



Technovation iridescent
CANADA

Technovation 2017 Hack Day

Hack Day Trainer: Ahmed

Carrer pathway through Engineering

- ➔ B. Eng. in Software Engineering
Lakehead University
- ➔ Project Management Office
IBM (Automation, Legal Deliverables...)
- ➔ M. Eng. in Technology Innovation Management
Carleton University
- ➔ Cybersecurity Research and Development
VENUS Cybersecurity Corp.

3rd year at Technovation!

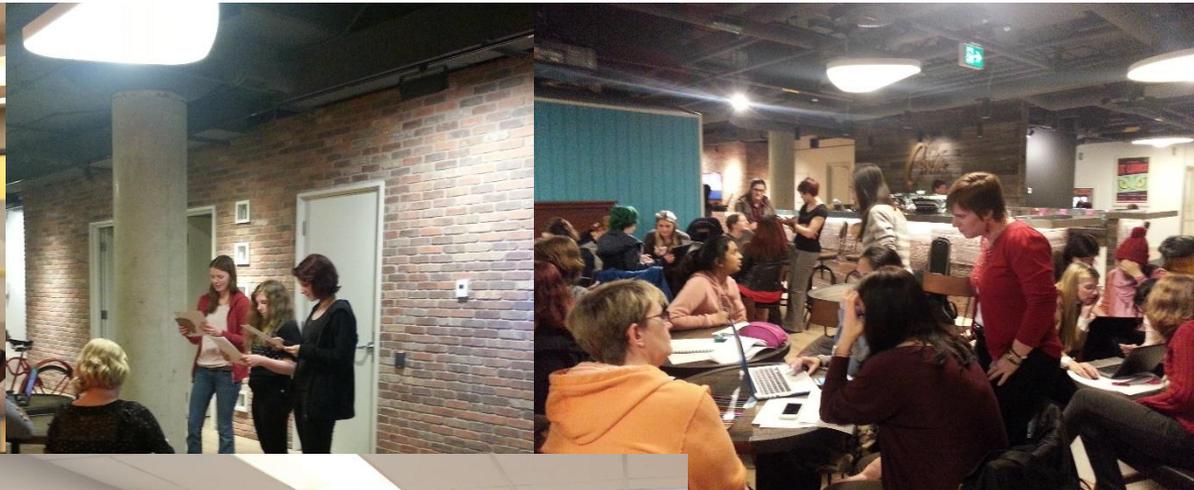
Welcome to Technovation

- ➔ Technovation will teach the skills you need to emerge as tech entrepreneurs and leaders.
- ➔ Working in teams of 3-5 you identify a problem in your community, and build a technology business to solve it.
- ➔ Technovation takes you through 4 stages of launching a mobile app startup, inspired by the principles of design thinking:
 - Ideation** - Identify a problem in the community
 - Technology** - Develop a mobile app solution
 - Entrepreneurship** - Build a business plan to launch the app
 - Pitch** - Take the business to market

Technovation Ottawa

Students from across Ottawa

January to April (12 Weeks)



What we are covering today

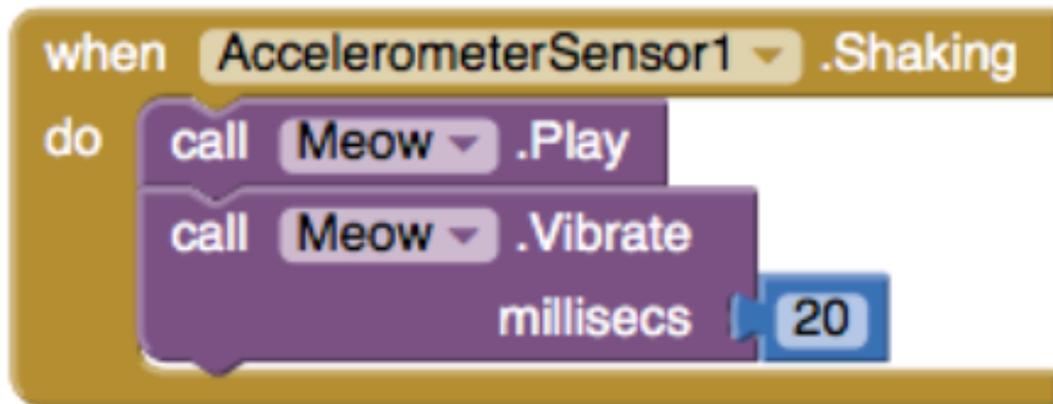
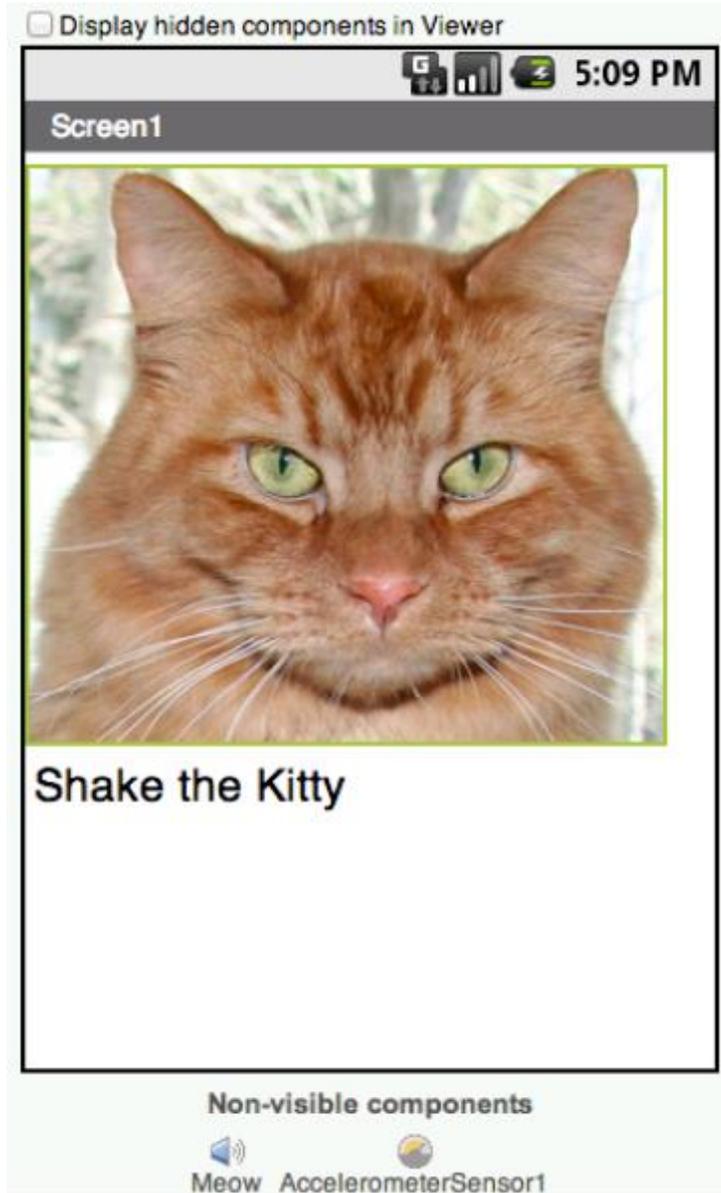
1. What you will need for today
2. Problem solving exercises
3. Application development cycle
4. Setting up your phone
 - Installing AI Companion
5. Tips for mobile development
6. Creating your first app “Talk to Me” (App 1)
 - Tutorial walkthrough
 - Downloading application
 - Save and distribute
7. -----Lunch-----
8. Slide show (App 2)
9. Colored Dots (App 3)
10. Continue Learning
11. Technovation: Next Steps
12. Math App Challenge

Why Learn App Inventor?

- ➔ Create your own mobile applications
- ➔ Use it for school projects
- ➔ Solve real world problems
- ➔ Create a prototype for your business

Problem solving exercises

What Does This App do?



Shaking Phone

Make something happen when you shake your phone.



```
when AccelerometerSensor1 .Shaking
do
  call Meow .Play
  call Meow .Vibrate
  milliseconds 20
```

The **AccelerometerSensor.Shaking** event will detect when the phone is shaking and then the Meow sound will play and the phone will vibrate for 20 milliseconds.

What Does This App do?



Non-visible components


SpeechRecognizer1

when PressAndSpeakButton .Click

do call SpeechRecognizer1 .GetText

when SpeechRecognizer1 .AfterGettingText

result

do set TextLabel . Text to SpeechRecognizer1 . Result

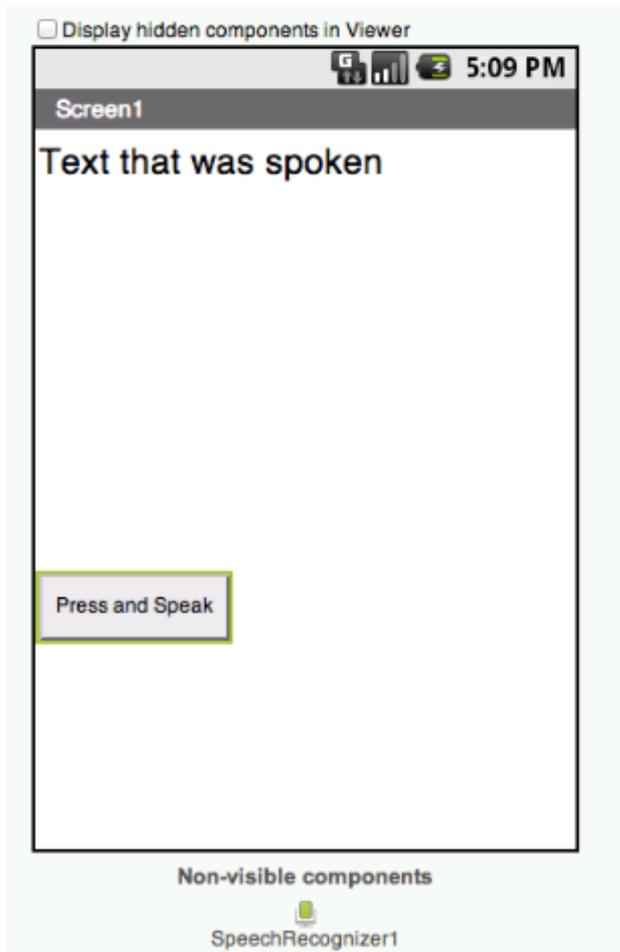
when SpeechRecognizer1 .BeforeGettingText

do set TextLabel . Text to " "



Speech Recognition

Display the text of what is being said on the phone screen.



```
when PressAndSpeakButton .Click
do call SpeechRecognizer1 .GetText
```

```
when SpeechRecognizer1 .AfterGettingText
result
do set TextLabel . Text to SpeechRecognizer1 . Result
```

```
when SpeechRecognizer1 .BeforeGettingText
do set TextLabel . Text to ""
```

When the **PressAndSpeakButton** is clicked the **SpeechRecognizer** event is called and is ready for you to speak.

