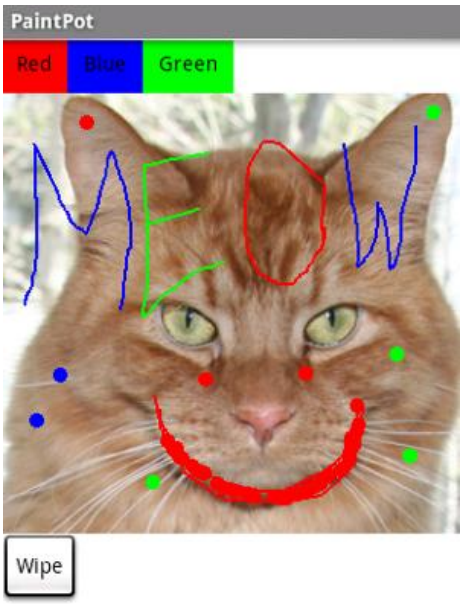
PRÁCTICA 2. PaintPot

Primera parte

Este tutorial presenta el componente **"Canvas"** (lienzo), para la creación de gráficos simples de dos dimensiones. Construirás una aplicación que te permitirá **dibujar en la pantalla del teléfono** en diferentes colores:



Con la aplicación ***PaintPot*** , podrás:

- Sumergir tu dedo en una "olla virtual" de pintura para dibujar con ese color.
- Arrastrar el dedo por la pantalla para dibujar una línea.
- Tocar la pantalla para hacer puntos.
- Utilizar el botón en la parte inferior para borrar la pantalla.
- Más adelante podrás incluir una imagen como fondo del dibujo.

Este tutorial presenta los siguientes conceptos AppInventor:

- El **Canvas component** (componente de lienzo) componente para el dibujo.
- El control de diseño de la pantalla con el **Arrangement components** (Orden de componentes)**.**
- Los **controladores de eventos** que toman **argumentos.**
- **Variables.**

Comenzamos la aplicación

Inicia un nuevo proyecto en la ventana del **Designer**, y el nombre de ***"PaintPot"*** .

Título de la pantalla

Para empezar, ves al panel **Properties** de la derecha del **Designer** y cambiar la pantalla de **título** a **"PaintPot"**.

Hay tres nombres en App Inventor, y es fácil confundirlos:

1. El nombre que elijas para **tu proyecto** (en este caso, ***PaintPot***). Este también será el nombre de la aplicación si se empaqueta para el teléfono.
2. El nombre **"Screen1"**, que es el nombre de la **pantalla** de componentes. Verás que aparece en el panel Componentes en el **Designer**. No se puede cambiar el nombre del primer componente de la pantalla en la versión actual de la aplicación AppInventor pero puedes crear pantallas adicionales con nombres que puedes elegir.
3. El **título** de la pantalla, que es lo que verás en la barra de título del teléfono. El título es una propiedad de la **pantalla** de componentes. El título empieza siendo**"Screen1"** . Sin embargo, se puede cambiar, como estás haciendo por *PaintPot* . Resumiendo, el nombre y el título de Screen1 son inicialmente el mismo, pero se puede cambiar el título, si quieres.

Configurar los componentes

Vamos a usar estos componentes para hacer ***PaintPot*** :

