

“BallBounce” – A simple game app

Learn about animation in App Inventor by making a Ball (a sprite) bounce around the screen (or a Canvas)

1

Open **Google Chrome** browser and type the URL <https://code.appinventor.mit.edu> in the **address bar** of the browser

Following Screen appears in response of the step 1

Welcome to MIT App Inventor!

Continue Without An Account

or

Your Revisit Code: - - -

2

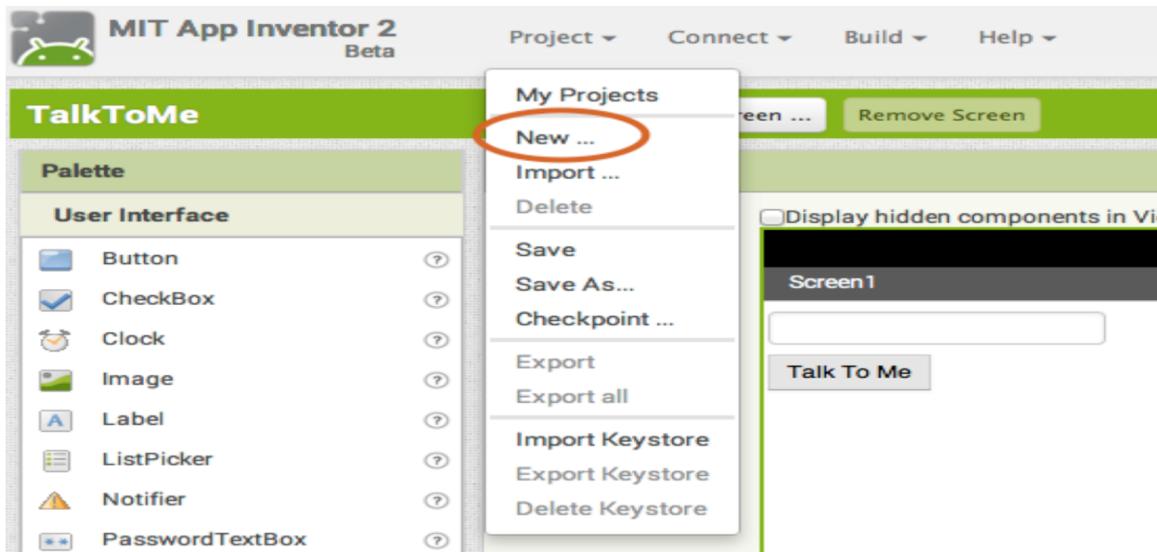
Click “Enter with Revisit Code” after entering your code.



3 Start a New Project

Start a New Project

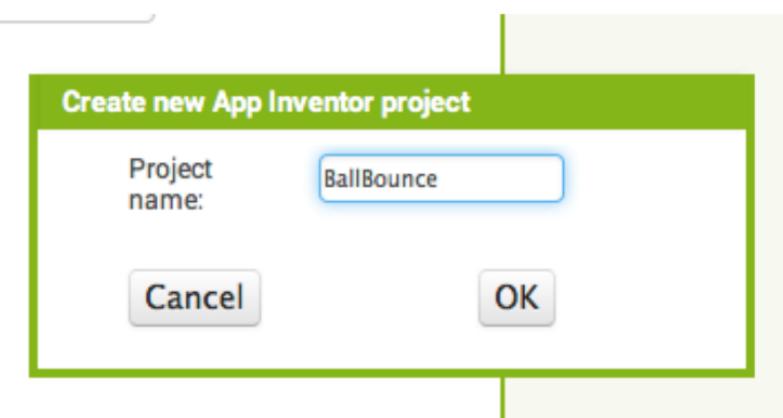
If you have another project open, go to My Projects menu and choose New Project.



4

Name the Project

Call it something like "BallBounce". Remember, no spaces. But underscores are OK.