The **BeforeGettingText** event will be triggered before speech has been received and recognized. Then the **Label** will display no text on the screen.

The **AfterGettingText** event will be triggered once speech has been received and recognized. Then the **Label** will display the text on the screen.



What Does this App do?

when Texting1 .MessageReceived

number messageText

do set Texting1 . Message to " I'm driving right now, I'll text you later. "

set Texting1 . PhoneNumber to get number

call Texting1 .SendMessage

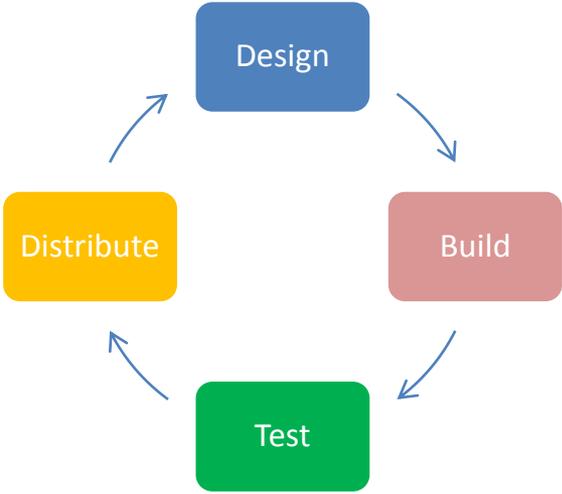
call TextToSpeech1 .Speak

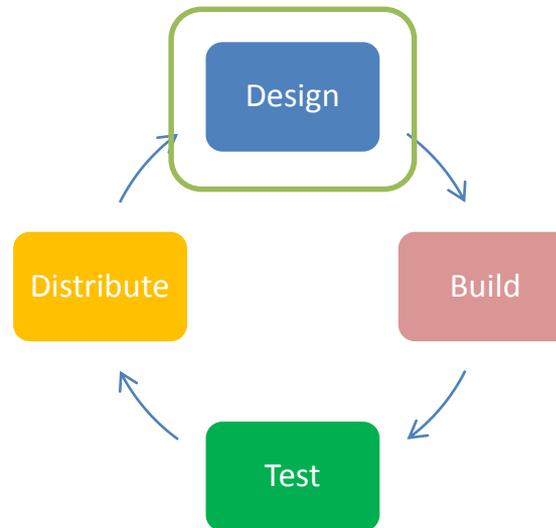
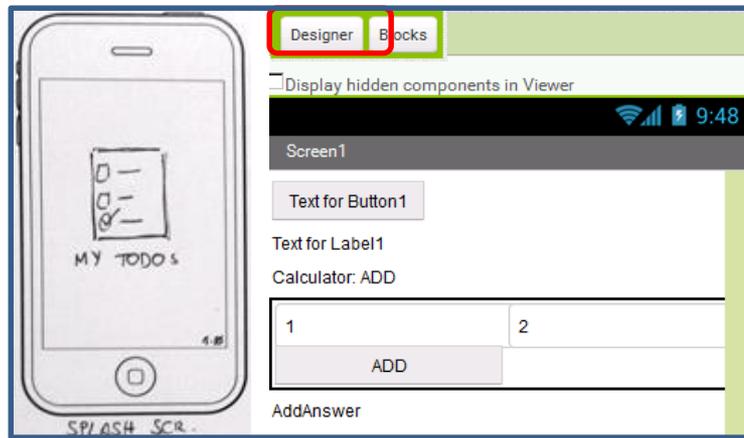
message join " message from "

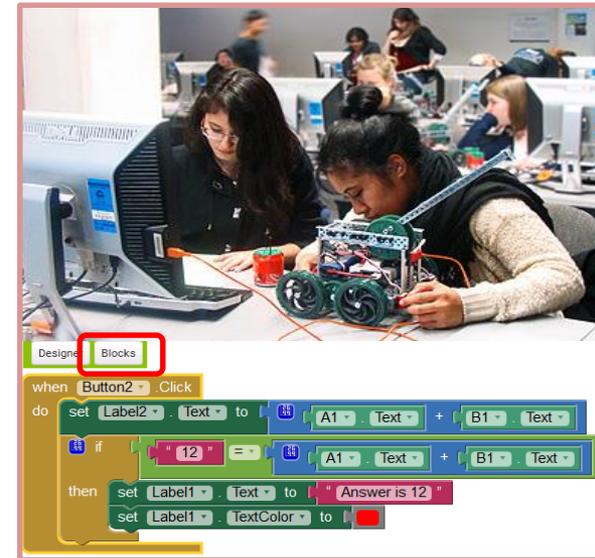
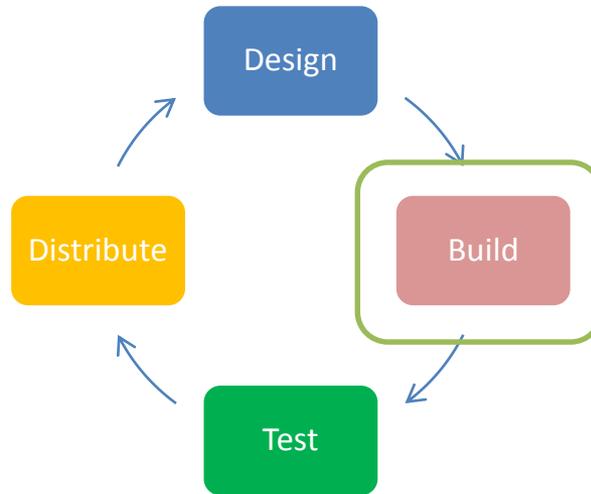
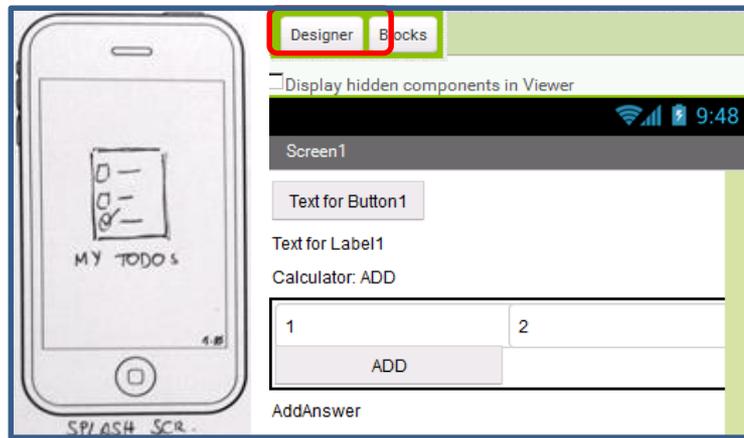
get number

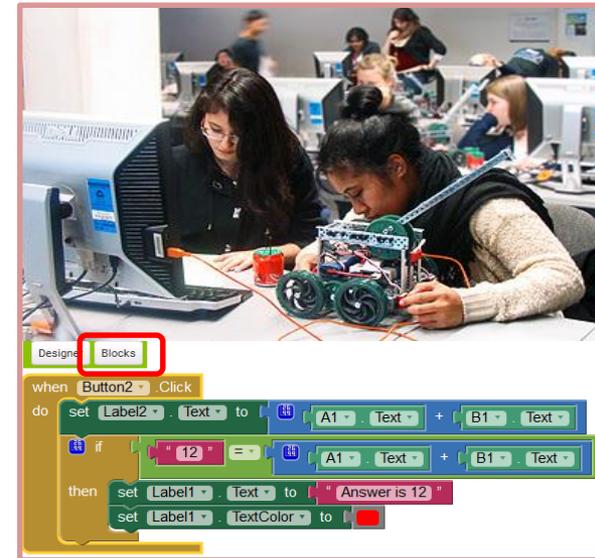
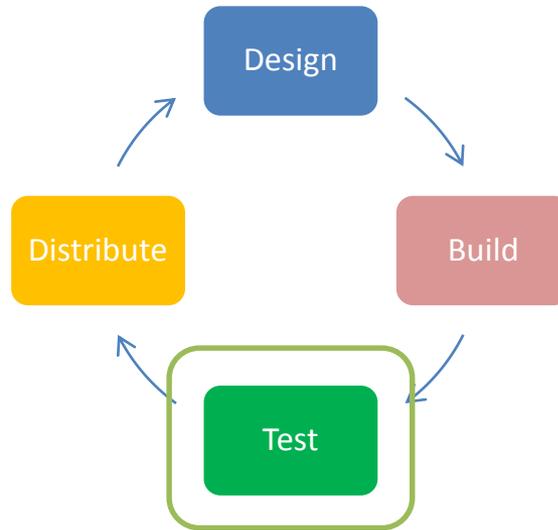
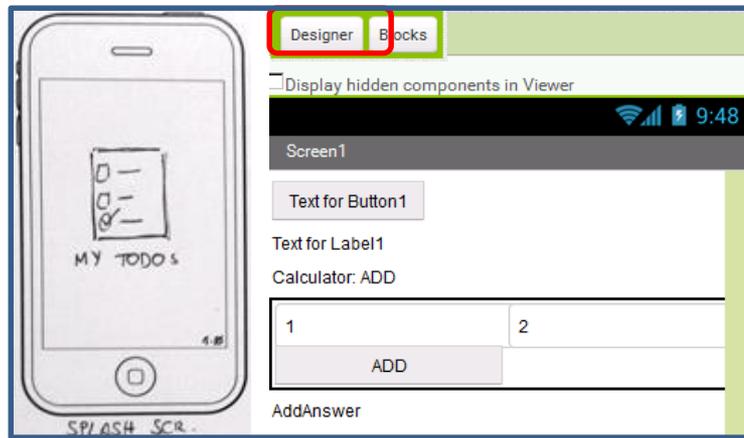
get messageText