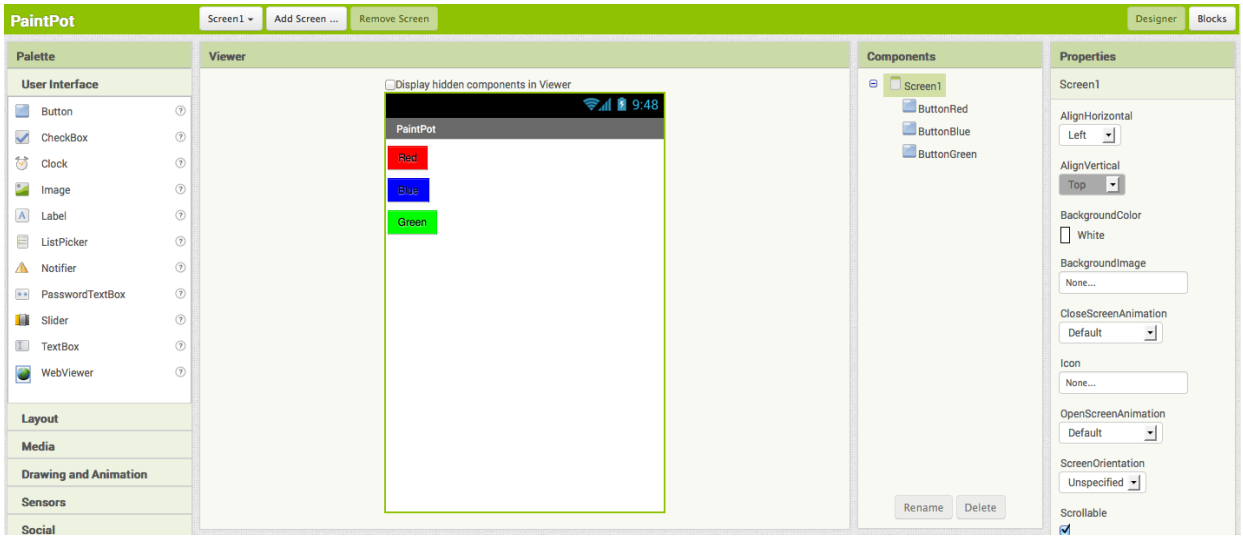
- Tres **Buttons** de selección de **color rojo** , **azul** o **verde** de la pintura, y otro botón para limpiar el dibujo.
- Un **Canvas** , la superficie de dibujo. Este lienzo tiene un **BackgroundImage** , que será la del gatito de *HelloPurr*. También puedes dibujar en un lienzo en blanco. Es sólo una imagen de fondo para el lienzo o una pantalla en blanco.
- También hay un componente que no se ve: se utiliza un **HorizontalArrangement** para colocar los **tres botones** de colores **alineados**.

Eso hace cinco componentes en total. Vamos a por ellos y construir la aplicación.

Botones de colores

- Arrastra un **componente Button** a la pantalla y cambia atributo **Text** del botón a **"Rojo"** y define su **BackgroundColor** como **rojo**.
- Haz clic en ***Button1*** en la lista de los componentes en el visor para seleccionarlo y utiliza el **Rename ... Button** para cambiar su nombre de **"Button1"** a **"ButtonRed"**.
- Del mismo modo, haz dos botones más de **azul** y **verde**, llámalos **"ButtonBlue"** y **"ButtonGreen"**, colócanlos verticalmente debajo del botón rojo.

Debe de quedar así. El uso de nombres significativos hace que los proyectos sean más legibles para ti y para los demás.