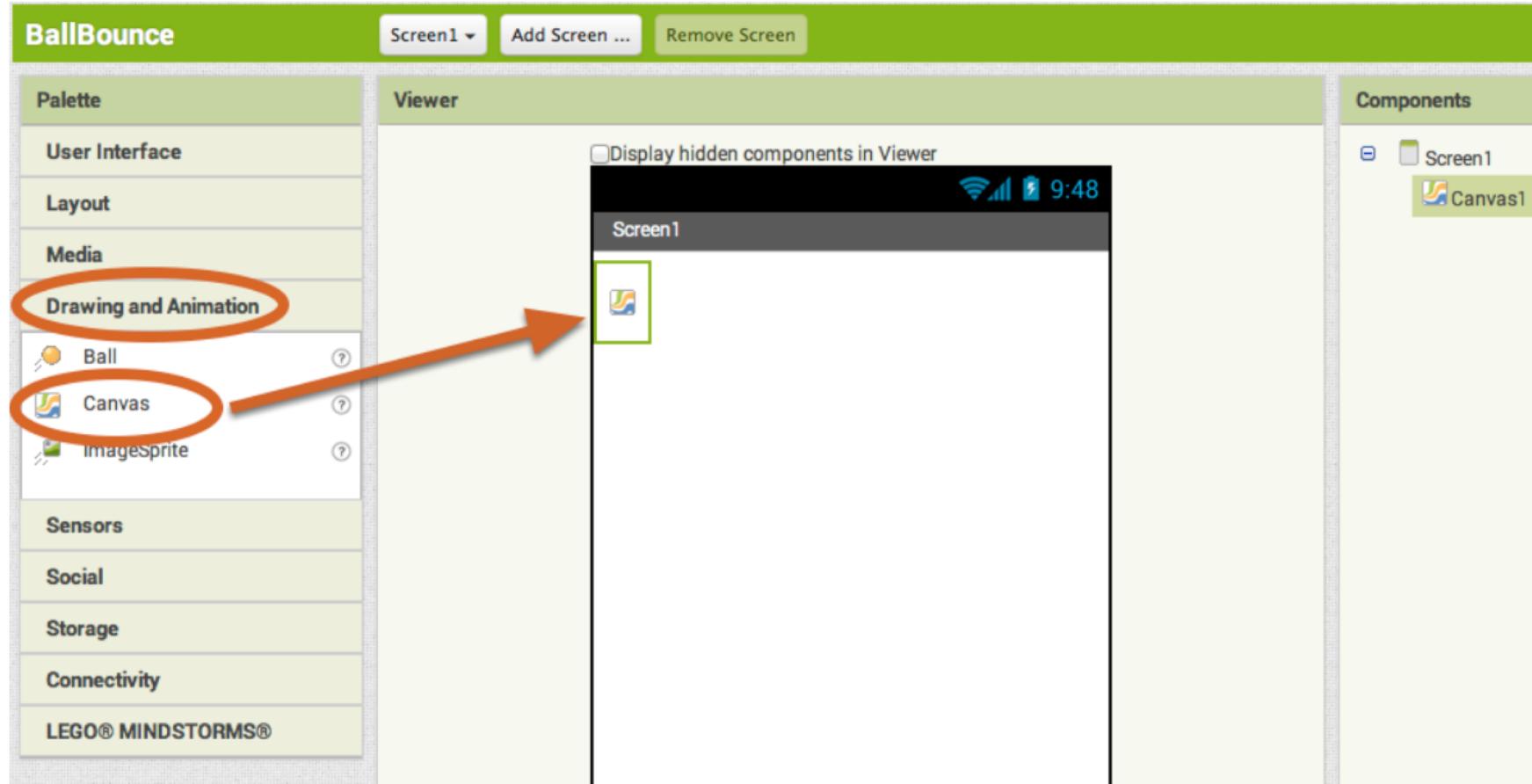




5

Add a Canvas

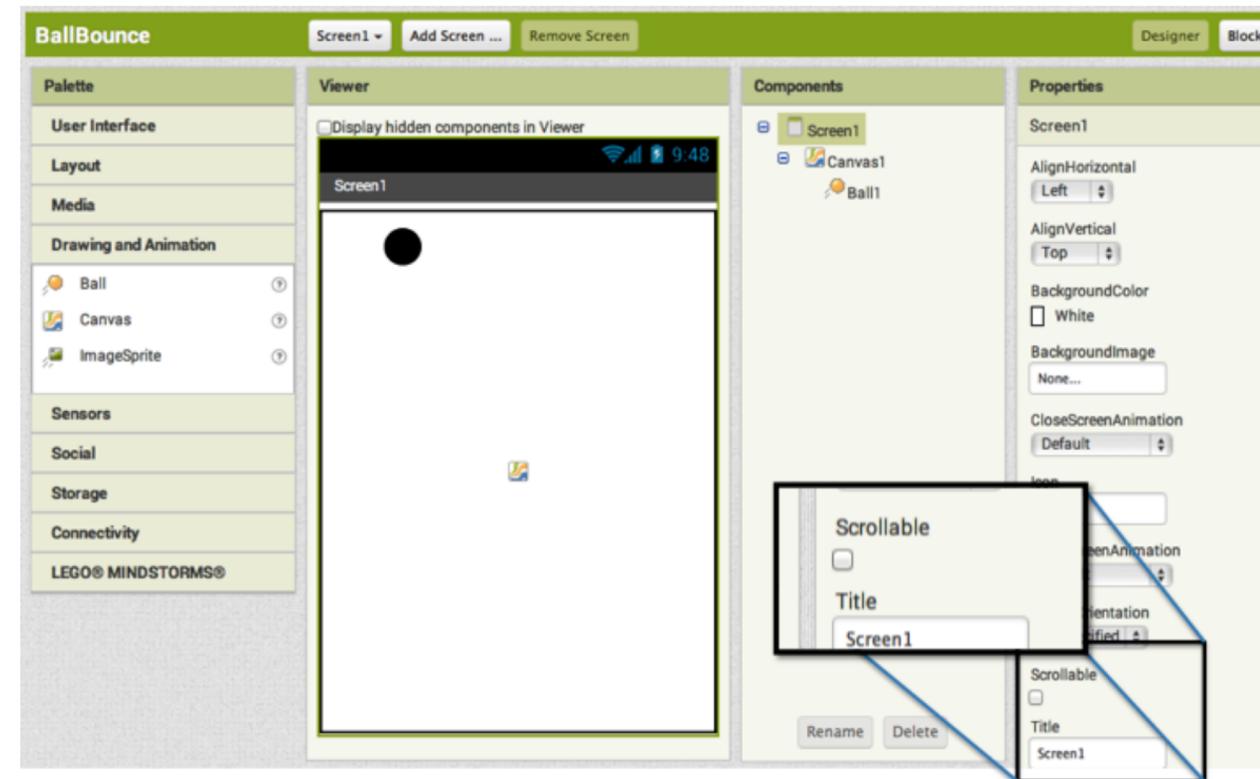
From the **Drawing and Animation drawer**, drag out a **Canvas component** and drop it onto the viewer.



6

Set the Screen so that it does not Scroll

The default setting for App Inventor is that the screen of your app will be "*scrollable*", which means that the user interface can go beyond the limit of the screen and the user can scroll down by swiping their finger (like scrolling on a web page). When you are using a Canvas, you have to *turn off the "Scrollable" setting (UNCHECK THE BOX)* so that the screen does not scroll. This will allow you to make the Canvas to fill up the whole screen.