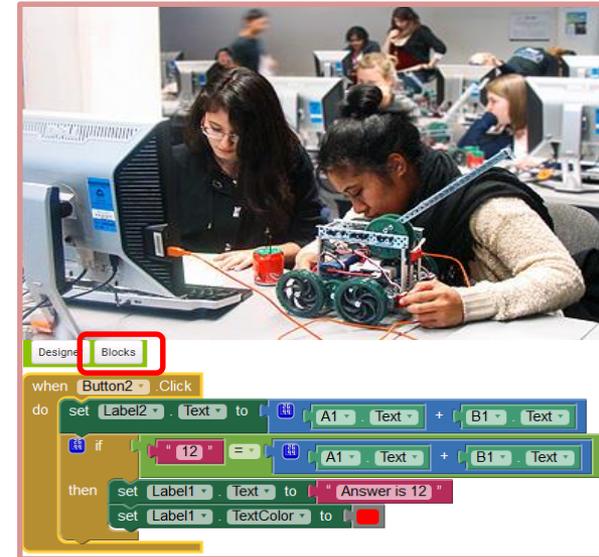
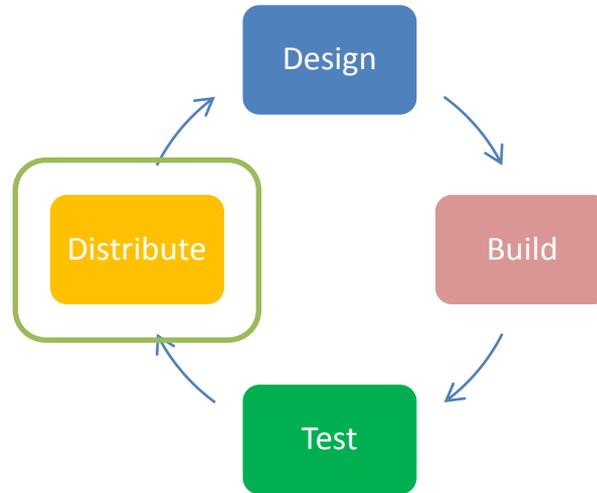
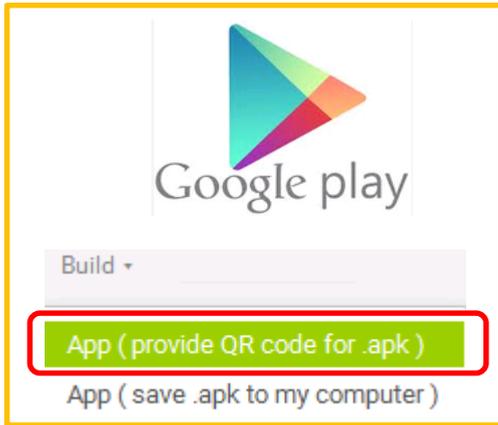
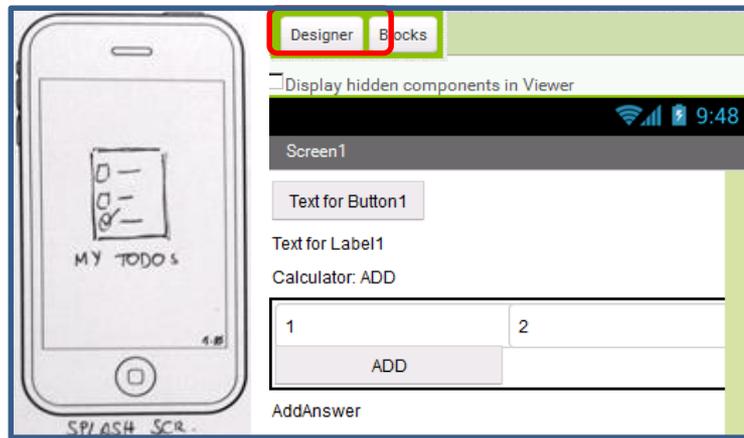
Process for creating apps











Setting Up Your Phones

Getting Started

What you need:

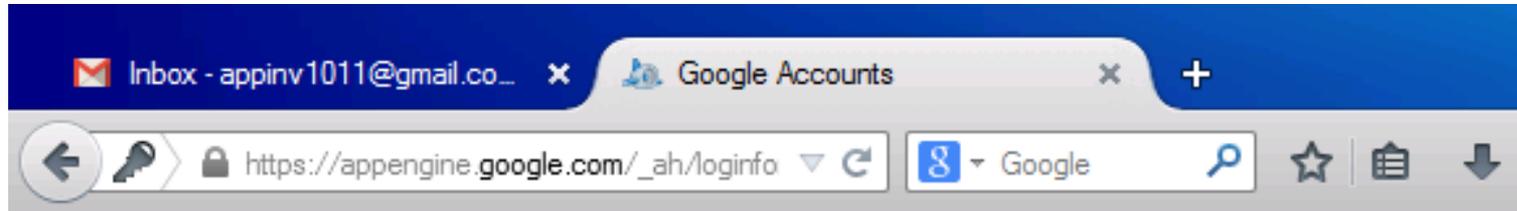
- Internet access
- Laptop (Windows or Mac)
- Gmail account
- Computer with Firefox 3.6/Chrome 4.0/ Safari 5.0 web browser
- Android phone or tablet with OS 2.3 or higher

Log in to Gmail

- ➔ Go to Google and click on Sign In
- ➔ If you have an account sign in now
- ➔ Otherwise, click on New Account and create one

Go to App Inventor

ai2.appinventor.mit.edu



Google accounts

The application MIT AppInventor Version 2 is requesting permission to access your Google Account.

Please select an account that you would like to use.

🌐 appinv1011@gmail.com

Google is not affiliated with the contents of **MIT AppInventor Version 2** or its owners. If you sign in, Google will share your email address with **MIT AppInventor Version 2** but not your password or any other personal information.

Allow

No thanks

[Sign in to another account](#)

Remember this approval for the next 30 days

Welcome to MIT App Inventor 2

Welcome to the nb146j Release!

Read the [Release Notes](#) for more information.

This release uses Companion version 2.35

Got an Android phone or tablet? Find out how to
[Set up and connect an Android device.](#)

Don't have an Android device? Find out how to
[Set up and run the Android emulator.](#)

Continue

Do Not Show Again

Description of MIT App Inventor

From this Site you can access MIT App Inventor, which lets you develop applications for Android devices using a web browser and either a connected phone or emulator. You can also use the Site to store your work and keep track of your projects. App Inventor was originally developed by Google. The Site also includes documentation an educational content, and this is being licensed to you under the Creative Commons Attribution 3.0 Unported license ([CC BY 3.0](#)).

Account Required for Use of MIT App Inventor

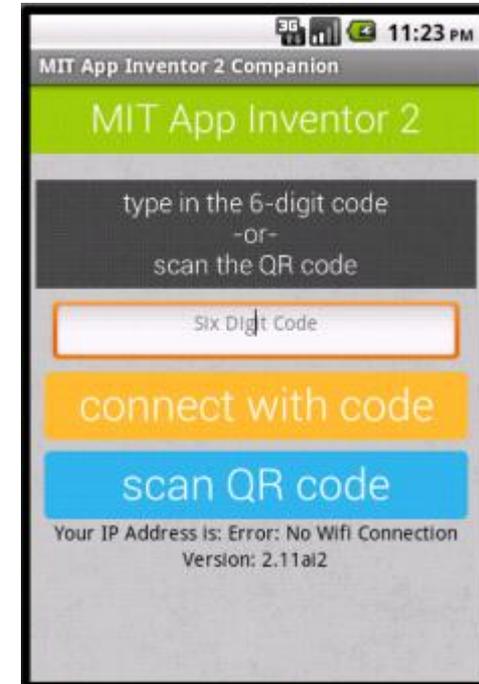
In order to log in to MIT App Inventor, you need to use a Google account. Your use of that account is subject to Google's Terms of Service for



I accept the terms of service!

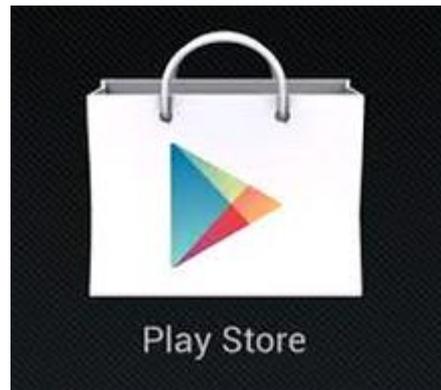
AI Companion on your phone or tablet

- ➔ Can see changes made in real-time
- ➔ Operate and test your app with your actual phone
- ➔ GPS, Camera, and accelerometer might work
- ➔ Application is only temporarily running on the phone. Application is not stored in the phone



Downloading AI Companion

- ➔ Go to the Play Store in your phone



Search for “MIT app inventor”

The image shows a screenshot of the Google Play Store interface. At the top, the Google Play logo is on the left, and a search bar contains the text "mit app inventor" with a magnifying glass icon on the right. Below the search bar, a navigation menu on the left lists categories: Store, Apps, Movies & TV, Music, Books, Newsstand, and Devices. The main content area displays search results under the heading "Apps". Three app cards are visible. The first card, "MIT AI2 Companion" by MIT Center for Mobile Learning, is highlighted with a red rounded rectangle. It features a green Android robot icon with puzzle pieces. The second card, "MIT AICompanion" by MIT Center for Mobile Learning, features a green Android robot icon with a Wi-Fi symbol. The third card, "AppInventorApps" by A.I.APPS, features a blue Android robot icon with the letters "AI" on its chest. Each card shows a 5-star rating and the word "FREE".

Google play

mit app inventor

Search All results

Apps

MIT AI2 Companion
MIT Center for Mobile L
★★★★★ FREE

MIT AICompanion
MIT Center for Mobile L
★★★★★ FREE

AppInventorApps
A.I.APPS
★★★★★ FREE

My Play activity

Creating your first app

“Talk to Me”

Let's get started

The image shows a browser window with the MIT App Inventor 2 interface. The browser tabs include 'Inbox - appinv1011@gmail.co...' and 'MIT App Inventor 2'. The address bar shows 'ai2.appinventor.mit.edu'. The page header features the MIT App Inventor 2 logo and the text 'MIT App Inventor 2 Beta', along with navigation links for 'Projects', 'Connect', 'Build', and 'Help'. A green bar contains two buttons: 'Start new project' (highlighted with a red box) and 'Delete Project'. Below this, a 'My Projects' section is visible. A modal dialog box titled 'Create new App Inventor project' is open, containing a 'Project name:' label and a text input field with the value 'TalkToMe' (highlighted with a red box). At the bottom of the dialog are 'Cancel' and 'OK' buttons.