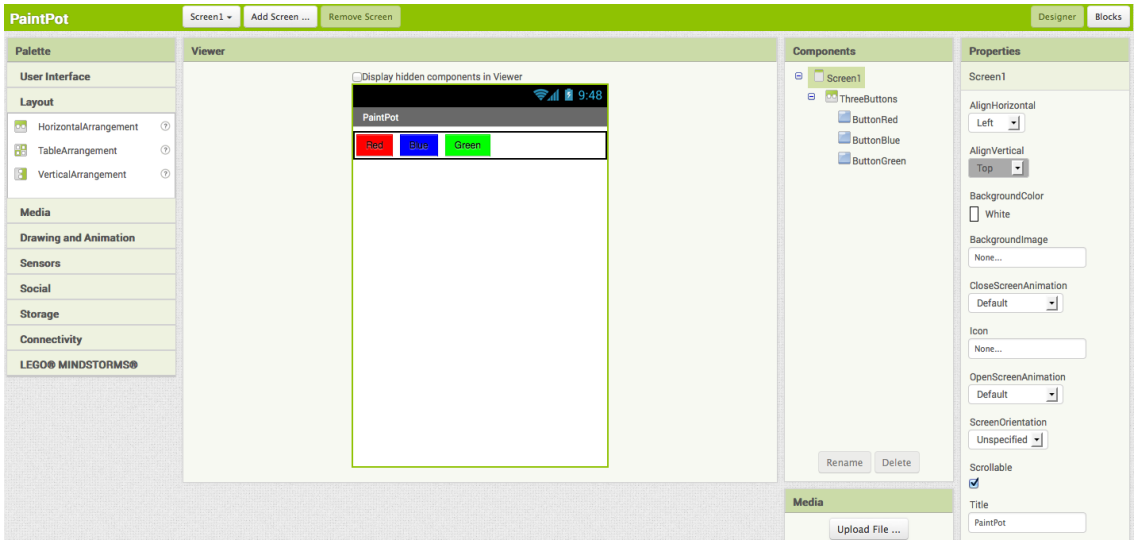


Ordenar la pantalla

El siguiente paso es hacer que los botones se **alineen horizontalmente**. Esto se hace utilizando un componente **HorizontalArrangement**.

1. De la categoría **Palette's Layout**, arrastra un componente **HorizontalArrangement** y colocarlo debajo de los botones. Cambia el nombre de este componente de **"HorizontalArrangement1"** a **"ThreeButtons"**.
2. En el panel **Properties**, cambia el **Ancho** de ***ThreeButtons*** a **"Fill Parent ..."** de modo que llene toda la anchura de la pantalla.
3. Mueve los tres botones al **HorizontalArrangement.** Podrás ver una línea vertical de color azul que muestra dónde se está arrastrando la pieza.

Si nos fijamos en la lista de **Components** del proyecto, verás los tres botones colocados bajo ***ThreeButtons*** para mostrar que ahora son sus **subcomponentes**. Observa que todos los componentes están colocados bajo ***Screen1*** .



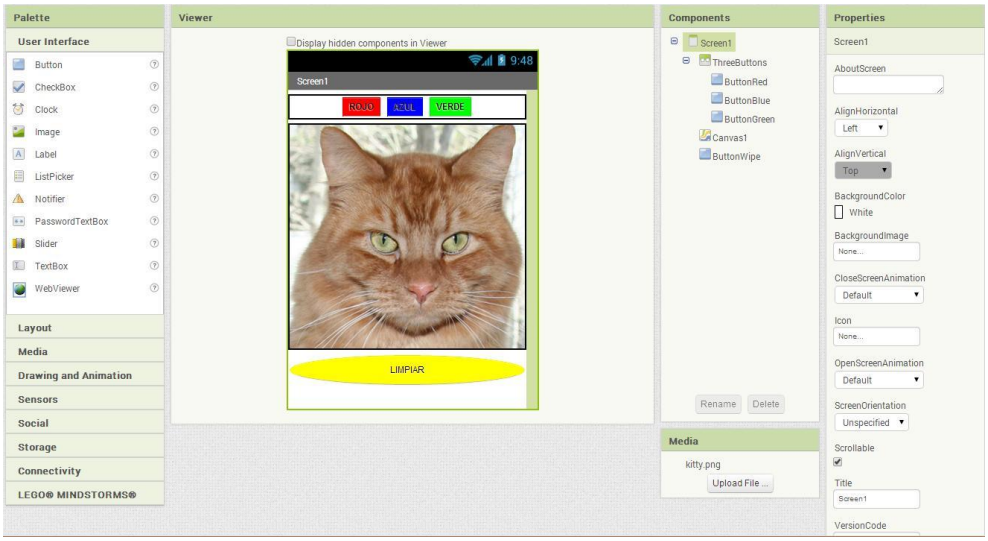
En general, se utiliza **Screen Arrangement** para crear diseños **verticales** u **horizontales** simples. También puedes crear diseños más complejos anidando componentes **Screen Arrangement** , existe un componente llamado **TableArrangement**.

