

“DigitalDoodle” – Drawing App

Draw a line on the screen as the user drags a finger around.

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

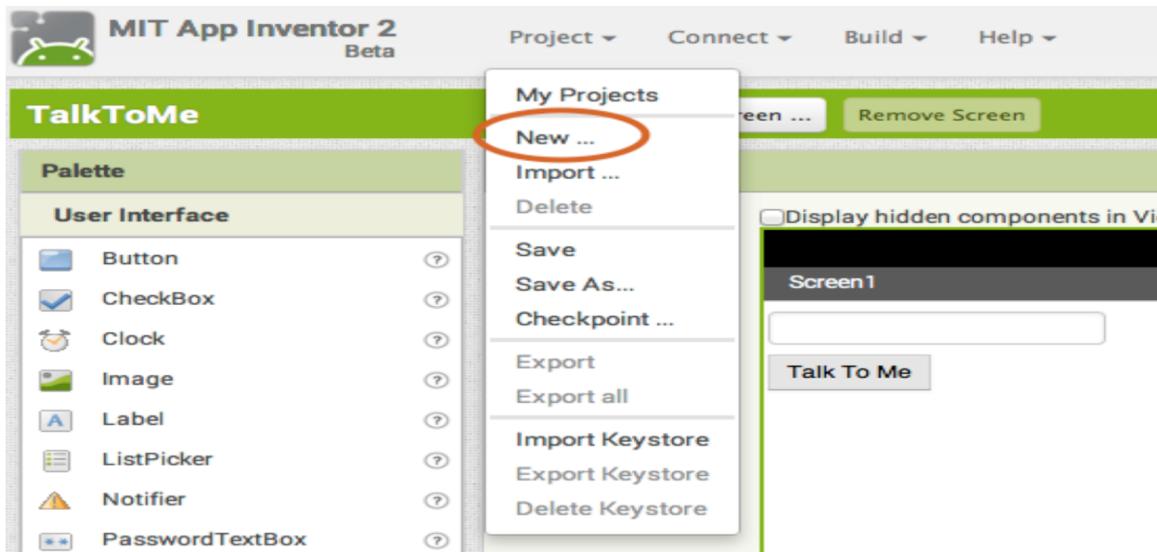
Click “Enter with Revisit Code” after entering your code.



3 Start a New Project

Start a New Project

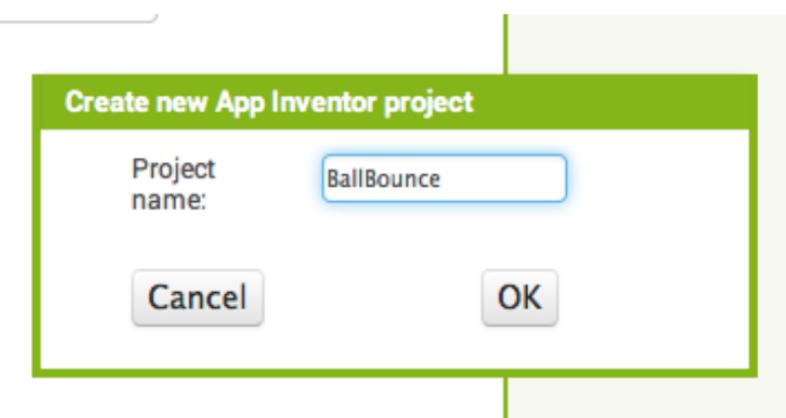
If you have another project open, go to My Projects menu and choose New Project.



