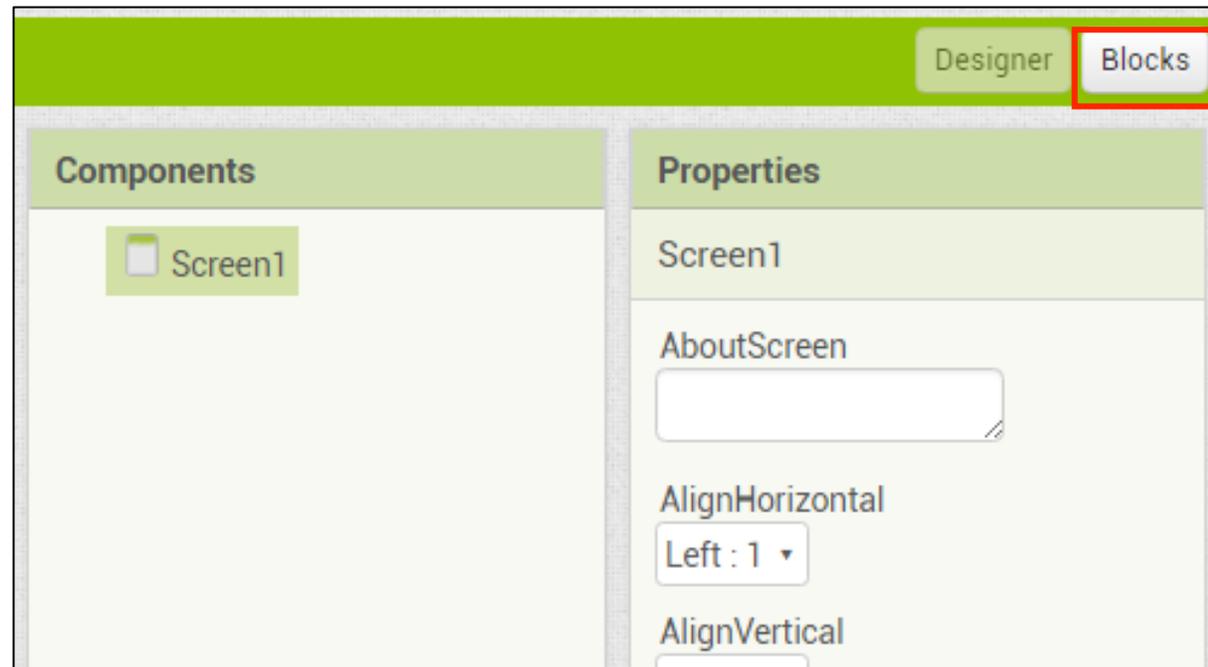
- Screen1
 - Canvas1
 - ImageSprite1
 - ImageSprite2
 - ImageSprite3
 - Clock1

At the bottom of the Components panel are "Rename" and "Delete" buttons.

Below the Components panel is the **Media** section, which includes a "Balloon.png" file and an "Upload File ..." button.

Step 7: Switch to the blocks window to write code!

Now that all components have been added to the app, we will write code to tell the app what to do with them! To do so, switch to the blocks window by clicking the “Blocks” button in the upper right corner.





Step 7 continued: Get to know the blocks window

TakePictureWithTim Screen1 Add Screen ... Remove Screen Designer Blocks

Blocks

- Built-in
 - Control
 - Logic
 - Math
 - Text
 - Lists
 - Colors
 - Variables
 - Procedures
- Screen1
 - Canvas1
 - Tim
 - CameraButton
 - Camera1
- Any component

Rename Delete

Media

- TimTheBeaver.png
- TimTheBeaver1.png
- Upload File ...

Viewer

Built-in blocks:
These are always available and handle things like math, text logic, and control

Component blocks:
These correspond to the components you've added to your app

An example of two assembled blocks

```
when CameraButton Click  
do call Camera1 TakePicture
```

Viewer:
Where you assemble the blocks into a program

Show Warnings

Step 8: Start coding!

Every half a second (or 500 milliseconds), Clock1's timer will go off—that's because we set the TimerInterval property to 500! When this happens, we would like one of our three Image Sprites to appear and begin falling to the bottom of the screen. To do this, we have to choose a Sprite, and we would like that choice to be random. App Inventor lets us do this!

1. First, click on Variables in the Blocks menu and drag out a block that looks like this: 
2. Change "name" to "randomNumber"
3. Click on Math in the blocks menu and drag out a block that looks like this: 
4. Change "0" to "1" and then click it onto the "initialize global name" block

Your final result should look like this:



initialize global randomNumber to 1

Variables
store
information
—like a
number!