Getting Started

The screenshot shows the 'TalkToMe' application designer interface. At the top, there is a green header bar with the application name 'TalkToMe' on the left, a dropdown menu set to 'Screen1', and buttons for 'Add Screen ...' and 'Remove Screen'. On the far right of this header, a 'Designer' button is highlighted with a red rectangular box.

Below the header, the interface is divided into four main panels:

- Palette:** Labeled 'User Interface', it contains a list of UI components: Button, CheckBox, Clock, Image, Label, ListPicker, Notifier, PasswordTextBox, Slider, TextBox, and WebViewer. A blue callout box with the text 'Palette: Choose components' is overlaid on this panel.
- Viewer:** Contains a preview of the application screen. At the top, there is a status bar with icons for Wi-Fi, signal strength, battery, and the time '9:48'. Below the status bar, the text 'Screen1' is displayed. A blue callout box with the text 'Viewer: Arrange components' is overlaid on this panel.
- Components:** Shows a hierarchical view of the components. A single component named 'Screen1' is listed. A blue callout box with the text 'Components List (Heirarchical View)' is overlaid on this panel.
- Properties:** Shows the settings for the selected 'Screen1' component. Properties include 'AlignHorizontal' (set to 'Left'), 'AlignVertical' (set to 'Top'), 'BackgroundColor' (set to 'White'), and 'BackgroundImage' (set to 'None...'). A blue callout box with the text 'Properties: Change component settings' is overlaid on this panel.

Palette

1 **User Interface**

- Button
- CheckBox
- Clock
- Image
- Label
- ListPicker
- Notifier
- PasswordTextBox

Viewer

Display hidden components in Viewer

Screen 1

Text for Button 1 3

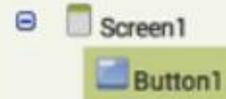
1. Click and hold on "Button"
2. Drag over to the Viewer and drop.
3. A Button appears on the Viewer.

Viewer

Display hidden components in Viewer



Components



Properties

Button1

BackgroundColor

Default

Enabled



FontBold



FontItalic



FontSize

14.0

FontTypeface

default

Image

None...

Shape

default

ShowFeedback



Text

Talk T

A context menu for the 'Talk To Me' button. It contains the following items: 'Shape' with a dropdown menu set to 'default'; 'ShowFeedback' with a checked checkbox; 'Text' with a text input field containing 'Talk To Me'; 'TextAlignment' with a dropdown menu set to 'center'; and 'TextColor' with a color selection area set to 'Default'. A 'Delete' button is also visible to the left of the menu.

Rename

Delete

Media

Palette

User Interface

Layout

Media

- Camcorder
- Camera
- ImagePicker
- Player
- Sound
- SoundRecorder
- SpeechRecognizer
- TextToSpeech**
- VideoPlayer

Drawing and Animation

Sensors

Social

Storage

Connectivity

LEGO® MINDSTORMS®

Viewer

Display hidden components in Viewer

Screen1

Talk To Me

Drop here.
Component will
automatically
show up in
Non-visible
components
area
below

Non-visible components

TextToSpeech1

Components

- Screen1
 - Button1
 - TextToSpeech1

Rename Delete

Media

Upload File ...

Designer

Blocks

Components

- Screen1
 - Button1

Properties

Button1

BackgroundColor

Default

Enabled

FontBold

FontItalic



Blocks

- ▢ Built-in
 - ▢ Control
 - ▢ Logic
 - ▢ Math
 - ▢ Text
 - ▢ Lists
 - ▢ Colors
 - ▢ Variables
 - ▢ Procedures
- ▢ Screen1
 - ▢ Button1
 - ▢ TextToSpeech1
- ▢ Any component



Viewer

Built-in Blocks are always available. They handle things like math, text, logic, and control.

Component Blocks correspond to the components you've chosen for your app.

 0  0

Show Warnings

Workspace where you assemble the blocks into a program.

Trash for deleting unneeded blocks.



Blocks

Built-in

Control

Logic

Math

Text

Lists

Colors

Variables

Procedures

Screen 1

Button1

TextToSpeech1

Any component

Viewer

when Button1 ▾ .Click

do

when Button1 ▾ .GotFocus

do

when Button1 ▾ .LongClick

do

when Button1 ▾ .LostFocus

do

when Button1 ▾ .BackgroundColor

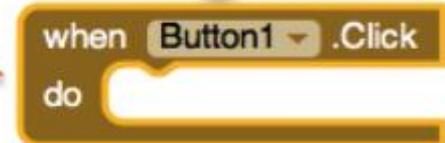
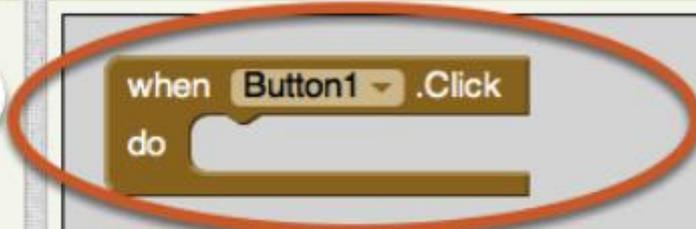
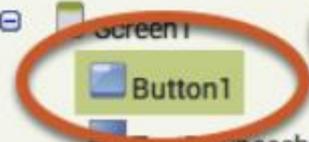
3

when Button1 ▾ .Click

do

2

1



TalkToMe

Screen1 ▾

Add Screen ...

Remove Screen

Blocks

Built-in

Control

Logic

Math

Text

Lists

Colors

Variables

Procedures

Screen1

Button1

TextToSpeech1

Any component

Viewer

when TextToSpeech1 ▾ .AfterSpeaking

result

do

when TextToSpeech1 ▾ .BeforeSpeaking

do

call TextToSpeech1 ▾ .Speak

message

TextToSpeech1 ▾ . Country ▾

set TextToSpeech1 ▾ . Country ▾ to

TextToSpeech1 ▾ . Language ▾

when Button1 ▾ .Click

do

call TextToSpeech1 ▾ .Speak

message

1

2

3

Blocks

Built-in

Control

Logic

Math

Text

Lists

Colors

Variables

Procedures

Screen1

Button1

TextToSpeech1

Viewer

The Viewer displays a script for a button click event:

```
when Button1 Click  
do  
  call TextToSpeech1 .Speak  
  message
```

The 'Text' block in the palette is circled in orange. An orange arrow points from this block to the 'Text' block in the script, which is also circled in orange. Other blocks in the palette include 'join', 'length', 'is empty', 'compare texts', and 'trim'.

when **Button1** .Click

do call **TextToSpeech1** .Speak

message

" Congratulations! You've made your first app. "