4

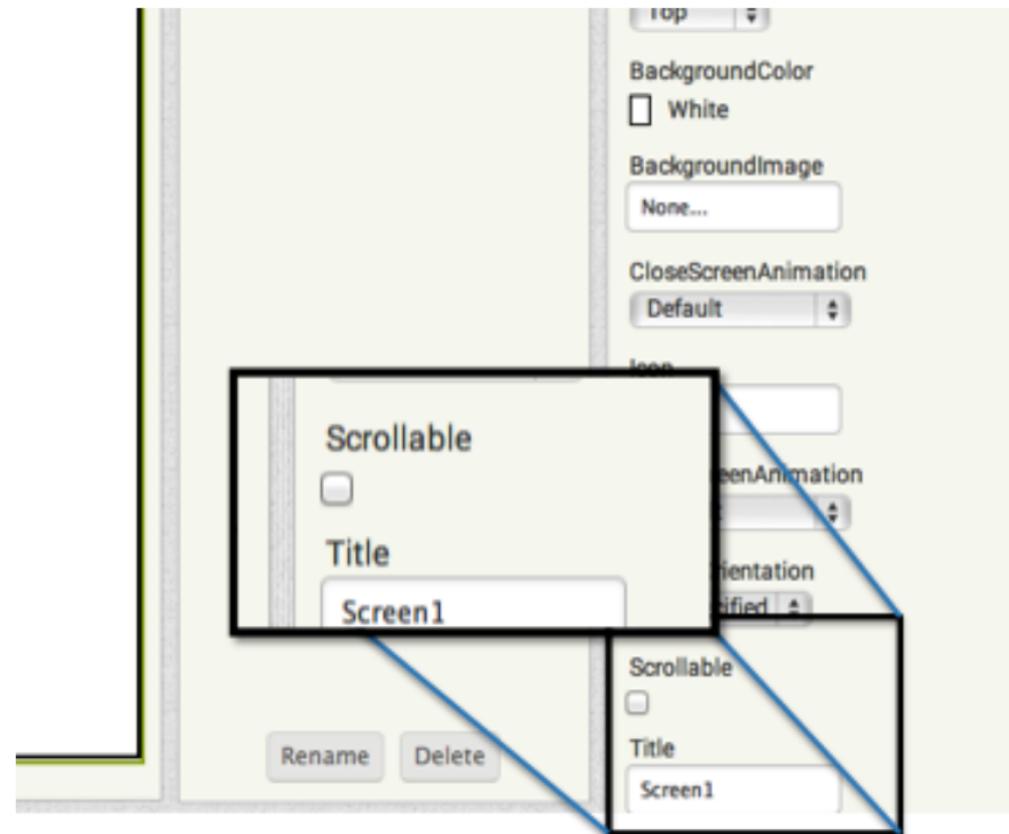
Name the Project

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



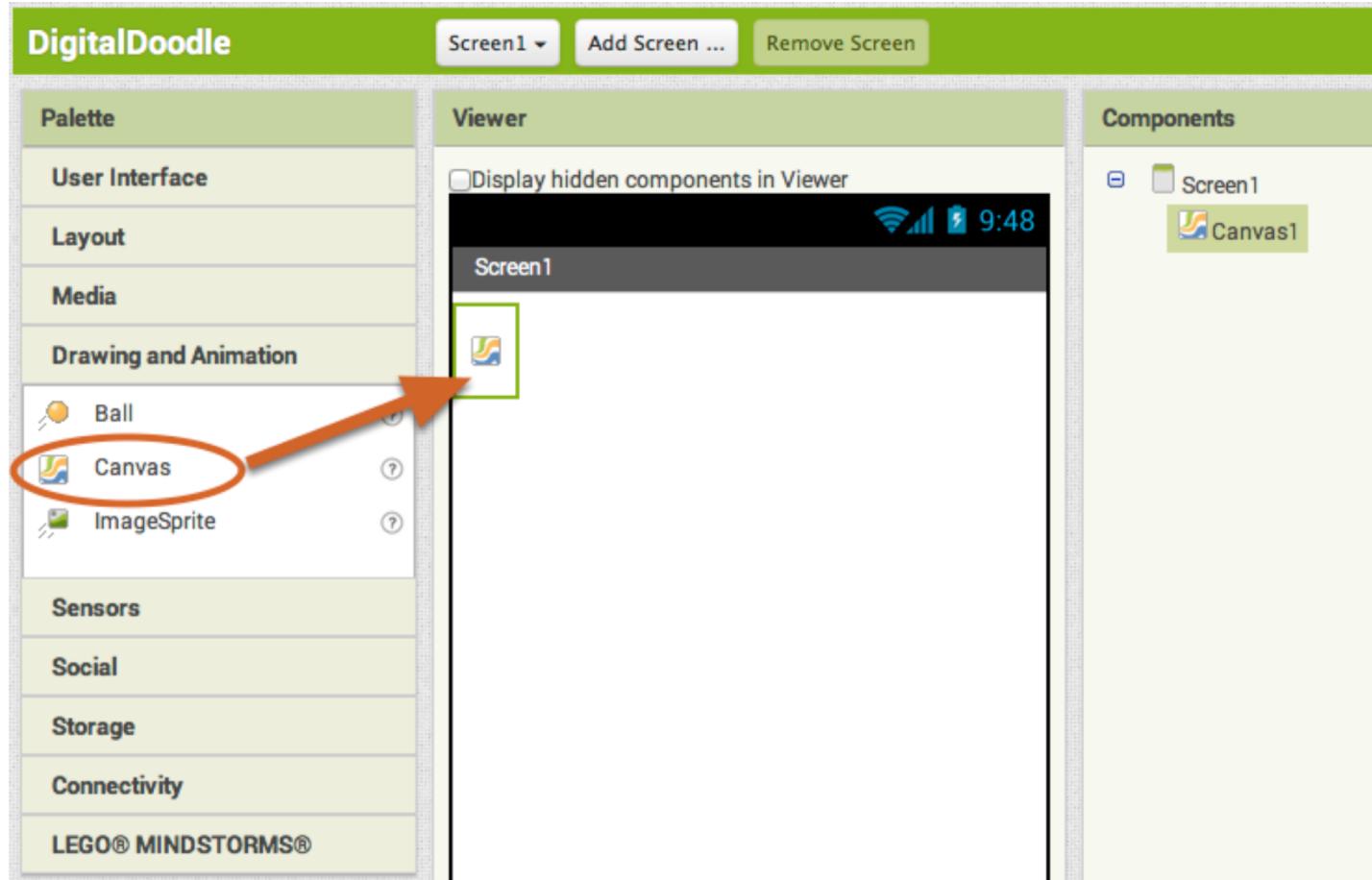
5 Set the Screen so that it does not scroll

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.



6 Add a Canvas

From the Drawing and Animation drawer, drag out a Canvas component.