Step 8 continued

1. Next, click on Clock1 and drag out a block that looks like this:



2. Click on Variables again and drag out a block that looks like this:



3. Choose "global randomNumber" from the drop-down list

4. Click on Math again and drag out a block that looks like this:



5. Change "100" to "3" and lock the block into place with the "set global randomNumber" block to get this:



6. Snap these two into place inside the "when Clock1.Timer" block

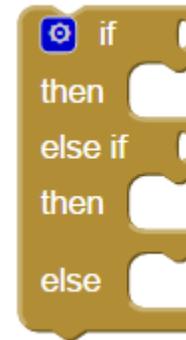
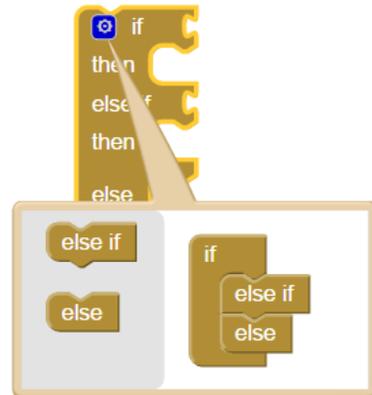
Your final result should look like



Step 8 continued

We would like the app to do different things depending on which number was chosen by App Inventor —because we asked it for a random number between one and three, it could have given us a 1,2, or 3.

1. Click on Control and drag out a block that looks like this:
2. Click on the “if, then” block’s settings button—the square blue one in the corner
3. Drag out an “else if” and an “else” block from the left and lock them into place in the “if” block on the right. The “if, then” block should now look like this:



Make sure your “if then” block looks exactly like the one shown



Step 8 continued

Now we need to fill in our “if, then” block. If App Inventor has given us the number 1, we want to make ImageSprite1 appear; if it has given us the number 2, we want to make ImageSprite2 appear; otherwise, it must have given us the number 3, and so we want to make ImageSprite3 appear.

1. Click on Logic and drag out a block that looks like this: 
2. Click on Variables and drag out a block that looks like this: 
3. Choose “global randomNumber” from the dropdown list and then click this block into place in the Logic block’s first space.
4. Click on Math and choose a block that looks like this: 
5. Change “0” to “1” and click it into place in the Logic block’s second space

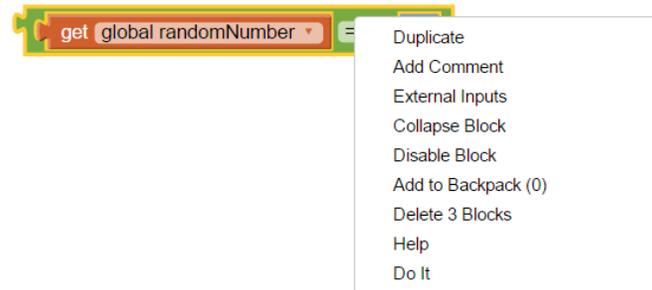
Your final result should look like



Step 8 continued

You have written code that checks if the randomNumber variable is equal to 1. With just a small tweak it will be able to check if randomNumber is equal to 2.

1. Copy and paste the block of code you just created. To do so, right click on it and choose “Duplicate from the drop-down list”



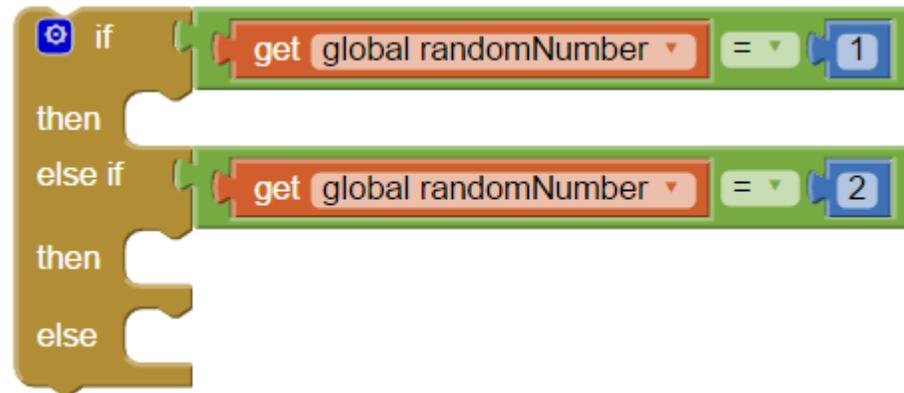
2. On the duplicate block, change the “1” to a “2”

Your final result should look like



Step 8 continued

Now, click the two blocks you've just made into place in the "if, then" block, positioned as shown below:



Step 8 continued

Now we will tell the app what to do, depending on the random number that is selected.