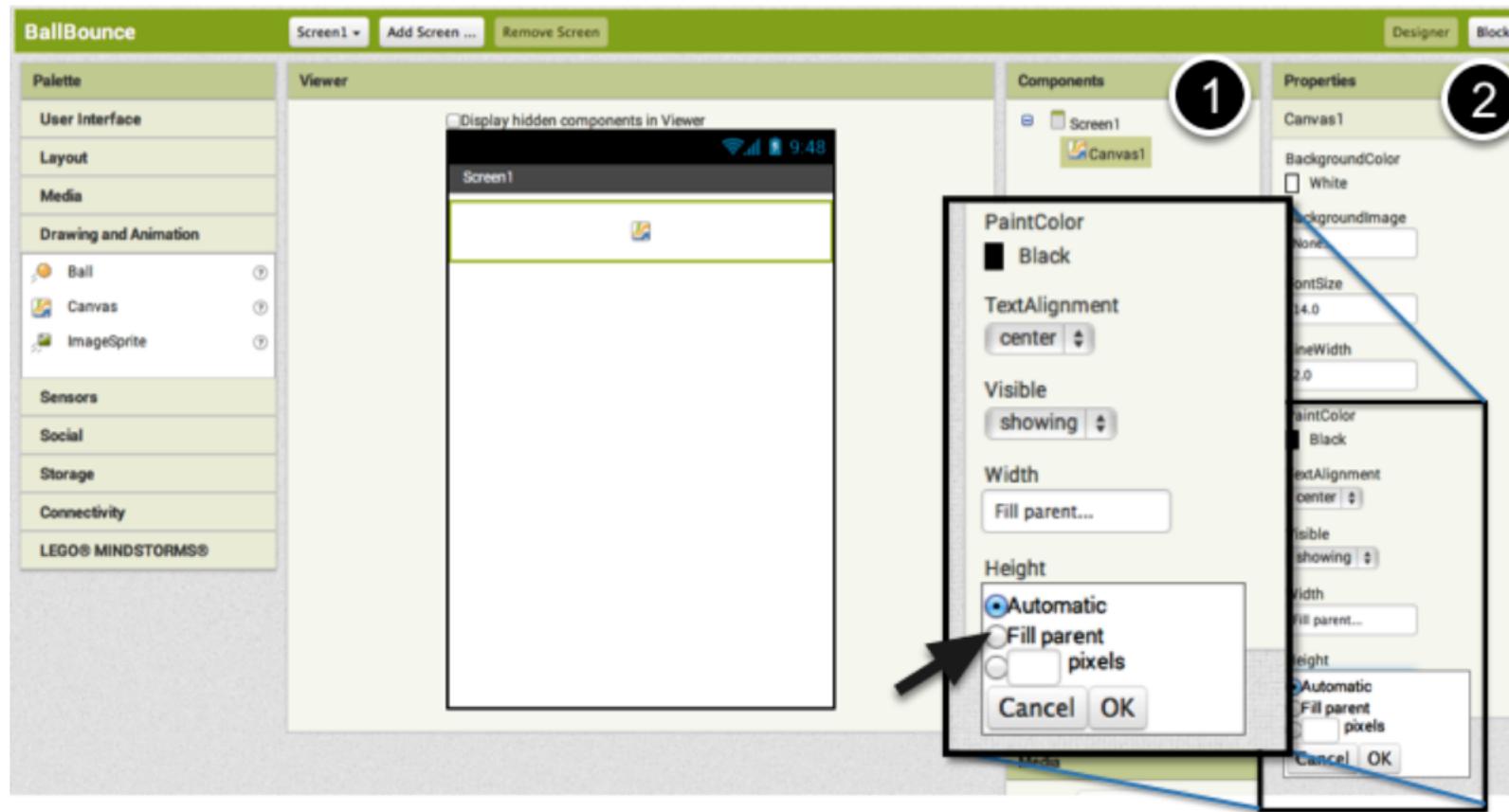




7

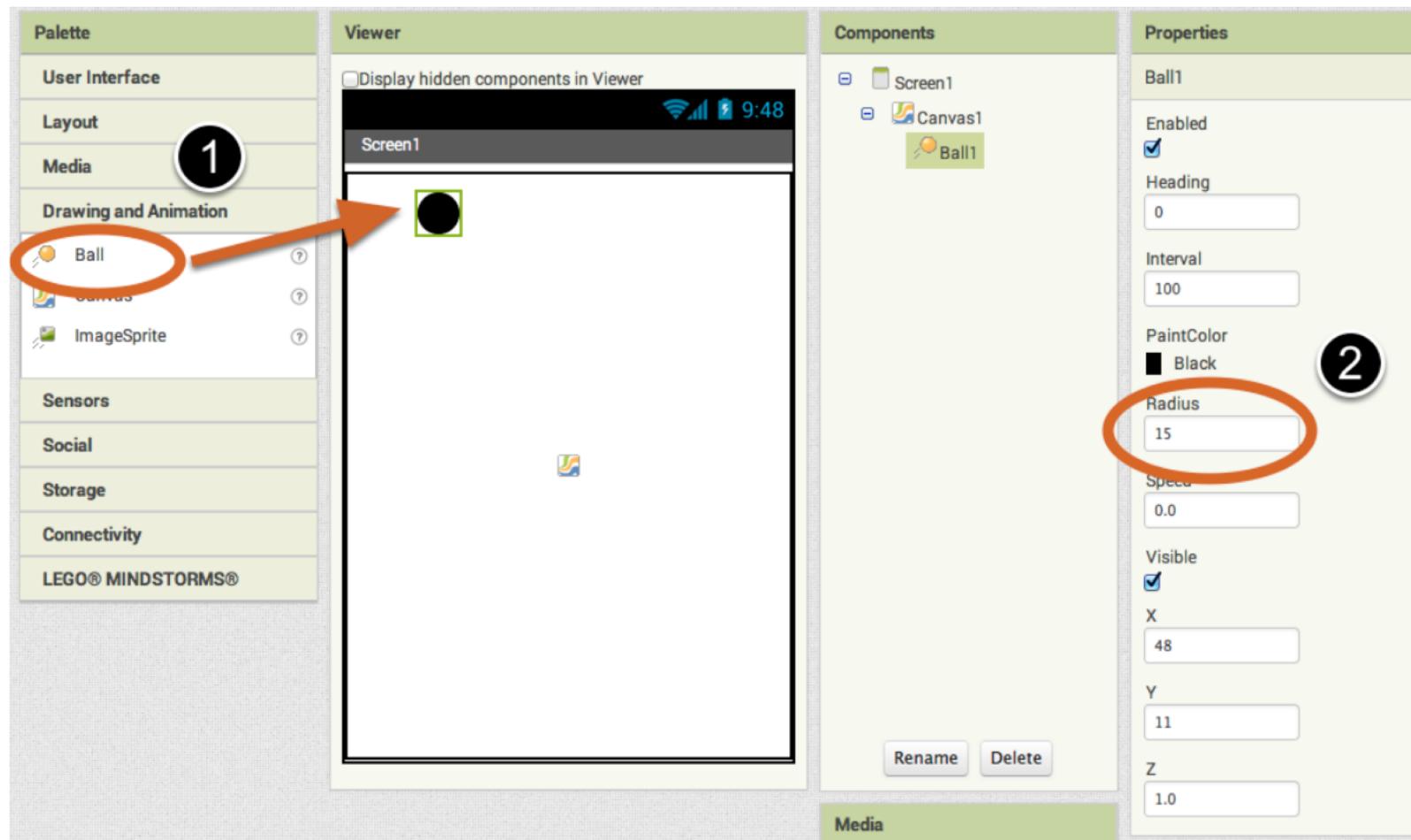
Change the Height and Width of the Canvas to “*Fill Parent*”

Make sure the Canvas component is selected (#1) so that its properties show up in the Properties Pane (#2). Down at the bottom, *set the Height property to “Fill Parent”*. Do the same with the *Width property*.



8 Add a Ball

Now that we have a Canvas in place, we can add a Ball Sprite. This can also be found in the **Drawing and Animation drawer**. Drag out a **Ball component** and drop it onto the Canvas (#1). If you'd like the ball to show up better, you can change its **Radius** property in the Properties pane (#2).