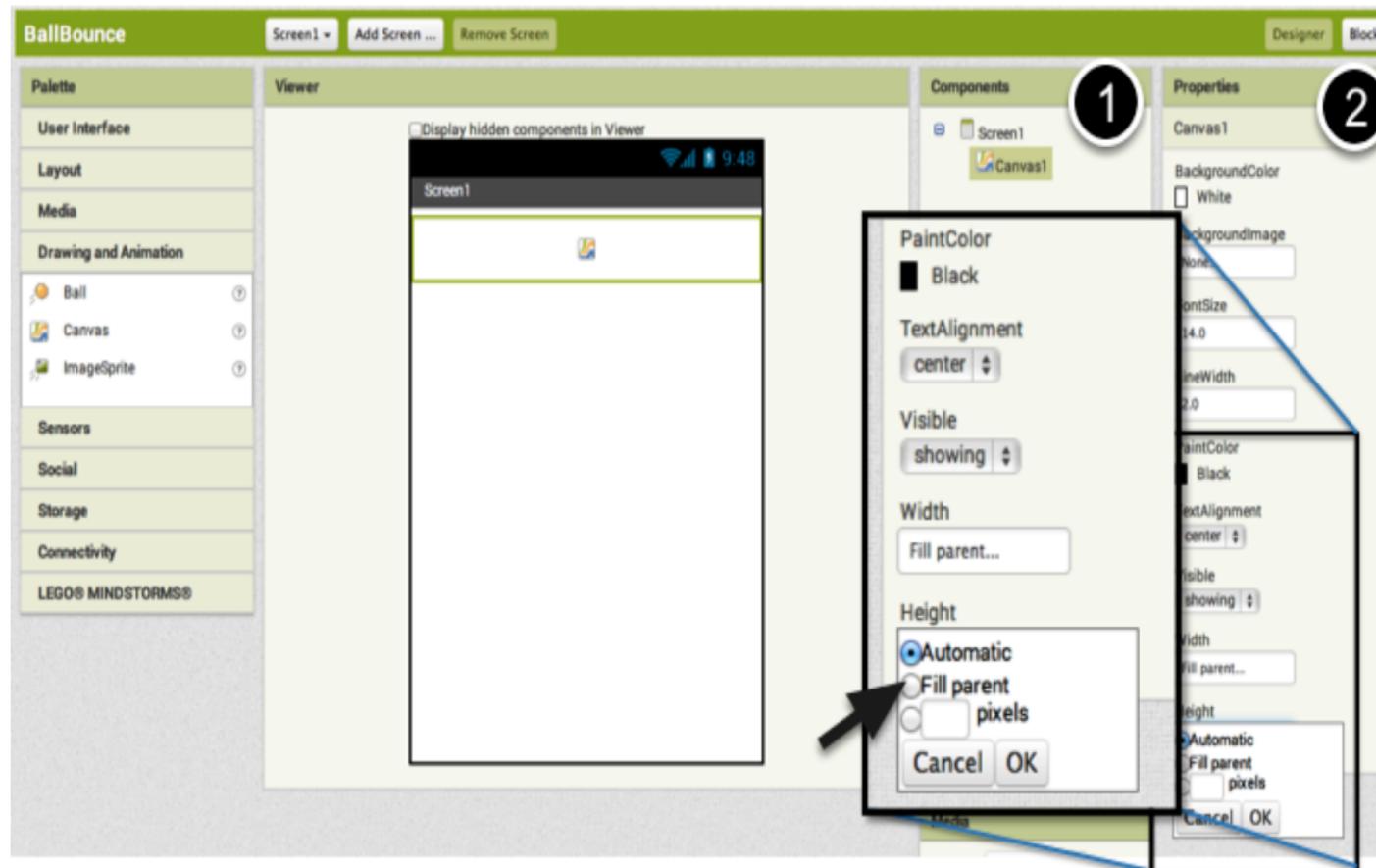




7

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*.