Lienzo de dibujo y borrar pantalla.

Los dos últimos componentes son el de **Canvas** (lienzo) y el **botón de borrar**.

1. De la paleta de categorías **Palette's Drawing and Animation** arrastra un componente **Canvas** a la pantalla. Cambia su nombre a **"DrawingCanvas".** Establece su**Ancho** a **"Fill de Parent"** y su **Altura** a 300 píxeles.
2. Agregar una **imagen de fondo** al **Canvas** . Haz clic en el campo que contiene **"None..."** junto a **BackgroundImage** en **Properties panel**. Puedes utilizar el mismo archivo kitty.png, o utilizar otra imagen.
3. Desde la paleta, arrastra el último botón a la pantalla, colocándolo bajo el lienzo. Cambia su nombre a **"ButtonWipe"** y cambia su atributo **text** a **"Limpiar"**.

Ahora ya has completado los pasos para configurar el aspecto de la aplicación. Así es como este debería ser en el **Designer**.



A continuación, definirás cómo se comportarán los componentes.

Añadir comportamientos a los componentes

Haga clic en el botón de **Blocks** para cambiar al **Editor de bloques**. Primero vas a configurar **los botones que cambian el color de la pintura**. Posteriormente agregarás**bloques** para decidir lo que ocurre cuando alguien **toca o arrastra la pantalla.**

Agregar controladores de eventos de botón

En el **Editor de bloques**:

1.- Abre la paleta del ***ButtonRed*** y arrastre el bloque **when ButtonRed.Click** .

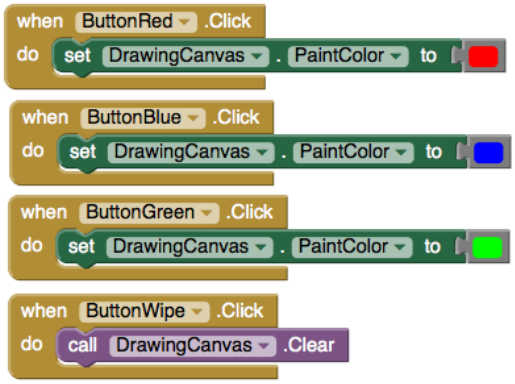
2.-Abre la paleta ***DrawingCanvas***. Arrastra el **set DrawingCanvas.PaintColor** y colócalo en el hueco **do** de **when ButtonRed.Click** .

3.- Abre la palata **Colors** y arrastra el bloque para el color rojo y ponerlo en **set DrawingCanvas.PaintColor to** . (Hacer clic en un bloque de color después de haber sido colocado mostrará una tabla de colores que usted puede elegir.)

4.- Repite los pasos 2-4 para los botones de **color azul** y **verde**.

5.- El último botón a configurar es el botón **Limpiar**. Arrastra un controlador **when ButtonWipe.Click** de la paleta ***ButtonWipe.*** Desde la paleta ***DrawingCanvas***, arrastra**call DrawingCanvas.Clear** y colócalo en el hueco de **do** del bloque **when ButtonWipe.Click**.

Los bloques de los botones deben quedar así:



Añadir Eventos táctiles. Puntos

Vamos a organizar los componentes para que al **tocar el lienzo,** se obtenga **un punto** en el lugar **donde se toca**, y si se **arrastra el dedo lentamente** por el lienzo, **dibujar una línea.**

- En el **Editor de bloques**, abre la paleta de **Canvas** y arrastra el bloque **when DrawingCanvas.Touched** al área de trabajo. Tan pronto como se arrastra el bloque de salida, puedes observar tres **nombres de argumento** (resaltados en naranja), ubicados en la parte superior del bloque de **x** , **y** , y **touchedSprite** . Estos **argumentos**también se conocen como **variables locales** y pueden obtener acceso mediante el **get** o **set bloque** que se encuentra en la **paleta de las variables** y luego seleccionar la **variable** adecuada en el **menú desplegable**. También pueden acceder a estas **variables** al mover el **cursor sobre el nombre** resaltado y **seleccionar la variable** que se desea utilizar.