1. Click on ImageSprite1 and drag out a block that looks like this: 
2. Click on Logic and drag out a True block. Click it into place with the “set ImageSprite1.Enabled” to get this: 
3. Duplicate the block as you did before, by right clicking. For the duplicated block, replace “Enabled” with “Visible” by clicking on the second dropdown list.
4. Lock the two blocks together.

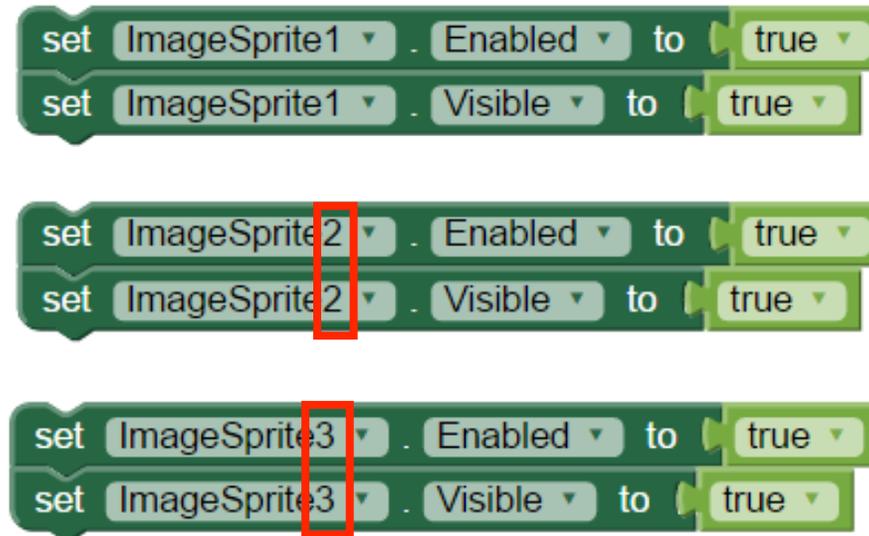
Your final result should look like this:



Step 8 continued

1. Take the group of two blocks and duplicate it twice (so that you get three copies).
2. Keep the original as it was, but change “ImageSprite1” to “ImageSprite2” in one of the copies, and to “ImageSprite3” in the other.

Your final result should look like this:



Step 8 continued

Now piece it all together! Place the three groups of blocks you created in the previous slide inside the “if, then” block as shown below, and put the “if, then” block inside the “when Clock1.Timer” block.

```
when Clock1.Timer
do
  set global randomNumber to random integer from 1 to 3
  if
    get global randomNumber = 1
  then
    set ImageSprite1.Enabled to true
    set ImageSprite1.Visible to true
  else if
    get global randomNumber = 2
  then
    set ImageSprite2.Enabled to true
    set ImageSprite2.Visible to true
  else
    set ImageSprite3.Enabled to true
    set ImageSprite3.Visible to true
```

Double-check to make sure everything is in the right place





The code you have written so far should look like this:

```
initialize global randomNumber to 1

when Clock1.Timer
do
  set global randomNumber to random integer from 1 to 3
  if get global randomNumber = 1
  then
    set ImageSprite1.Enabled to true
    set ImageSprite1.Visible to true
  else if get global randomNumber = 2
  then
    set ImageSprite2.Enabled to true
    set ImageSprite2.Visible to true
  else
    set ImageSprite3.Enabled to true
    set ImageSprite3.Visible to true
```



Step 9: Testing!

Great job! You just wrote code in App Inventor! But does your code do what we want it to? To find out, we're going to have to learn how to test our app...