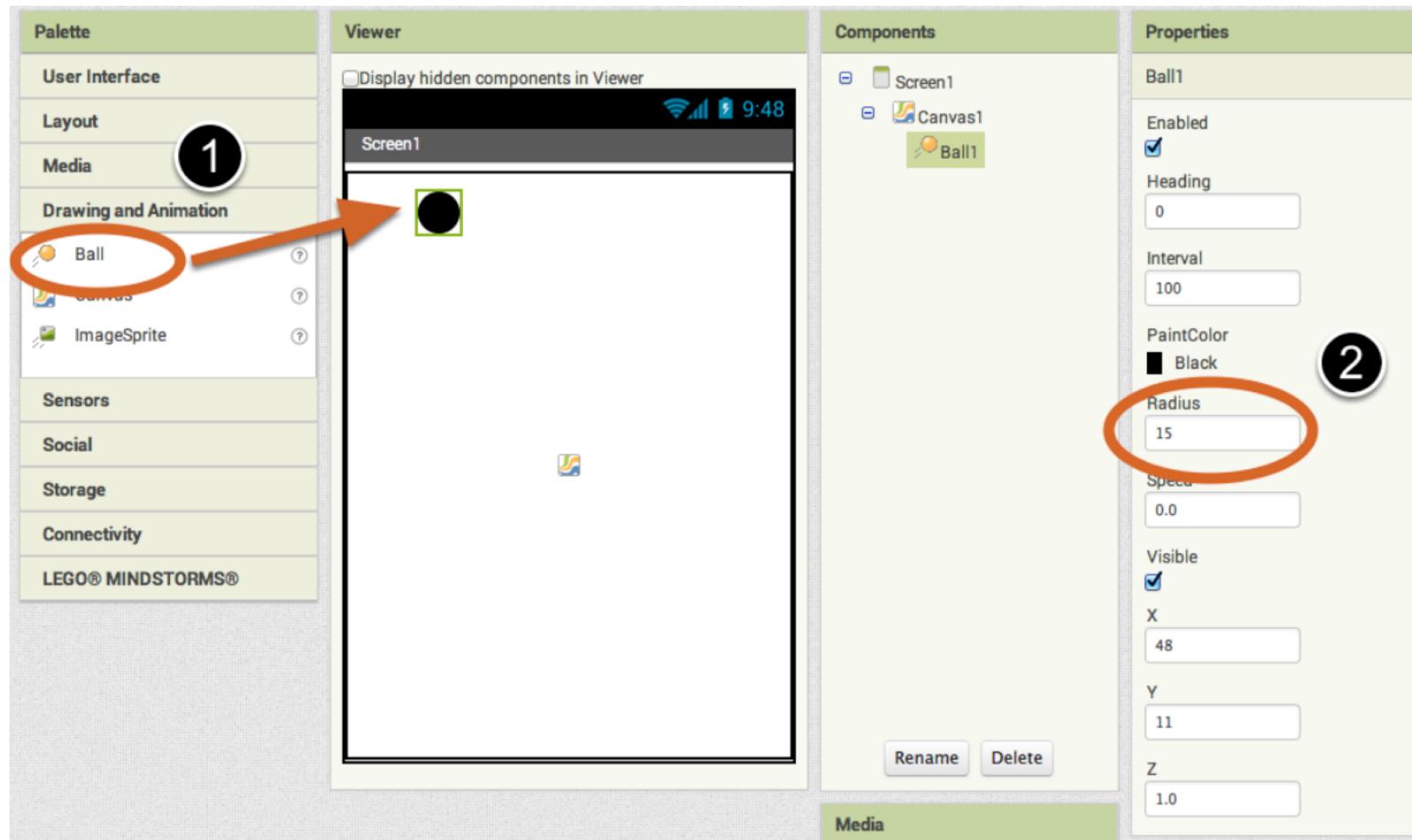


8

Switch to Blocks Editor

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 Get a Canvas.Dragged event block

In the Canvas1 drawer, pull out the **when
Canvas1.Dragged** event.

The image shows a Scratch script editor interface titled "DigitalDoodle". The top menu bar includes "Screen1", "Add Screen ...", and "Remove Screen". The left sidebar, titled "Blocks", lists categories: Built-in (Control, Logic, Math, Text, Lists, Colors, Variables, Procedures), Screen1 (Canvas1, which is highlighted with a red circle), and Any component. The main area, titled "Viewer", displays three scripts attached to the "Canvas1" component:

- A yellow "Control" script: "when [Canvas1].Dragged" with a "do" slot containing no blocks.
- A yellow "Control" script: "when [Canvas1].Flung" with variables "x", "y", "speed", "heading", "xvel", "yvel", and "flungSprite" in its slot.
- A yellow "Control" script: "when [Canvas1].TouchDown" with variables "x" and "y" in its slot.