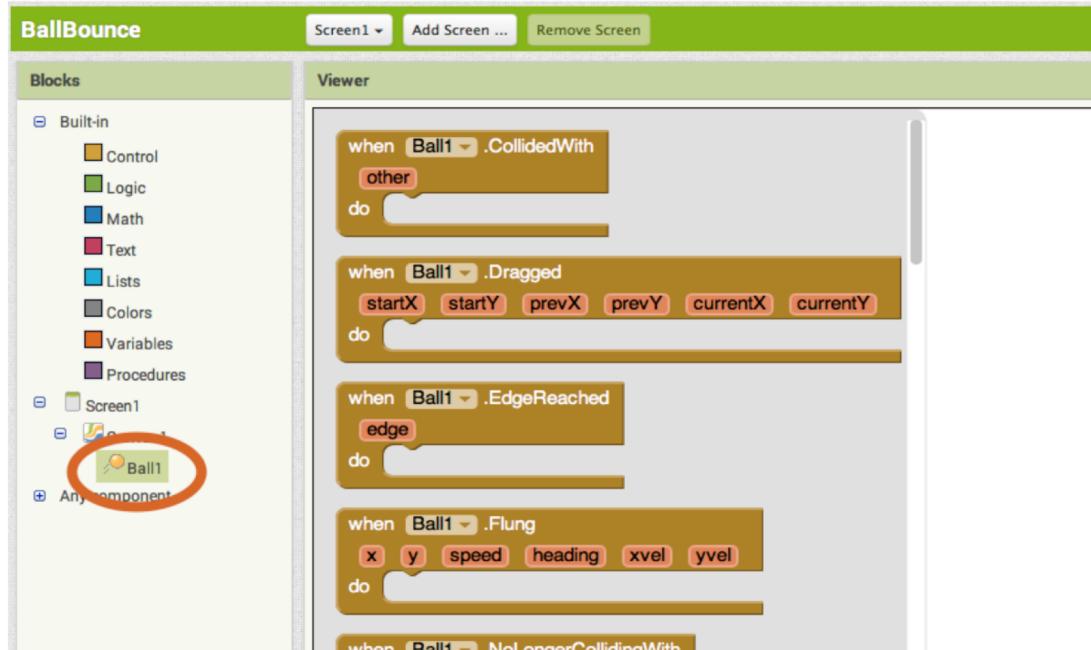


9 Switch to Blocks Editor



10

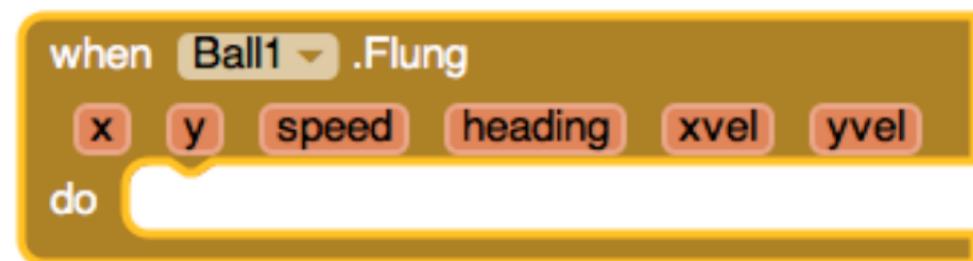
Open the Ball1 Drawer to view the Ball's blocks



11

Drag out the Flung Event Handler

Choose the block **when Ball1.Flung** and drag-and-drop it onto the workspace. Flung refers to the user making a "Fling gesture" with his/her finger to "fling" the ball. Fling is a gesture like what a golf club does, not like how you launch Angry Birds! In App Inventor, the event handler for that type of gesture is called *when Flung*.



What is Fling Gesture?

Drag and **fling** may seem similar but drag is the type of scrolling that occurs when a user drags their finger across the touchscreen, while a **fling** gesture occurs when the user drags and then lifts their finger quickly.