MIT App Inventor 2
Beta

Project ▾

Connect ▾

Build ▾

TalkToMe

Screen1 ▾

AI Companion

Emulator

USB

Reset Connection

Palette

User Interface



Button



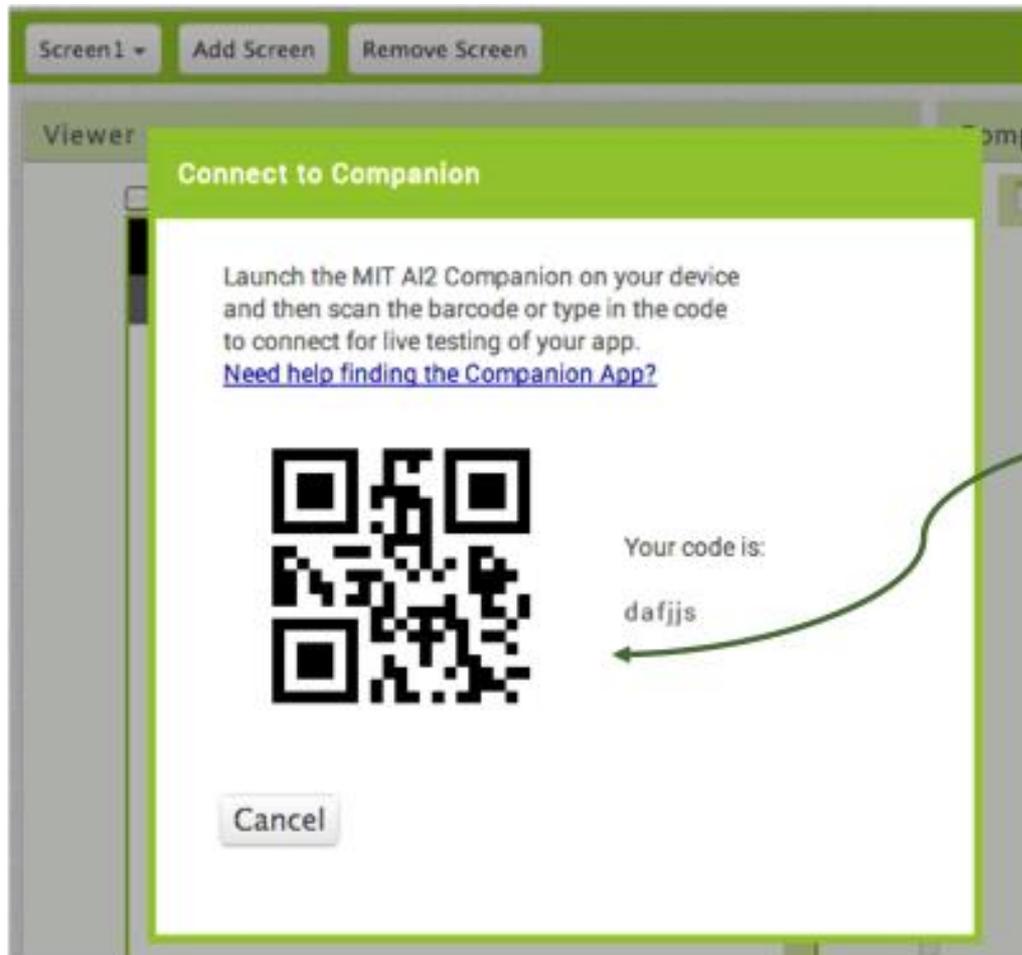
Check Box

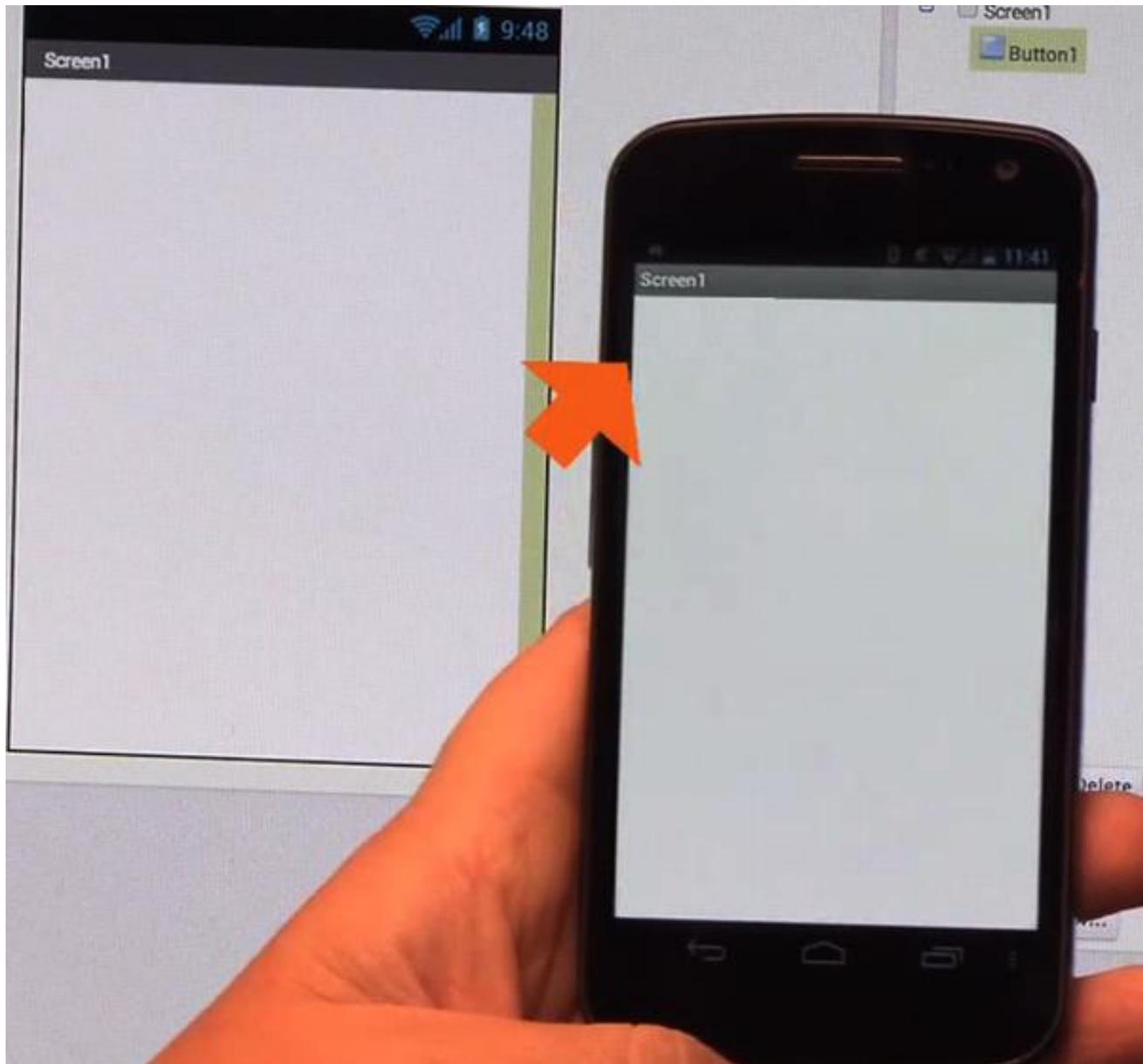


Viewer

Display hidden

Screen1





TalkToMe

Blocks

☰ Built-in

-  Control
-  Logic
-  Math
-  Text
-  Lists
-  Colors
-  Variables
-  Procedures

☰  Screen 1

 TextBox1

My projects

Start new project

Import project (.aia) from my computer ...

Import project (.aia) from a repository ...

Delete Project

Save project

Save project as ...

Checkpoint

Export selected project (.aia) to my computer

Export all projects

Import keystore

Export keystore

Delete keystore

Step 2: Inputting what to say

The screenshot displays an IDE interface with three main panels: Palette, Viewer, and Components.

- Palette:** Labeled "User Interface", it lists various UI components. The "TextBox" component is circled in red, and a red arrow points from it to the Viewer.
- Viewer:** Shows a mobile application screen titled "Screen1". At the top, there is a status bar with icons for Wi-Fi, signal strength, and battery, along with the time "9:48". Below the status bar, a white rectangular area is highlighted with a green border, representing the area where the TextBox is being added. Below this area is a button labeled "Talk To Me".
- Components:** A tree view showing the hierarchy of components. "Screen1" is the root, containing "TextBox1", "Button1", and "TextToSpeech1".

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables
- Procedures

Screen1

- TextBox1
- Button1
- TextToSpeech1
- AccelerometerSensor1

Any component

Viewer

set TextBox1 . Height to

TextBox1 . Hint

set TextBox1 . Hint to

TextBox1 . MultiLine

set TextBox1 . MultiLine to

TextBox1 . NumbersOnly

set TextBox1 . NumbersOnly to

TextBox1 . Text

set TextBox1 . Text to

TextBox1 . TextColor



```
when Button1 .Click
do call TextToSpeech1 .Speak
message TextBox1 . Text
```

“ Congratulations! You've made you

Blocks

Built-in

Control

Logic

Math

Text

Lists

Colors

Variables

Procedures

Screen1

HorizontalArrangement1

Viewer

initialize global name to

get

set to

initialize local name to .Click

in do call TextToSpeech1 .Speak

message

TextToTalk . Tex

initialize local name to

in



TalkToMe

Screen1 ▾

Add Screen ...

Remove Screen

Blocks

Built-in

- Control
- Logic
- Math
- Text
- Lists
- Colors
- Variables
- Procedures

Viewer

initialize global name to

get

set to

initialize local name to

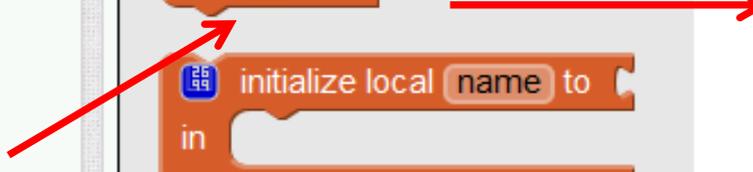
in

initialize global textToSpeak to

" "

set to

global textToSpeak



- Blocks
- Built-in
 - Control
 - Logic
 - Math
 - Text
 - Lists
 - Colors
 - Variables

Viewer

```
initialize global name to  
get  
set to  
initialize local name to  
in
```

```
initialize global textToSpeak to " "  
when Button1 .Click  
do  
  set global textToSpeak to TextBox1 . Text  
  call TextToSpeech1 .Speak  
  message [get] global textToSpeak
```

global textToSpeak





TalkToMe

Screen1 v Add Screen ... Remove Screen

Blocks

- Built-in
 - Control
 - Logic
 - Math
 - Text
 - Lists
 - Colors
 - Variables
 - Procedures
- Screen1
 - TextBox1
 - Button1
 - TextToSpeech1
- Any component

Viewer

```
initialize global textToSpeak to ""  
  
when Button1 .Click  
do  
  set global textToSpeak to TextBox1 . Text  
  call TextToSpeech1 .Speak  
    message get global textToSpeak  
  
  " Congratulations! You've made your first app. "
```

TalkToMe

Blocks

☰ Built-in

-  Control
-  Logic
-  Math
-  Text
-  Lists
-  Colors
-  Variables
-  Procedures

☰  Screen 1

-  TextBox1

My projects

Start new project

Import project (.aia) from my computer ...

Import project (.aia) from a repository ...

Delete Project

Save project

Save project as ...

Checkpoint

Export selected project (.aia) to my computer

Export all projects

Import keystore

Export keystore

Delete keystore

Lunch

Lunch Instructions

- ➔ You will be given a ticket to eat at the University Residence Cafeteria
- ➔ You are expected to get back to this room by 1:00pm
- ➔ Stick together – we will go over in groups

Have a good lunch

Making a slideshow

Designing your screen

- ➔ For this app you want to create a slideshow by allowing your user to be able to navigate through images
- ➔ When you design your screen, you will need to put in an image
- ➔ You also need 'previous' and 'next' buttons for your user to click. This part will be done in the designer

SlideShow Screen1 Add Screen ... Screen2 Screen3 **Step 6** Designer Blocks

Palette

User Interface

- Button **Step 3**
- CheckBox
- DatePicker
- Image **Step 1**
- Label
- ListPicker
- ListView
- Notifier
- PasswordTextBox
- Slider
- Spinner
- TextBox
- TimePicker
- WebView

Layout **Step 2**

Media

Drawing and Animation

Sensors

Screen1

Display hidden components in Viewer

Check to see Preview on Tablet size.

Screen1

9:48

Technovation
iridescent

Previous Next

Components

- Screen1
 - Image1
 - HorizontalArrangement1
 - Previous
 - Next

Properties

Image1

Height

Fill parent...

Width

Fill parent...

Picture

TCLoGo.png...

RotationAngle

0.0

ScalePictureToFit

Visible

Rename Delete

Media

TCLoGo.png

Upload File ... **Step 4**

Step 5

1. Add an image to your screen.
 - Make it fit your screen by changing the width and the height!
 - We choose to make our image “fill parent”. When you select “fill parent” you are telling the component to fill the space of whatever it is in.
2. Add a horizontal layout to your screen below the image.
3. Add two buttons into the horizontal layout.
 - Click on the buttons in the component menu
 - Edit the text to make the buttons say ‘previous’ and ‘next’.

