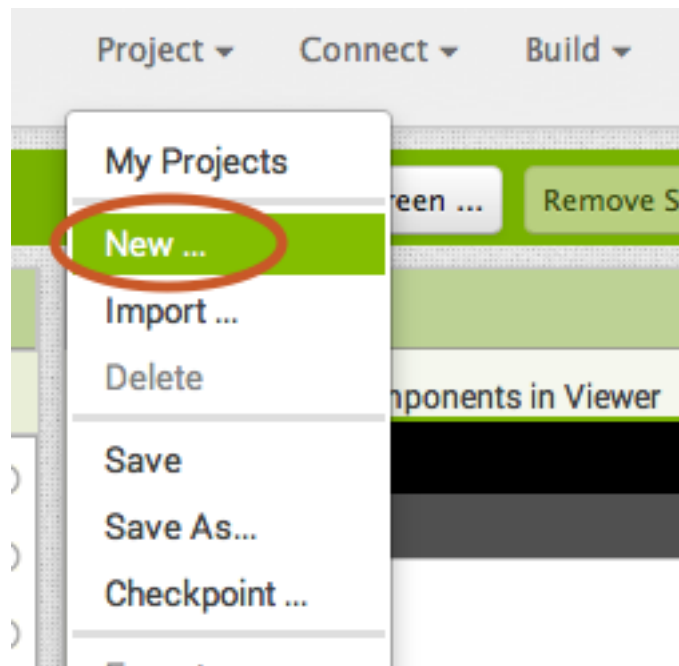


DigitalDoodle: Drawing App

This tutorial will show you how to draw a line on the screen as the user drags a finger around.

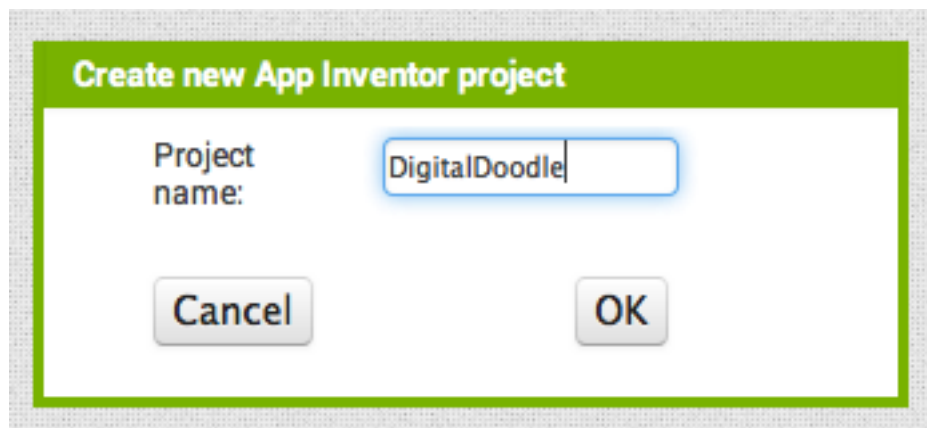
Start a New Project

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



Name the Project

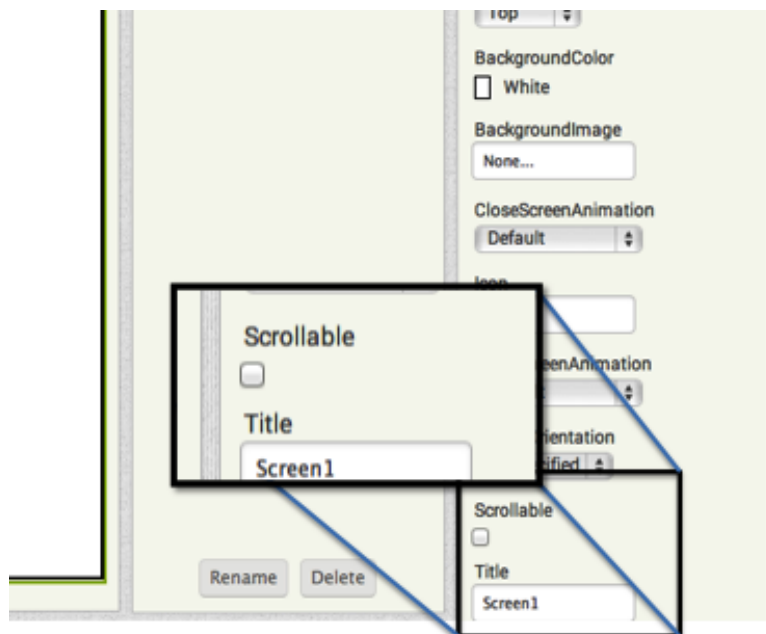
Call this project DigitalDoodle, or create your own name for this drawing app.