11 Get a Canvas.DrawLine call block

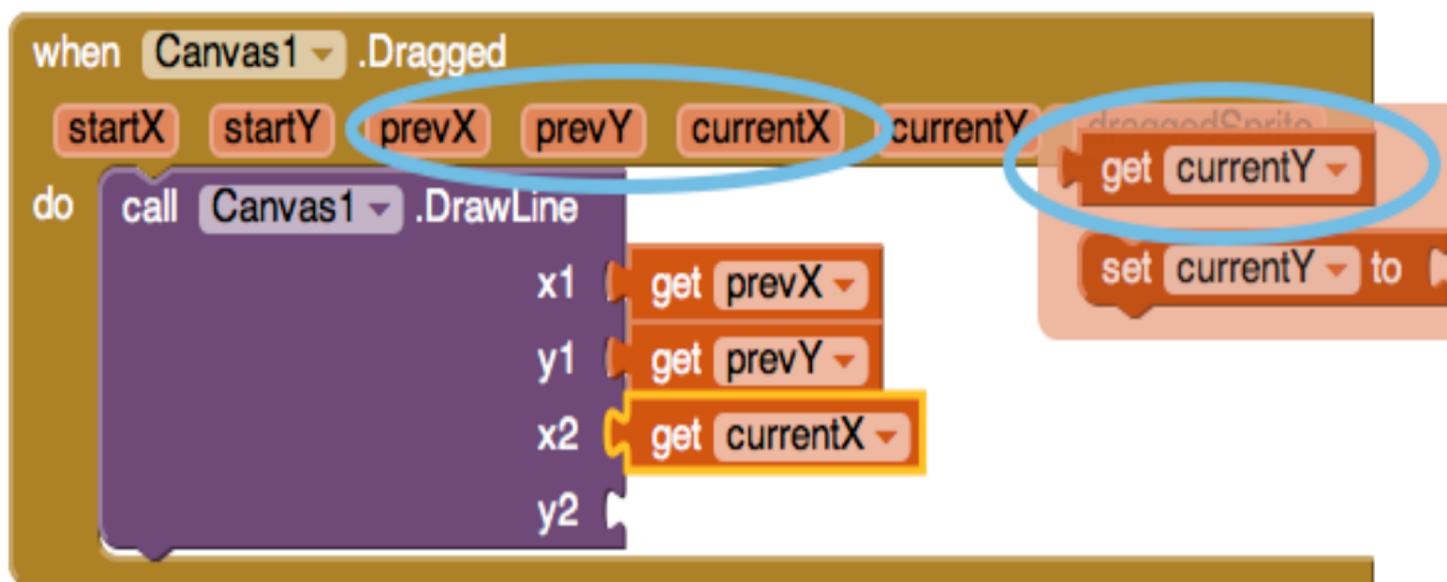
In the Canvas1 drawer, pull out the **when Canvas1.DrawLine** method block..

The image shows a Scratch script in the 'Viewer' tab. The script begins with a 'when [Canvas1] .Touched' event. Inside the loop, there is a 'do' block containing three 'call [Canvas1] .Clear' blocks. Below this, there are two more 'call [Canvas1] .DrawCircle' blocks, each with parameters x, y, and r. The entire 'do' block is circled in orange. At the bottom of the script, there is another 'call [Canvas1] .DrawLine' block with parameters x1, y1, x2, and y2, also circled in orange. The 'Blocks' palette on the left lists categories like Built-in, Control, Logic, Math, Text, Lists, Colors, Variables, Procedures, Screen1, and Any component. The 'Canvas1' category is highlighted with a green oval.

12

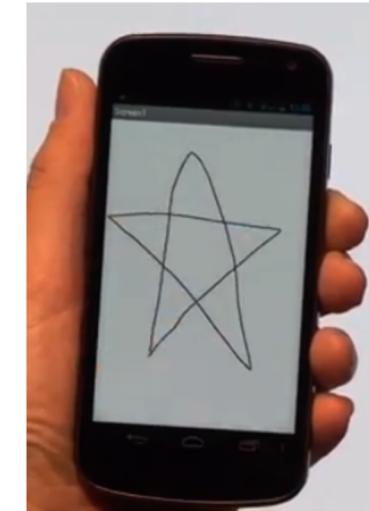
Use the get and set blocks from the Dragged block to fill in the values for the Draw Line block.

The Canvas Dragged event will happen over and over again very rapidly while the user drags a finger on the screen. Each time that Dragged event block is called, it will draw a small line between the previous location (prevX, prevY) of the finger to the new location (currentX, currentY). Mouse over the parameters of the Canvas1.Dragged block to pull out the get blocks that you need. (Mouse over them, don't click on them.)



13

Now test your app





Great work! Now extend this app

Here are some ideas for extending this app. You can probably think of many more!

- Change the color of the ink (and let the user pick from a selection of colors).
- Change the background to a photograph or picture.
- Let the user draw dots as well as lines (hint: Use DrawCircle block).
- Add a button that turns on the camera and lets the user take a picture and then doodle on it.