Step 9 continued: Connect to your phone

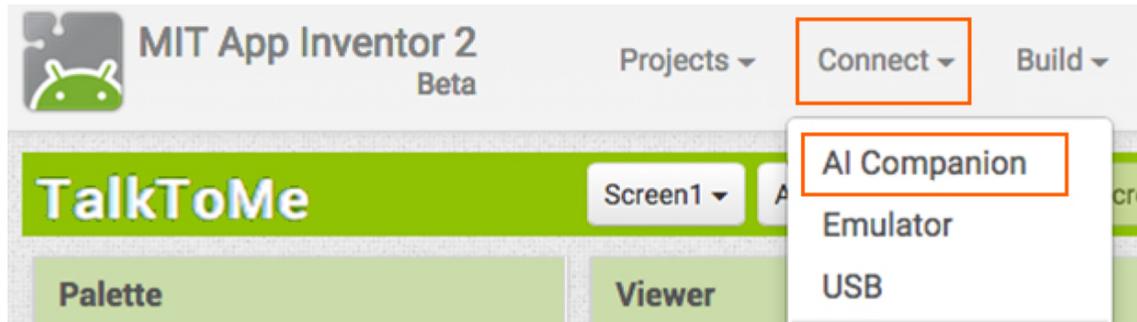
In order to test your app, you will need an Android phone with the MIT AI2 Companion app installed. To download the Companion from the app store, scan the QR code below or search directly for “MIT AI2 Companion” on the Google Play Store, <https://play.google.com/store>.



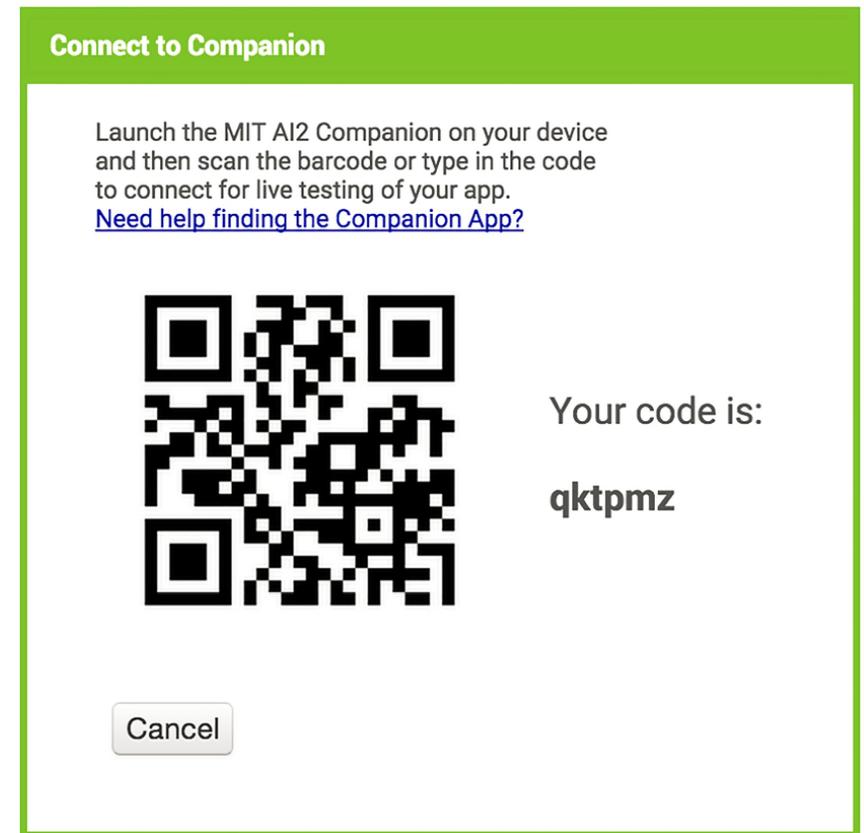
NOTE: If you do not have an android phone, or if you are unable to download the Companion app, you can still use App Inventor using an emulator. Visit: <http://appinventor.mit.edu/explore/ai2/setup.html> and follow the instructions under Option 2.

Step 9 continued

To connect to the AI2 Companion app, first choose “AI Companion” from the “Connect” drop down menu in the App Inventor site.

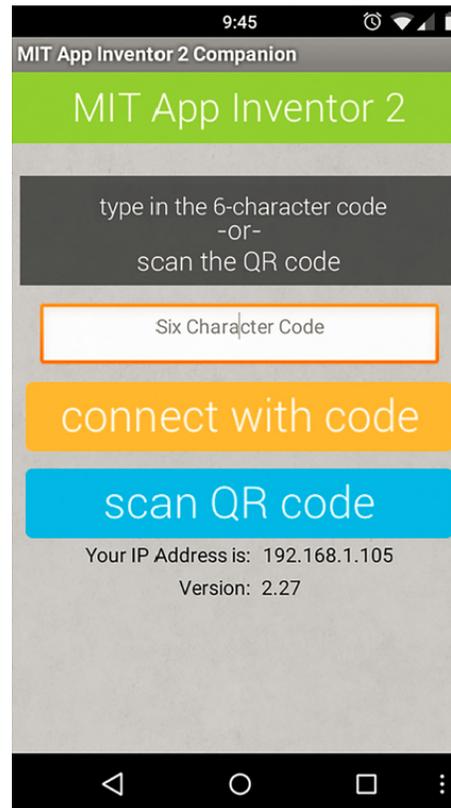


A QR code and 6-letter code will pop up.