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.





Add a Canvas

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

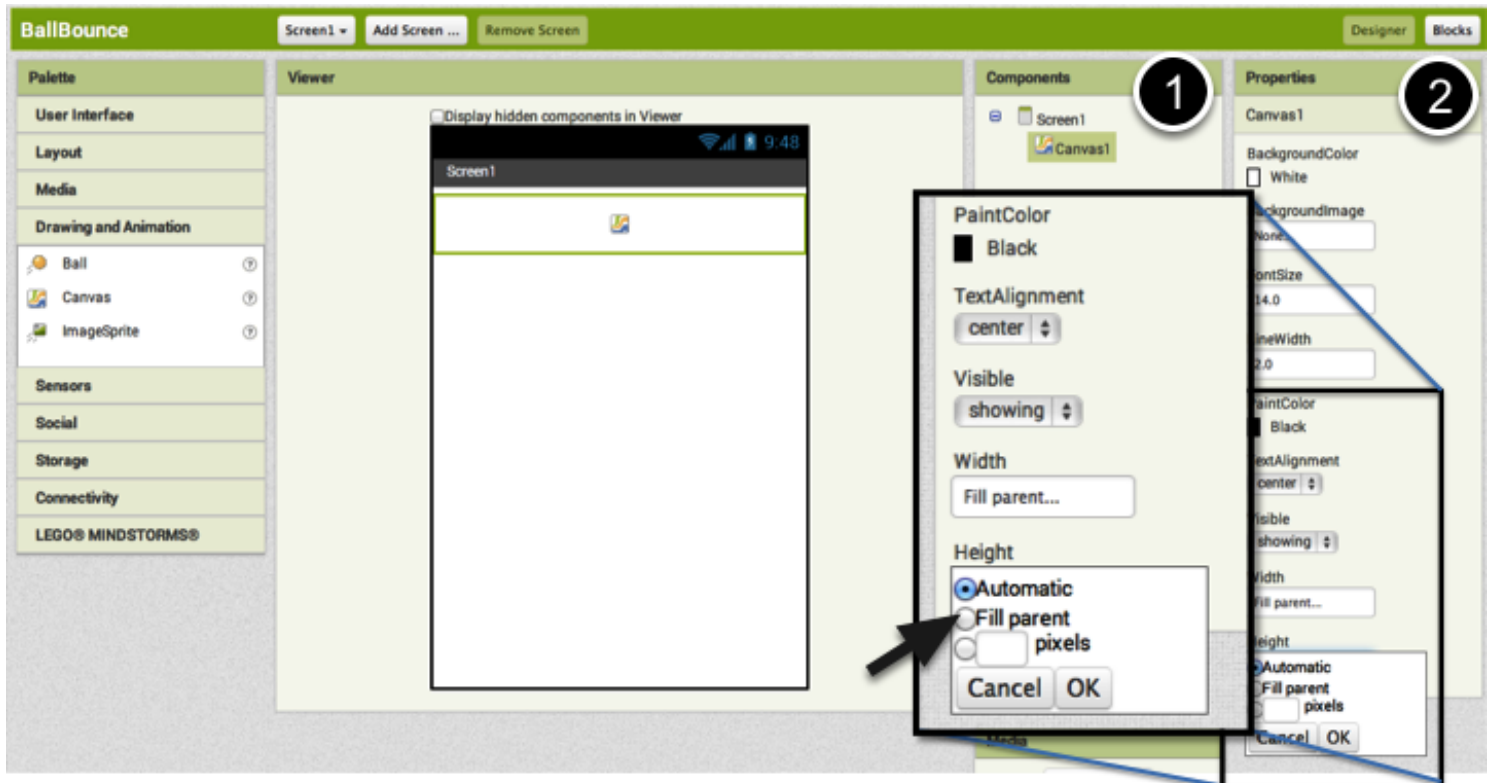
The screenshot displays the MIT App Inventor interface for an application named "DigitalDoodle". At the top, there are controls for "Screen1", "Add Screen ...", and "Remove Screen". The interface is divided into three main sections: "Palette", "Viewer", and "Components".