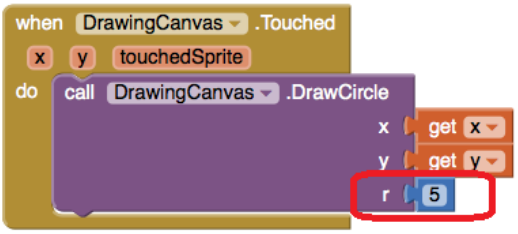
Ya has visto los eventos de clic de botón, son simples, realizan una acción. Otros controladores de eventos, como **when... Touched** necesitan información sobre el evento. En AppInventor, esta información se expresa **con el valor de los argumentos relacionados con el controlador de eventos**. Para el **when ... Touched**, los dos primeros argumentos representan las **coordenadas x e y de donde ocurrió el contacto.** El argumento **touchedSprite** lo explicaremos más adelante.

- Para este evento táctil, hacer que en el lienzo se dibuje un pequeño círculo en el **punto de coordenadas (x, y),** arrastra un comando **call DrawingCanvas.DrawCircle**desde la paleta de **Canvas** y colócala en el hueco de **do** de **when DrawingCanvas.Touched** .

En el lado derecho del bloque **call DrawingCanvas.DrawCircle** hay tres salidas que se deben especificar , los valores para las **coordenadas X e Y** dónde debe trazarse el círculo, y **r** , que es el radio del círculo. Para **x** e **y** , utilizará los valores de los argumentos que fueron suministrados al **controlador Touch**:

1. Mueve el cursor sobre la **variable x** (resaltada en color naranja). Encuentra el bloque **get x** y arrástralo a la salida correspondiente de **x** en el bloque **when DrawingCanvas.Touched** .
2. Haz lo mismo con la **variable y**.
3. También tendrás que especificar **el radio del círculo** para dibujar. **Cinco** (píxeles) es un buen valor para esta aplicación. Haz clic en un área vacía de la pantalla y escribe el número 5 seguido de retorno para crear un bloque de números con un valor de 5. El pulsar en el área en blanco de la pantalla se denomina **typeblocking** y es un atajo útil de saber. Esto se puede hacer de cualquier bloque, no sólo números. Coloca el bloque de **5** en el hueco de **radio.**

Así es como el controlador de eventos **touch** debe quedar:



Prueba lo que has hecho hasta ahora en el emulador. Toca un botón de color. Ahora toca el lienzo, y el cursor debe dejar una mancha en cada lugar donde toca. Al tocar el botón Limpiar debe limpiarse el dibujo.

Añadir Eventos táctiles. Líneas

Por último, agregar el controlador de **evento de arrastre**. Aquí está la diferencia entre un **contacto y un arrastre**:

- **Un contacto** (toque) es cuando pones el dedo sobre el lienzo y lo levantas sin moverlo.
- **Un arrastre** es cuando pones el dedo sobre el lienzo y mueves el dedo mientras lo mantienes en contacto.

Al arrastrar el dedo por la pantalla, se dibujan infinidad de líneas rectas una a continuación de otra, donde termina la primera empieza la última . Por tanto debemos hablar de la **última posición** y de la siguiente o **actual.**