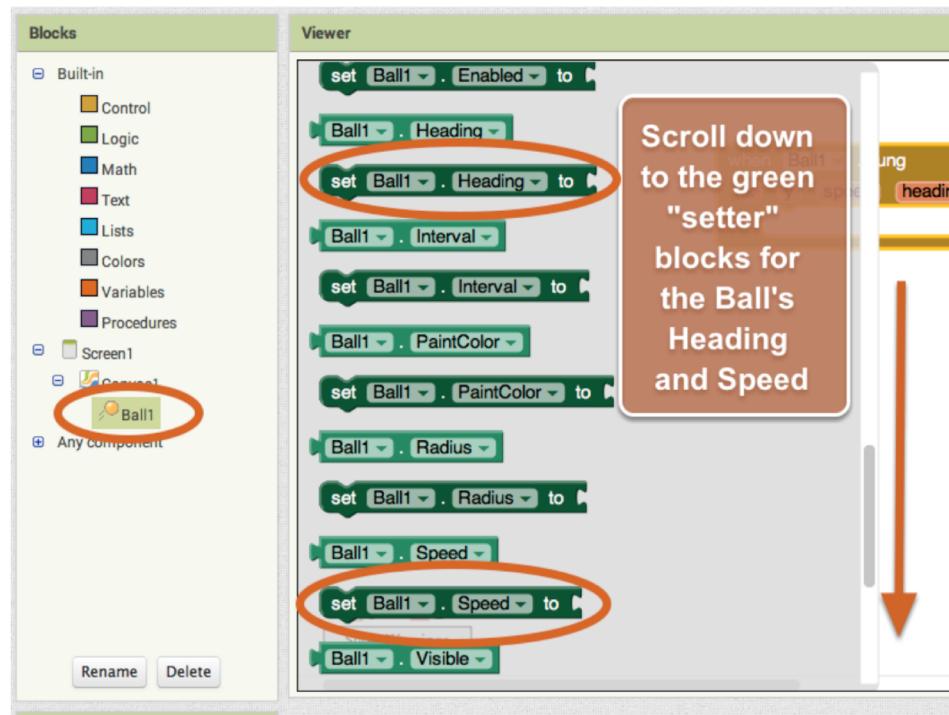
MUST
READ



12

Set the Ball's Heading and Speed. First get the setter blocks.

Open the Ball drawer and scroll down in the list of blocks to get the **set Ball1.Heading** and **set Ball1.Speed** blocks



13

Plug the set Ball1.Speed and set Ball1.Heading into Fling event handler

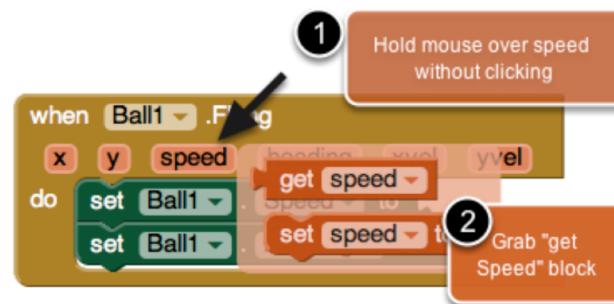




14

Set the Ball's speed to be the same as the Fling gesture's speed

Mouse over the "speed" parameter of the **when Ball1.Flung** event handler. The get and set blocks for the speed of the fling will pop up. Grab the **get speed** block and plug that into the **set Ball1.Speed** block.