- Palette:** A vertical list of component categories. The "Drawing and Animation" category is selected and highlighted. Within this category, the "Canvas" component is circled in orange. Other components listed include "Ball", "ImageSprite", "Sensors", "Social", "Storage", "Connectivity", and "LEGO® MINDSTORMS®".
- Viewer:** A central area showing a mobile device screen. The status bar at the top indicates "Screen1", signal strength, Wi-Fi, battery, and the time "9:48". The Canvas component is visible in the bottom-left corner of the screen, highlighted with a green border. An orange arrow points from the "Canvas" component in the Palette to this icon in the Viewer.
- Components:** A panel on the right showing the components currently on the screen. It lists "Screen1" and "Canvas1".



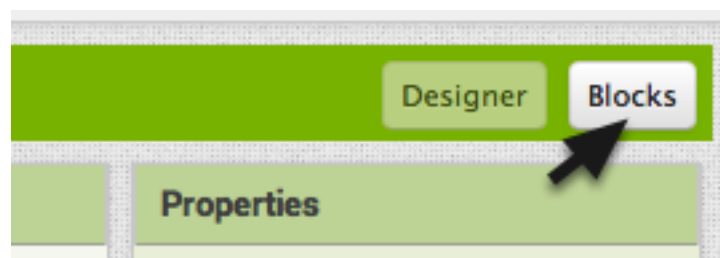
Change the Height and Width of the Canvas to Fill Parent

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**.



That's all for the Designer! Go over to the Blocks.

Believe it or not, for now this app only needs a Canvas. Go into the Blocks Editor to program the app.





Get a Canvas.Dragged event block

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

The screenshot shows the MIT App Inventor interface for an application named "DigitalDoodle". The interface is divided into two main sections: "Blocks" on the left and "Viewer" on the right. The "Blocks" section contains a "Built-in" category with various event and control blocks, and a "Canvas1" drawer which is circled in red. The "Viewer" section shows the code blocks for the application, with the "when Canvas1.Dragged" event block highlighted in red. This block has several input fields: startX, startY, prevX, prevY, currentX, currentY, and draggedSprite. Below it are two other event blocks: "when Canvas1.Flung" and "when Canvas1.TouchDown".

```
when Canvas1 .Dragged
  startX startY prevX prevY currentX currentY draggedSprite
do

when Canvas1 .Flung
  x y speed heading xvel yvel flungSprite
do

when Canvas1 .TouchDown
  x y
do
```