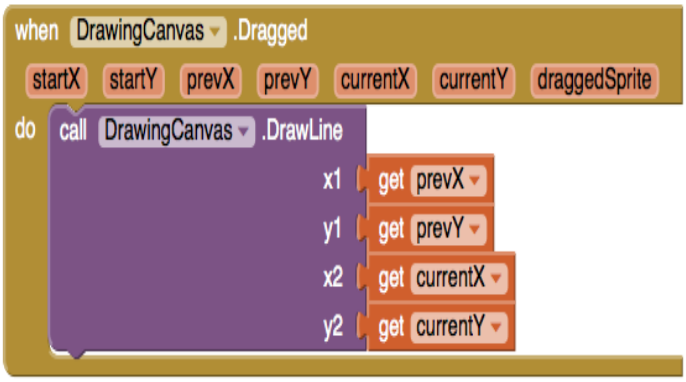
Un evento de arrastre viene con **6 argumentos**. Se trata de **tres pares de coordenadas X e Y que muestran:**

- La posición del dedo de a donde comenzó el arrastre.
- La posición actual del dedo.
- La posición inmediatamente anterior del dedo.

También hay un **sprite**, que vamos a ignorar para este tutorial.

Ahora trazaremos una línea entre la posición anterior y la posición actual mediante la creación de un **controlador de arrastre**:

1. Desde la paleta ***DrawingCanvas*** , arrastra el bloque **when DrawingCanvas.Dragged** al área de trabajo.
2. También desde ***DrawingCanvas*** , arrastra el bloque **call DrawingCanvas.DrawLine** en el hueco do ranura del bloque **when DrawingCanvas.Dragged**.
3. Arrastra un bloque **Prevx get** a la salida **x1** en **when DrawingCanvas.DrawLine** . Haz lo mismo con los otros espacios: **y1** debe obtener **prevY** , **x2** debe ser**CurrentX** y **y2** debe ser **CurrentY** .



Prueba la aplicación, arrastre el cursor por la pantalla para dibujar líneas y curvas. Toca la pantalla para hacer puntos. Utiliza el botón **Limpiar** para limpiar la pantalla.

En la segunda parte , veremos cómo utilizar variables globales para crear puntos de diferentes tamaños.

Segunda parte

En esta parte del tutorial crearemos puntos grandes y pequeños, como demostración de cómo utilizar *las **variables globales*** .

Comienzo

Asegúrate de que has completado la primera parte y que tienes cargado el proyecto.

Utiliza el **Guardar como** para hacer una **copia** de **PaintPot** para que puedas trabajar en la **nueva versión** sin afectar a la versión original. Nombre de la copia **"PaintPotV2"**(sin espacios). Después de guardar, debes ver **PaintPotV2** en el **Designer**.

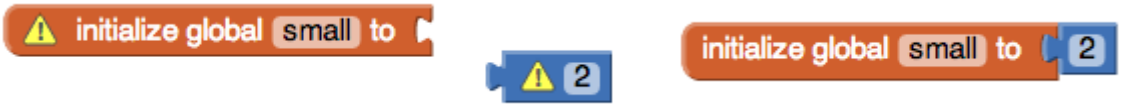
Creación de las variables

El tamaño de los puntos dibujados en el lienzo se determina en el controlador de eventos **when DrawingCanvas.Touched** donde la **call Drawing.DrawCircle** define con **r** , el radio del círculo igual a **5**. Para cambiar el grosor, todo lo que necesitamos hacer es utilizar diferentes valores para **r** . Utiliza **r = 2** para pequeños puntos y **r = 8** para los puntos grandes por ejemplo.

Comienza por crear l**as variables** para estos **valores:**

1. Abre el **Editor de bloques**, si no está ya abierto.
2. En el **Editor de bloques**, en la columna **Built-In**, abre la **paleta Variables**. Arrastra un bloque **initialize global name to**. Cambiar el **texto** que dice "**name**" por "**small**". Un signo de exclamación amarillo de advertencia aparecerá en el bloque. Si pasas el ratón sobre este verás un mensaje de advertencia explicando que el bloque tiene una salida vacía.
3. Es necesario conectar la salida con un bloque numérico que especifique el valor de "**small**", utiliza **2** como valor. La marca amarilla de advertencia desaparecerá, porque el socket vacío ha sido conectado.