The screenshot shows the Android Studio IDE with the following components and annotations:

- Step 1:** Points to the **Image** component in the **User Interface** section of the **Palette**.
- Step 2:** Points to the **Layout** category in the **Palette**.
- Step 3:** Points to the **Button** component in the **User Interface** section of the **Palette**.
- Step 4:** Points to the **Upload File ...** button in the **Media** section, specifically for **TCLoGo.png**.
- Step 5:** Points to the **Picture** property in the **Properties** panel, which is set to **TCLoGo.png...**.
- Step 6:** Points to the **Screen1** dropdown menu in the top toolbar.

The central **Designer** view shows a mobile screen with the **Technovation iridescent** logo and two buttons labeled **Previous** and **Next**. The **Components** panel on the right shows a tree view with **Image1**, **HorizontalArrangement1**, **Previous**, and **Next**. The **Properties** panel for **Image1** shows **Height** and **Width** set to **Fill parent...**, and **Picture** set to **TCLoGo.png...**.

4. Upload a picture you want to use in your slide show!
5. Select the picture you want to use on the first screen of your slide show.
 - Click on “Image1” in the “Components” menu and then “Picture” in the properties menu. You can then select a picture you uploaded.
6. Add two more screens.
 - For now you do not need to do anything with those screens, but after you code the blocks you will add buttons and images to them just like you did for this screen

The screenshot shows the SlideShow Designer interface with several annotations in maroon text and arrows:

- Step 1:** Points to the "Image" component in the "User Interface" section of the Palette.
- Step 2:** Points to the "Layout" section in the left sidebar.
- Step 3:** Points to the "Screen1", "Screen2", and "Screen3" dropdown menu.
- Step 4:** Points to the "Upload File ..." button in the "Media" section at the bottom.
- Step 5:** Points to the "Picture" property field in the "Properties" panel, which shows "TCLogo.png...".
- Step 6:** Points to the "Add Screen ..." button in the top header.

The central preview window shows a slide with the "Technovation iridescent" logo and "Previous" and "Next" buttons. The "Components" panel on the right shows a tree view with "Image1" selected under "Screen1". The "Properties" panel on the far right shows the properties for "Image1", including "Picture" (TCLogo.png...), "RotationAngle" (0.0), and "Visible" (checked).

Click on "Blocks" Button

7. Click on button you named 'previous' to see all the event handlers you can use. Grab the 'when button.click' event handler and drag it to your workspace.
8. Click on 'control' and find the 'open another screen screenName' block.
9. Get an empty text box and click it into the "open another screen screenName" block. Type the name of the last screen that will be in your slide show. For us, it was Screen3.
10. Click on your next button and grab the "when button.click" event handler again
11. Get another "open another screen screenName" block from control.
12. Get an empty text block and type in the next screen that will appear in your slide show. (For us, that was Screen2.)

SlideShow | Screen1 | Add Screen ... | Remove Screen | Designer | Blocks

Blocks

- Built-in
 - Control
 - Logic
 - Math
 - Text
 - Lists
 - Colors
 - Variables
 - Procedures
- Screen1
 - Image1
 - HorizontalArrangement1
 - Previous
 - Next
- Any component

Viewer

when Previous .Click
do open another screen screenName " Screen3 "

when Next .Click
do open another screen screenName " Screen2 "

Steps 8 & 11

Steps 9 & 12

Steps 7 & 10

Show Warnings

Using an image on your phone

The screenshot displays the App Inventor web interface in a browser window. The address bar shows the URL `ai2.appinventor.mit.edu/?locale=en#6427204498751488`. The interface is divided into several sections:

- Layout:** A central preview window titled "Picture1" showing a mobile app interface with a "Load Slideshow" button and "Previous" and "Next" navigation buttons. A "Check to see Preview on Tablet size." link is visible above the preview.
- Media:** A left-hand menu listing various media components. The "ImagePicker" component is circled in blue. Other components include Camcorder, Camera, Player, Sound, SoundRecorder, SpeechRecognizer, TextToSpeech, VideoPlayer, and YandexTranslate.
- Properties:** A right-hand panel showing the properties for the selected "ImagePicker1" component. Properties include BackgroundColor (Default), Enabled (checked), FontBold, FontItalic, FontSize (14.0), FontTypeface (default), Height (Automatic...), Width (Automatic...), Image (None...), Shape (default), and ShowFeedback (checked).
- Bottom Bar:** A taskbar at the bottom of the browser window showing open files like "blocks.png" and "Technovation Log...jpg".

Setting the image

The screenshot displays the MIT App Inventor web interface. At the top, a browser window shows the URL `ai2.appinventor.mit.edu/?locale=en#6427204498751488`. The interface is divided into several sections:

- Slideshow:** A green header bar with a dropdown menu set to "Screen1", buttons for "Add Screen ..." and "Remove Screen", and tabs for "Designer" and "Blocks".
- Blocks:** A left-hand sidebar containing a "Built-in" category with sub-categories like Control, Logic, Math, Text, Lists, Colors, Variables, and Procedures. Below this, a project-specific tree shows "Screen1" with components "Logo", "ImagePicker1", "HorizontalArrangement1", "Previous1", and "Next1".
- Viewer:** The main workspace showing three logic blocks:
 - when Next1 .Click** (yellow block) with a **do open another screen** block (orange) where `screenName` is set to `"Screen2"`.
 - when Previous1 .Click** (yellow block) with a **do open another screen** block (orange) where `screenName` is set to `"Screen3"`.
 - when ImagePicker1 .AfterPicking** (yellow block) with a **do set Logo . Picture to ImagePicker1 . Selection** block (green).
- Warnings:** A small area at the bottom left of the viewer showing 0 yellow and 0 red warning icons, with a "Show Warnings" button.
- Image Placeholder:** A large, faint image of a blue backpack is visible in the background of the viewer area.

At the bottom of the screen, a Windows taskbar shows various application icons and the system clock indicating 10:26 PM on 10/14/2016.

Types of Data

Types of Data?

	WhatsApp	Angry Birds	Slideshow
About	Message and call your friends	Game: shoot birds at pigs	Cycle through images (you made this)
Types of data	<ul style="list-style-type: none">• Your username• Your friend's usernames• The message you want to send• What time it is• Your location	<ul style="list-style-type: none">• Your score• levels you've completed	<ul style="list-style-type: none">• Your favorite images

Types of Data

➔ Numbers



➔ Strings



➔ Booleans



Variables

➔ Variable: Data that can change in value

- Your Age (string, number, or boolean?)
- Your Address (string, number, or boolean?)
- Student[yes or no] (string, number, or boolean?)

Variables

Variable: Data that can change in value

- Your Age (string, **number**, or boolean?) 
- Your Address (string, number, or boolean?)
- Student[yes or no] (string, number, or boolean?)

Variables

➔ Variable: Data that can change in value

- Your Age (string, **number**, or boolean?) 
- Your Address (**string**, number, or boolean?) 
- Student[yes or no] (string, number, or boolean?)

Variables

➔ Variable: Data that can change in value

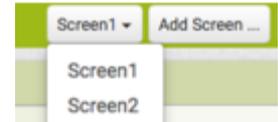
- Your Age (string, **number**, or boolean?) 
- Your Address (**string**, number, or boolean?) 
- Student[yes or no] (string, number, or **boolean**?) 

Variables

➔ Variable: Data that can change in value

- Your Age (string, **number**, or boolean?) 
- Your Address (**string**, number, or boolean?) 
- Student[yes or no] (string, number, or **boolean**?) 

➔ Local variables work only on one screen



[Screen1]VeribA=3; [Screen2]VeribB=4; [Screen3] VeribA+VeribB=Error

➔ Global variables can be shared on all screens

[Screen1]Global VeribA=3; [Screen2]Global VeribB=4; [Screen3]VeribA+VeribB=7

initialize global VeribA to 

App Inventor Colored Dots

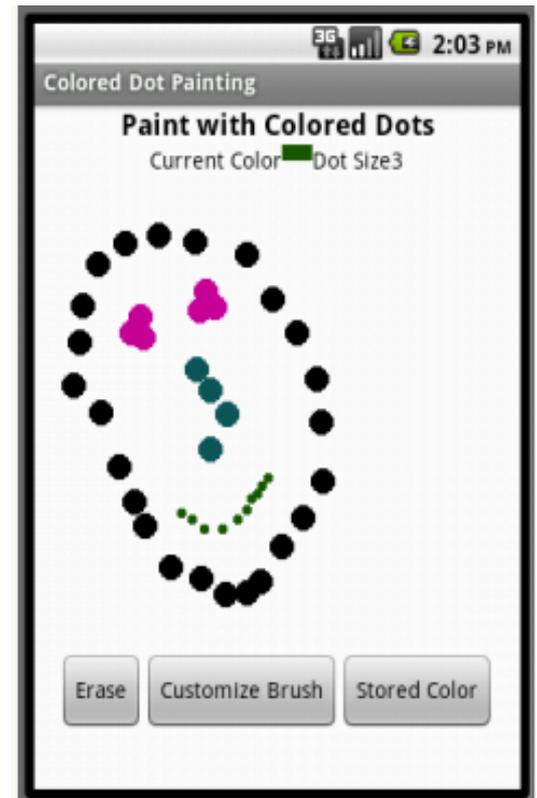
AI: Colored Dots

(Create multiple screens)

<http://appinventor.mit.edu/explore/ai2/colored-dots.html>

The [Colored Dot tutorial](#) teaches you how to create apps that have multiple screens. You'll learn how to:

- ➔ make an app with multiple screens
- ➔ pass values from one screen to another using TinyDB
- ➔ how to fill and use the ListPicker element



Starting Screen

The image shows a mobile application development IDE interface. The main window is titled "Viewer" and displays a preview of a mobile application screen. The screen has a status bar at the top with the time 9:48 and signal strength indicators. Below the status bar is a header labeled "Screen1". The main content area contains a title "Paint With Colored Dots" in a yellow box, followed by two input fields: "Current Color" and "Dot Size". A large white canvas is in the center, with a small brush icon in the middle. At the bottom of the canvas are three buttons: "Erase", "Customize Brush", and "Stored Color".