Step 9 continued: Open the Companion app

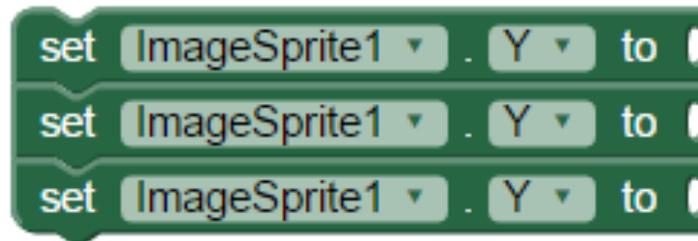
Open the AI2 Companion app. You can then either input the 6-letter code or scan the QR code to connect.



Step 10: More coding!

Okay, great, we have balloons that appear randomly and fall to the bottom of the phone, but we would like for something to happen when we touch them.

1. Click on ImageSprite1 and drag out a block that looks like this:
2. Also under ImageSprite1, click on a block that looks like this:
3. Duplicate it twice to get three copies, and click them together to form a stack:

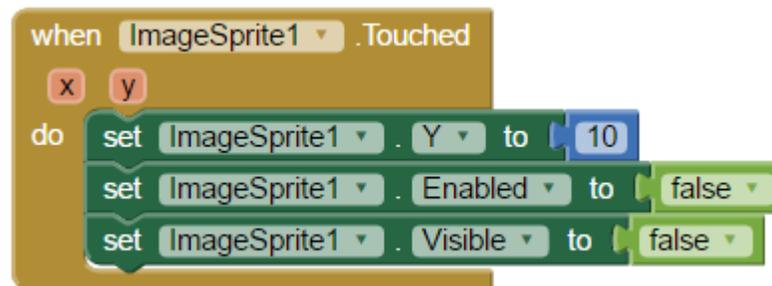


Step 10 continued

When each balloon is touched, we want to return it to its original state. This means resetting its Y, Visible, and Enabled properties.

1. Grab the stack of blocks you made in the last slide. Change the Y of one block to Visible using the dropdown menu, and the other Y to Enabled.
2. Set ImageSprite1.Y to 10, and ImageSprite1.Visible and ImageSprite1.Enabled to False.
3. Click this stack into the “when ImageSprite1.Touched” block.

Your final result should look like this:



Step 10 continued

The app should do the same thing when ImageSprite2 or ImageSprite3 are touched. That means we can reuse the same stack of blocks and just change the sprite name.

1. Duplicate the when ImageSprite1.Touched group of blocks twice (to get three copies)
2. For one of the group of blocks, change all mention of “ImageSprite1” to “ImageSprite2”
3. For the other, change all mention of “ImageSprite1” to “ImageSprite3”



The code you have just added should look like this:

```
when ImageSprite1 .Touched
  x y
  do
    set ImageSprite1 . Y to 10
    set ImageSprite1 . Enabled to false
    set ImageSprite1 . Visible to false

when ImageSprite2 .Touched
  x y
  do
    set ImageSprite2 . Y to 10
    set ImageSprite2 . Enabled to false
    set ImageSprite2 . Visible to false

when ImageSprite3 .Touched
  x y
  do
    set ImageSprite3 . Y to 10
    set ImageSprite3 . Enabled to false
    set ImageSprite3 . Visible to false
```

Step 11: Testing and debugging!

Awesome! You're all done coding your app. Open the App Inventor Companion app and make sure everything is working properly. Remember, your app should:

- Reveal a new balloon at the top of the canvas every half a second
- Have that balloon drop to the floor
- Remove the balloon if the user clicks on it

If you want to keep building, check out ways to extend this app in Balloon Pop Part 2!





MIT App Inventor



Thanks for coding with us!
You've done a great job. Check
out more tutorials at
[http://appinventor.mit.edu/
explore/hour-of-code.html](http://appinventor.mit.edu/explore/hour-of-code.html)