Estos son los pasos:



Has definido una **variable global** llamada **"small**" cuyo **valor** es el número **2**.

Define ahora una **variable global "big"** , cuyo **valor** sea **8**.

Finalmente, define una **variable global "dotsize"** y define un **valor inicial de 2**.

Uso de variables

Ahora vuelve al **controlador de eventos touch** configurado en **la primera parte** y cambia el bloque **call to drawCircle** para que utilice el valor de **dotsize** en lugar de utilizar siempre **5**.

En el **Editor de bloques**, abre la paleta de **Variables**. Arrastra un bloque **get** y haz click en la **lista desplegable**. Debes ver tres nuevas variables en el menú desplegable:**small, big, y dotsize.**

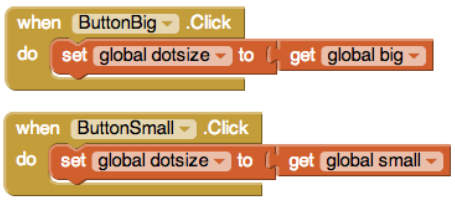
Ves al controlador de eventos **when MyCanvas.Touched** y sustituye el **bloque número 5** en **call drawCircle** por el get **dotsize** de la paleta de **Variables**.

Estos bloques fueron creados y puestos en la lista desplegable **get** y **set** de forma automática, de manera similar a la forma en que X e Y fueron creados en el menú desplegable al definir el controladorde eventos **when DrawingCanvas.Touched** en la **primera parte** de este tutorial. **"Global"** significa **"variable global"**, en contraste con los argumentos de controladores de eventos, que se consideran **"variables locales"**. La diferencia es que **los valores de los argumentos sólo son accesibles dentro del cuerpo del controlador de eventos**, mientras que **las variables globales son accesibles desde todo el programa**.

Cambio de los valores de las variables

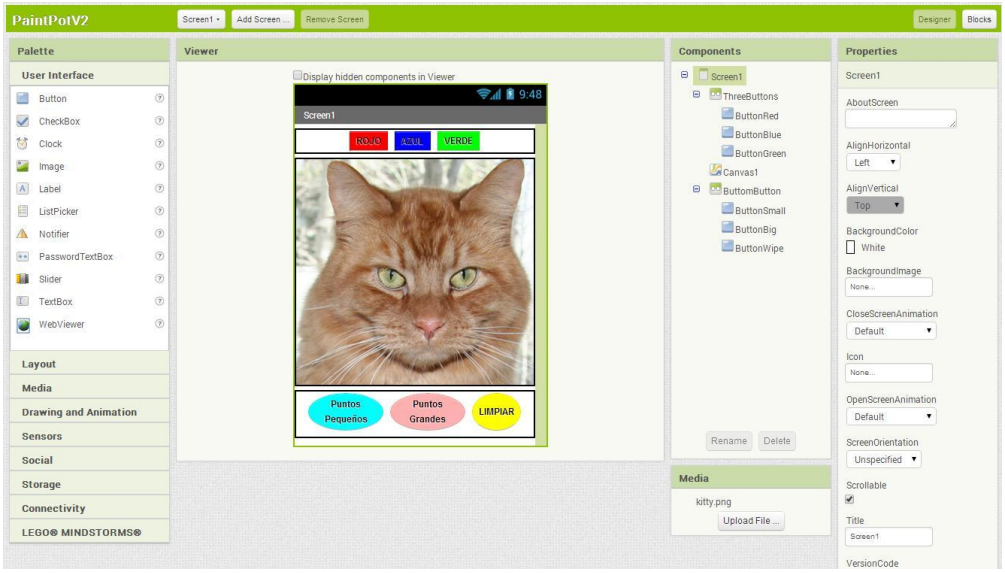
Ahora configuraremos una manera de cambiar el tamaño de punto para que sea pequeño (2) o grande (8). Haz esto con los controladores de botones.