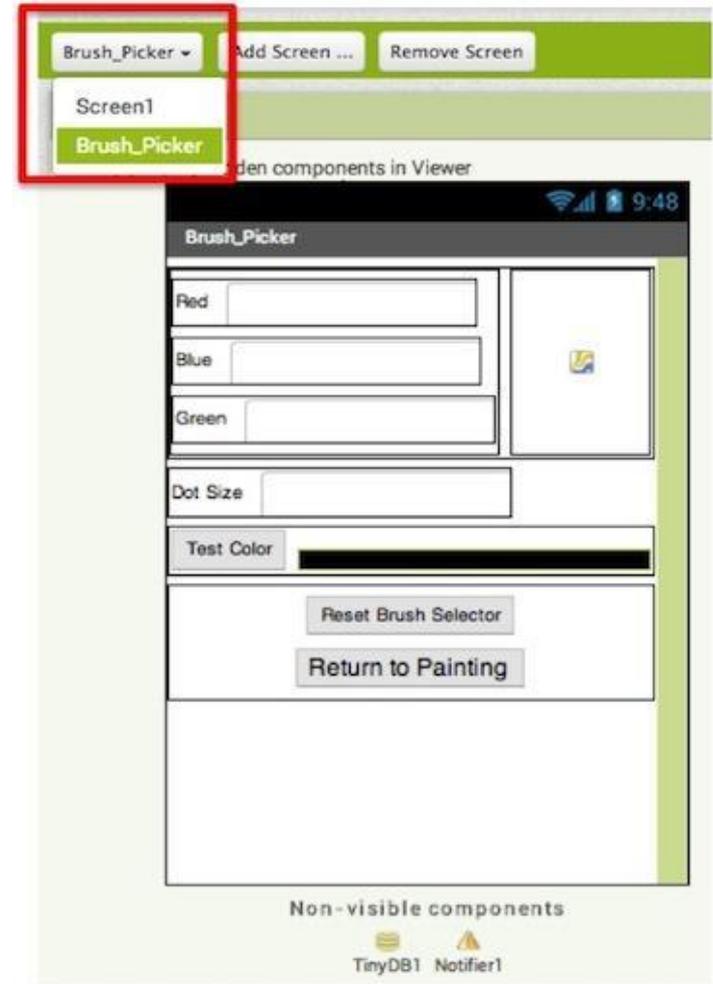
Below the viewer, there is a section for "Non-visible components" which includes a "TinyDB1" component represented by a database icon.

The right-hand side of the IDE is titled "Components" and shows a hierarchical tree view of the application's components. The tree starts with "Screen1", which contains a "titleLabel" component. Below this is a "HorizontalArrangement1" container, which includes "ColorLabel", "ColorSample", "DotSizeLabel", and "DotSizeValue" components. Another "HorizontalArrangement2" container follows, containing "EraseButton", "openBrushPicker", and "ListPicker1" components. At the bottom of the tree is a "TinyDB1" component. At the bottom of the "Components" panel are "Rename" and "Delete" buttons.

At the very bottom of the IDE interface is a "Media" section with an "Upload new..." button.

Multiple Screens

- ➔ You can add screens in the designer and use the screen transitions in blocks editor to decide which screen to go to next
 - For Example: pushing the menu button go to the menu screen
- ➔ Screen 1 will always be the screen the app starts on – it's probably best to make it a welcome screen



Starting with one colour



```
initialize global currentColor to   
  
when Screen1.Initialize  
do  
  call TinyDB1.StoreValue  
  tag 0  
  valueToStore "Black"  
  call TinyDB1.StoreValue  
  tag "Black"  
  valueToStore get global currentColor  
  set ColorSample.BackgroundColor to   
  set DotSizeValue.Text to 3  
  set ListPicker1.Elements to call populateList
```

```
when EraseButton.Click  
do call Canvas1.Clear  
  
when Canvas1.Touched  
x y touchedAnySprite  
do  
  set Canvas1.PaintColor to get global currentColor  
  call Canvas1.DrawCircle  
  centerX get x  
  centerY get y  
  radius DotSizeValue.Text  
  fill true
```

- ➔ On Screen 1 set up your starting colour
- ➔ Set up what happens when the screen is touched

TinyDB

- ➔ Besides opening screens and returning values, the different screens in a multiple screen app can communicate through TinyDB. To do this, give every screen its individual TinyDB component.
- ➔ ColoredDots uses TinyDB to let you name the colors you create and save them for later use. The saving and naming will be done in Brush_Picker

```
initialize global counter to 0
initialize global tinyDBList to make a list

to populateList
  result
  do
    while test
      get global counter >= 0 and get global counter < get global numberOfColors
      do
        add items to list list
        item
        get global tinyDBList
        call TinyDB1 .GetValue
        tag
        get global counter
        valueIfTagNotThere " "
        set global counter to
        get global counter + 1
      result
      get global tinyDBList
```

Getting ready to add colours

initialize global `numberOfColors` to `1`

when `openBrushPicker` .Click

do open another screen with start value `screenName` `"Brush_Picker"`
startValue `make a list` `get global numberOfColors`
`get global currentColor`
`DotSizeValue` . `Text`

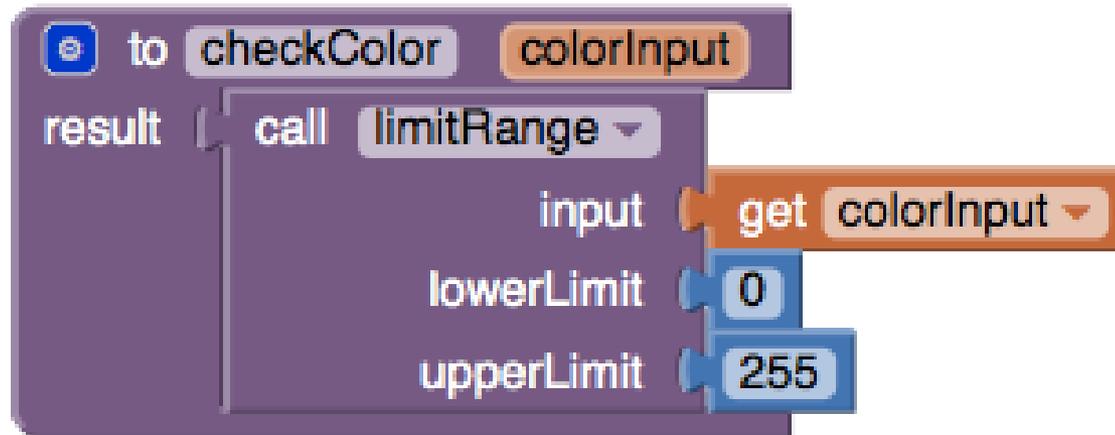
when `Screen1` .OtherScreenClosed

`otherScreenName` `result`

do `set DotSizeValue` . `Text` to `select list item list` `get result`
index `2`
`set global currentColor` to `select list item list` `get result`
index `1`
`set global numberOfColors` to `select list item list` `get result`
index `3`
`set ColorSample` . `BackgroundColor` to `get global currentColor`

Brush Picker

- The main job of Brush_Picker is to create a color from the red-green-blue values entered in the text boxes and provide that color to Screen1.
- One thing Brush_Picker needs to check is that it's using good values for colors and dot size. Each of the red, green, blue values should be a number between 0 and 255.



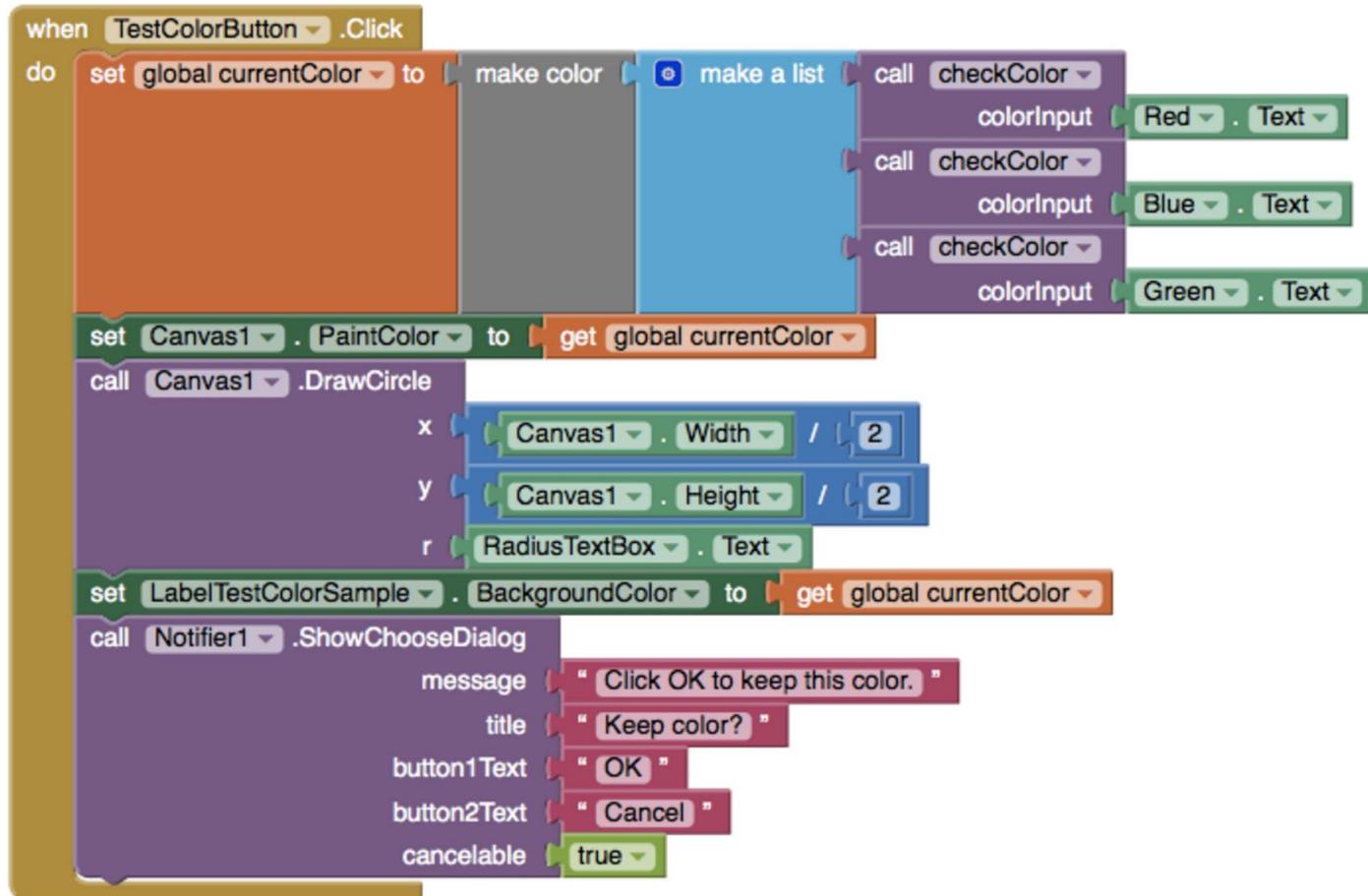
Setting up the brush picker

```
initialize global numberOfColors to 0
initialize global currentColor to 

when Brush_Picker.Initialize
do
  set global currentColor to select list item list get start value index 2
  set global numberOfColors to select list item list get start value index 1
  set RadiusTextBox.Text to select list item list get start value index 3

when ReturnToPainting.Click
do
  close screen with value result
  make a list
  get global currentColor
  call limitRange
  input RadiusTextBox.Text
  lowerLimit 3
  upperLimit 50
  get global numberOfColors
```