15

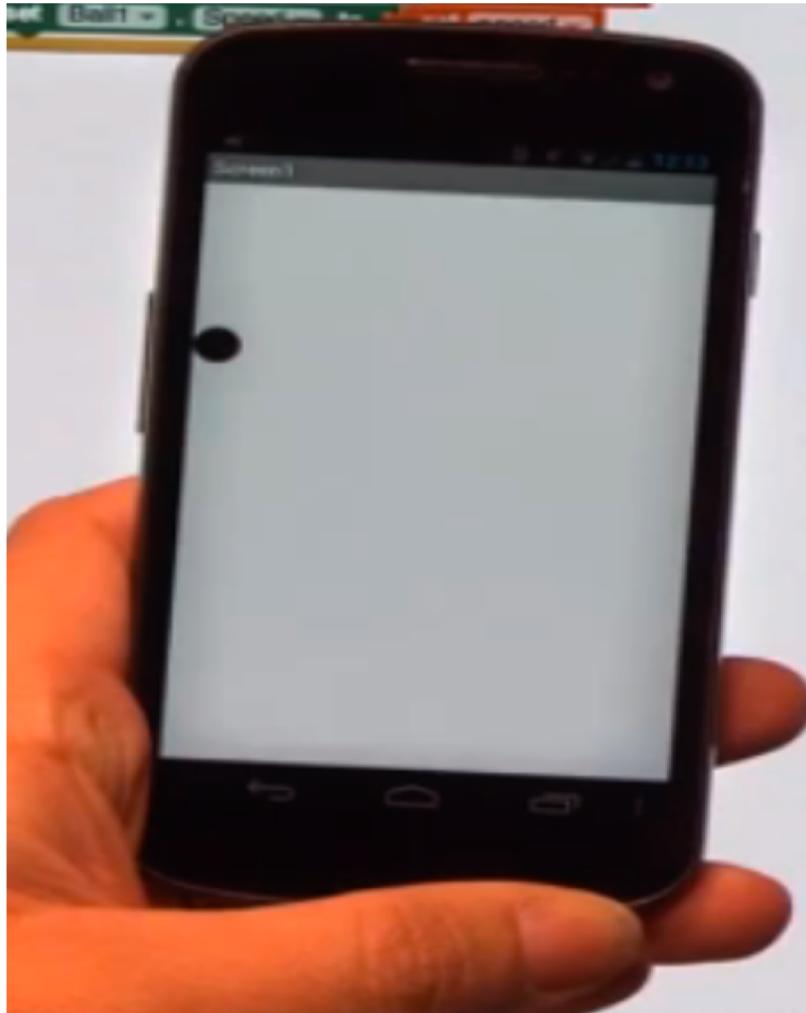
Set the Ball's heading to be the same as the Fling gesture's heading

Do the same for the Ball's heading. Mouse over the **heading** parameter and you'll see the **get heading** block appear. Grab that block, and click it into the **set Ball1.Heading** block.



16

Test it out



17

Why does the Ball get stuck on the side of the screen?

After flinging your ball across the screen, you probably noticed that it got stuck on the side. This is because the ball's heading has not changed even though it hit the side of the canvas. To make the ball "**bounce**" off the edge of the screen, we can program in a new event handler called "**When Edge Reached**".



18 Add an Edge Reached Event

Go into the Ball1 drawer and pull out a **when Ball1.EdgeReached do** event.

The screenshot shows the MIT App Inventor interface with the Blocks palette on the left and the Viewer on the right. The Ball1 component is selected in the drawer. In the viewer, several event blocks are listed:

- when Ball1.CollidedWith other do [empty slot]
- when Ball1.Dragged startX startY prevX prevY currentX currentY do [empty slot]
- when Ball1.EdgeReached edge do [empty slot] (this block is circled in red)
- when Ball1.moving x y speed heading xvel yvel do [empty slot]
- when Ball1.NoLongerCollidingWith other do [empty slot]

Below the blocks are 'Rename' and 'Delete' buttons.

19

Go back into the Ball1 drawer and pull out a **Ball.Bounce** block

The screenshot shows the MIT App Inventor interface with the Blocks palette on the left and the Viewer on the right. The Ball1 component is selected in the drawer. In the viewer, several event blocks are listed:

- when Ball1.TouchDown x y do [empty slot]
- when Ball1.TouchUp x y do [empty slot]
- when Ball1.Touched x y do [empty slot]
- call Ball1.Bounce edge (this block is circled in red) (the 'Ball.Bounce' block is shown in the viewer with an orange arrow pointing to it)
- call Ball1.CollidingWith other



20

Add the edge value for the Ball.Bounce block

The **Ball.Bounce** method needs an edge argument. Notice that the **Ball1.EdgeReached** event has an "edge" as a parameter. We can take the **get edge** block from that argument and plug it into the call **Ball1.Bounce** method. Grab the **get edge** block by mouse over (hover your mouse pointer over) the "edge" parameter on the **when Ball1.EdgeReached** block.

21

Final blocks should look like this. Now test it out!