Get a Canvas.DrawLine call block

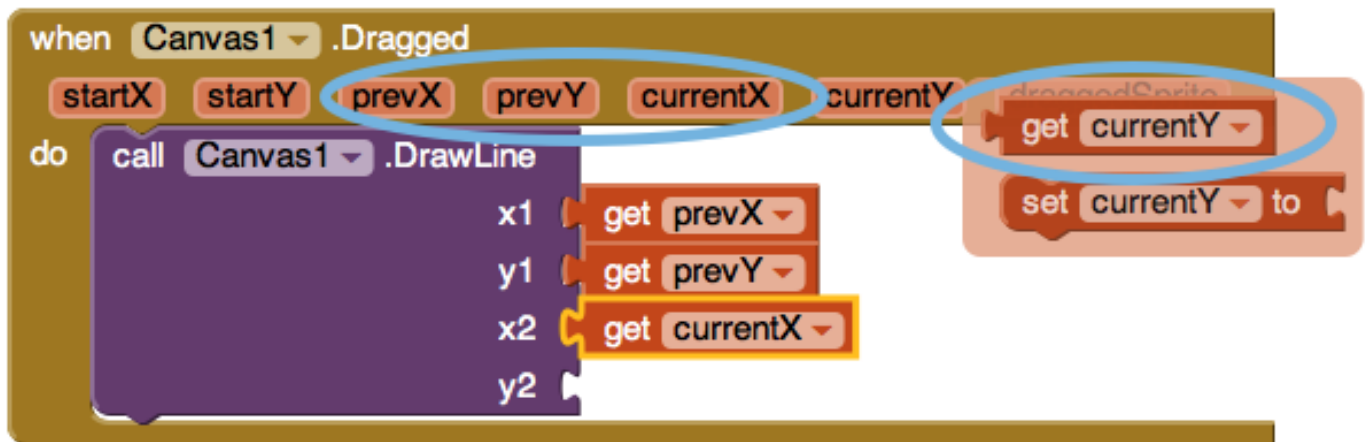
In the Canvas1 drawer, pull out the **when Canvas1.DrawLine** method block..

The screenshot displays the MIT App Inventor interface. On the left is the 'Blocks' palette, and on the right is the 'Viewer' showing a sequence of code blocks. In the 'Blocks' palette, the 'Canvas1' drawer is expanded, and the 'Canvas1' block is circled in orange. In the 'Viewer', a 'when Canvas1.Touched' block is connected to a 'do' block containing several 'call' blocks: 'call Canvas1.Clear', 'call Canvas1.DrawCircle' (with inputs 'x', 'y', and 'r'), 'call Canvas1.DrawLine' (with inputs 'x1', 'y1', 'x2', and 'y2'), and 'call Canvas1.DrawPoint'. The 'call Canvas1.DrawLine' block is circled in orange.



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

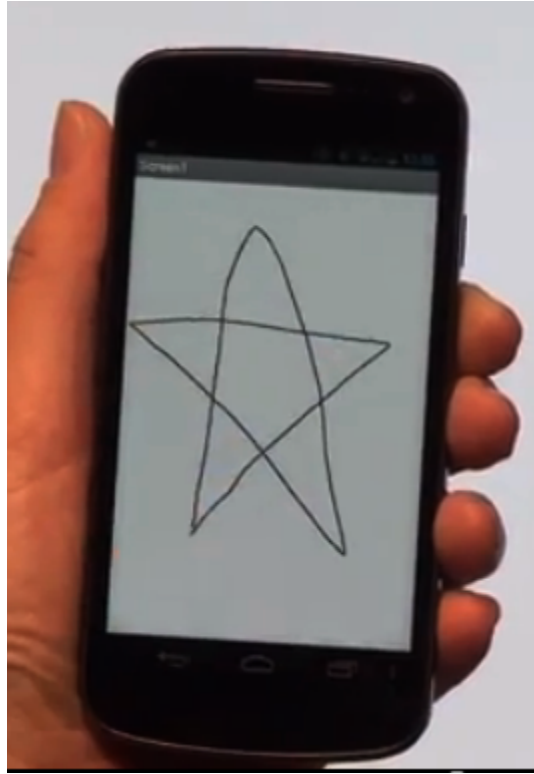
The Canvas Draggged event will happen over and over again very rapidly while the user drags a finger on the screen. Each time that Draggged event block is called, it will draw a small line between the previous location (prevX, prevY) of the finger to the new location (currentX, currentY). Mouse over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





Test it out!

Go to your connected device and drag your finger around the screen. Do you see a line?



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). See [Paint Pot tutorial](#).
- 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.