Testing the colour



Saving the colour

```
when Notifier1 .AfterChoosing
  choice
  do
    if
      compare texts get choice = " OK "
      then
        call Notifier1 .ShowTextDialog
          message " Enter a name for this color. "
          title " Store this color. "
          cancelable true

when Notifier1 .AfterTextInput
  response
  do
    call TinyDB1 .StoreValue
      tag get global numberOfColors
      valueToStore get response
    call TinyDB1 .StoreValue
      tag get response
      valueToStore get global currentColor
    set global numberOfColors to get global numberOfColors + 1
```

Resetting colours

```
when ResetColorButton .Click
do
  set Red . Text to " "
  set Green . Text to " "
  set Blue . Text to " "
  set Red . Hint to " 0-255 "
  set Green . Hint to " 0-255 "
  set Blue . Hint to " 0-255 "
  set RadiusTextBox . Text to 3
  set LabelTestColorSample . BackgroundColor to 
  call Canvas1 .Clear
```

Selecting a stored color

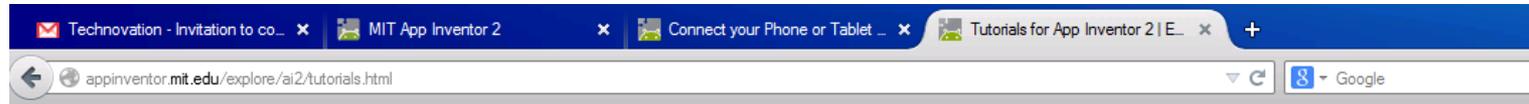
```
initialize global counter to 0
initialize global tinyDBList to make a list

to populateList
result
do while test
  get global counter >= 0 and get global counter < get global numberOfColors
  do
    add items to list list
    item get global tinyDBList
    call TinyDB1 .GetValue
    tag get global counter
    valueIfTagNotThere " "
    set global counter to get global counter + 1
  result get global tinyDBList

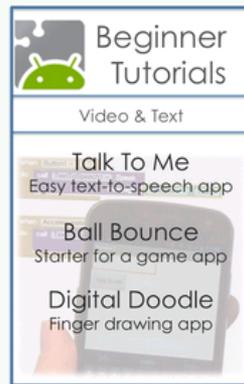
when ListPicker1 .AfterPicking
do
  set global currentColor to call TinyDB1 .GetValue
  tag ListPicker1 . Selection
  valueIfTagNotThere " "
  set ColorSample . BackgroundColor to get global currentColor
```

Continuing your learning (Tutorials)

appinventor.mit.edu/explore/ai2/tutorials



Tutorials for App Inventor 2



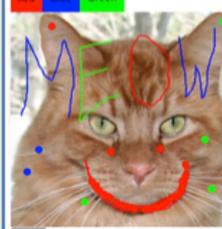
Beginner Tutorials

Video & Text

- Talk To Me
Easy text-to-speech app
- Ball Bounce
Starter for a game app
- Digital Doodle
Finger drawing app



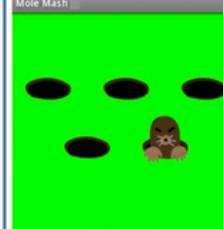
PaintPot Tutorial



Wipe



Mole Mash Tutorial



Score: 2



appinventor.org
app building for everyone.

Tutorials • Online Book
Course-in-a-box

App Inventor

Create Your Own
Android Apps

David Wolber, Hal Abelson,
Ellen Spertus & Liz Looney

O'REILLY

There are many more tutorials available below. Scroll down to browse the list, or check the appropriate boxes and click "Filter":

Filter by Tutorial Topic

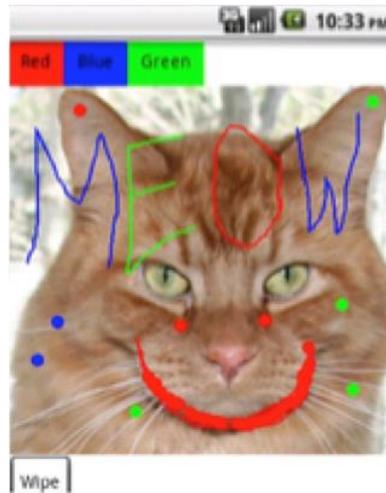
- | | | | |
|---|---|---|---------------------------------------|
| <input type="checkbox"/> Sprites | <input type="checkbox"/> Drawing Canvas | <input type="checkbox"/> Multiple Screens | <input type="checkbox"/> Clock Timer |
| <input type="checkbox"/> Game | <input type="checkbox"/> SMS Texting | <input type="checkbox"/> Camera | <input type="checkbox"/> Video |
| <input type="checkbox"/> Activity Starter | <input type="checkbox"/> ListPicker | <input type="checkbox"/> Accelerometer | <input type="checkbox"/> File Sharing |
| <input type="checkbox"/> Data Storage | <input type="checkbox"/> External API | <input type="checkbox"/> Location Sensor | <input type="checkbox"/> GPS |
| <input type="checkbox"/> NFC (Near Field Comm.) | | | |

Tips for App Development

- ➔ Attention to detail is important
- ➔ Time will be spent troubleshooting & testing
 - Don't let this discourage you
- ➔ Enjoy the challenge, impress your friends, and never give up on troubleshooting the application
- ➔ Save often

appinventor.mit.edu/explore/ai2/tutorials

Have Fun!



Technovation: Next Steps

Start brainstorming ideas

- ➔ Think about issues within your community
 - School, neighbourhood, church, other communities ...
- ➔ Pull out any and all ideas
- ➔ Draw from personal experience

Themes

- ➔ [Poverty](#) - Eradicating extreme poverty, implementing social protection systems for all, and ensuring that all men and women have equal access to economic resources.
- ➔ [Environment](#) - Improving education and awareness about climate change and strengthening resilience to climate-change hazards in all countries.
- ➔ [Peace](#) - Significantly reducing violence, ending abuse of children, reducing corruption and bribery, ensuring equal access to justice for all, and ensuring public access to information.
- ➔ [Equality](#) - Ending discrimination against girls and women, enhancing the use of enabling technology to promote the empowerment of women, ensuring universal access to reproductive rights, and ensuring women's full and effective participation and opportunities for leadership.
- ➔ [Education](#) – Ensure healthy lives and promote well-being for all at all ages
- ➔ [Health](#) – Ensure inclusive and equitable quality education and promote lifelong learning opportunities for all.

Previous Technovation finalists

Apps to fight hunger

[Food Rescue by AurGrrls](#)

This app allows users to donate leftover food from movie shoots in Los Angeles to local foodbanks.

[InDaFridge by IDF](#)

This app stops people from wasting food by providing customized recipes based on what is in their refrigerator.

Apps to stop violence

[Neo Safety](#)

This app allows users to see if there has been recent crimes in the area they are currently in.

[Women Fight Back by Girls for Change From Dharavi](#)

This app allows women to report and ask for help when they are harassed or attacked.

Apps to go green

[Discardious by Team Charis](#)

This app helps users clean up their trash in their local communities by providing carts to pick up the trash.

[Loc8Don8 by California Coders](#)

This app helps people find places to donate things they don't need anymore.

Housekeeping

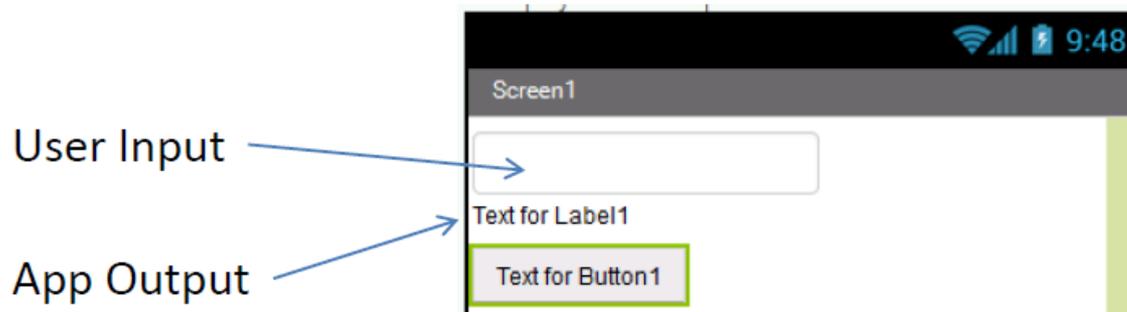
- ➔ Weekly meetings Wednesday's 7-9pm
 - IBM, 3755 Riverside Dr Feb 1st
- ➔ Join the facebook group Technovation Ottawa
- ➔ Register on Technovationchallenge.org
- ➔ Read your emails!!!!

Math App Challenge

(create a calculator)

For each member of your group work on a separate math problem

- ➔ Member 1: (Add) + 2 =
- ➔ Member 2: (Multiply) x 2 =
- ➔ Member 3: (Divide) / 2 =
- ➔ Member 4: (Subtract) - 2 =
- ➔ Member 5: (Exponent) ^ 2 =
- ➔ Member 6: (Multiply) x 4 =
- ➔ Member 7: (Subtract) - 4 =
- ➔ Member 8: (Multiply) x 5 =



Example: (Multiply)
[User Input is "3"] x 2 = [Answer is 6]