1. En el Designer, arrastra un componente **HorizontalArrangement** al panel del visor debajo del componente **DrawingCanvas** . Llámalo **"BottomButtons"**.
2. Arrastre el botón **ButtonWipe** a **BottomButtons** .
3. Arrastre dos componentes más de botón de la paleta a **BottomButtons** , colocándolos junto a **ButtonWipe** .
4. Nombra los botones **"ButtonBig"** y **"ButtonSmall"**, y define el parámetro text como "Puntos Grandes " y "Puntos Pequeños ", respectivamente.
5. En el **Editor de bloques**, crea un controlador **when ... Click** para **ButtonSmall** que defina con un **set global dotsize to > get global small** el tamaño del punto pequeño
6. Repite la operación para el controlador **ButtonBig** .

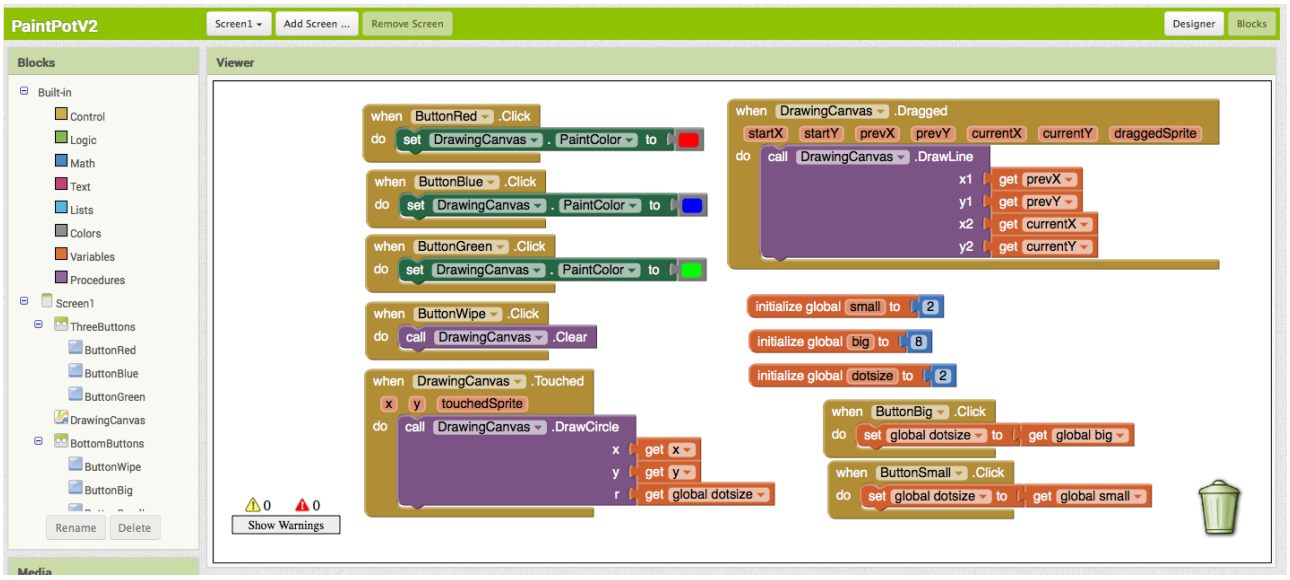


¡Ya está! Puedes dibujar en **PaintPot** y utilizar los nuevos botones para dibujar o bien puntos grandes o puntos pequeños. Ten en cuenta que si arrastras el dedo todavía produce una línea fina. Esto se debe a que los cambios que acabas de realizar no afectan al modo **DrawLine**.

Aquí está el programa completo en el Designer:



y en el Editor de bloques:



El programa que acabas de construir tiene un ligero error. Si comienzas a pintar antes de pulsar cualquiera de los botones de la pintura, el color de la pintura será de color negro, sin embargo, después de elegir un color, no hay manera de volver a negro. Piensa cómo se puede arreglar este defecto.