

We have an app for that!

TechSavvy 2016 – Oakton Community College
April 16th, 2016.

Who am I?

My name is
Jeny

- ▶ I am a Computer Security Analyst at Fermilab.
- ▶ (2011) Computer Scientist
 - ▶ Developer & Antivirus Team Leader at ZenOK
 - ▶ R&D&I Leader at 2Secure.
- ▶ (2014) Master in Computer Science
- ▶ (2014) Visiting Scientist for Intensity Frontier experiments at Fermilab.
- ▶ (2015) Instructor for Girls Who Code at Metea Valley High School.



girls who
code

Goal

- ▶ The purpose of this lesson is to introduce you to some of the programming basics concepts/terms, which the girls are likely to come across no matter which programming language they decide to learn.

We have an app for that!



DOWNLOAD
our APP

We have an app for that!

- Around 1300 apps (as of March 2016) are submitted to Apple App Store every day [1].
- Apps for mobile phones, laptops, desktops.
- Who make these apps?
 - Not only developers, people interested in coding
- How are these apps made?
 - Tools for different platforms and diverse programming languages.

[1] <http://www.pocketgamer.biz/metrics/app-store/>

Basic concepts

➤ Coding

- Writing code!
- Writing code tells the device what to do.
- The instructions are expressed in specific and precise steps.



MIT App Inventor

Tools

- Drag and drop components
- Similar to writing code, but instructions are expressed in blocks.



MIT App Inventor

The screenshot displays the MIT App Inventor 2 Beta web interface. At the top, the header includes the MIT App Inventor 2 Beta logo, navigation menus for Projects, Connect, Build, and Help, and user information for My Projects, Gallery, Guide, Report an Issue, English, and jeny.teheran@gmail.com.

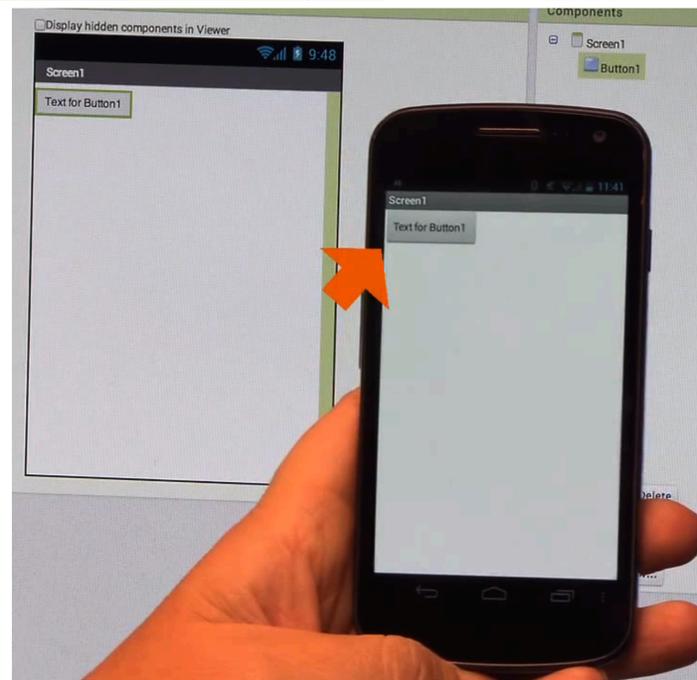
The main workspace is titled "Test" and features a toolbar with "Screen1", "Add Screen ...", and "Remove Screen" buttons. On the right side of the toolbar are "Designer" and "Blocks" tabs.

The interface is divided into four main panels:

- Palette:** Contains categories for "User Interface" (Button, CheckBox, DatePicker, Image, Label, ListPicker, ListView, Notifier, PasswordTextBox, Slider, Spinner, TextBox, TimePicker, WebView), "Layout", "Media", and "Drawing and Animation".
- Viewer:** Shows a mobile app preview for "Screen1" with a status bar at the top displaying signal strength, Wi-Fi, battery, and the time 9:48. Below the preview are checkboxes for "Display hidden components in Viewer" and "Check to see Preview on Tablet size".
- Components:** Lists the components currently on the screen, including "Screen1".
- Properties:** Shows the properties for the selected "Screen1" component, including "AboutScreen" (text input), "AlignHorizontal" (Left), "AlignVertical" (Top), "AppName" (Test), "BackgroundColor" (White), "BackgroundImage" (None...), "CloseScreenAnimation" (Default), "Icon" (None...), "OpenScreenAnimation" (Default), "ScreenOrientation" (Unspecified), and "Scrollable".

At the bottom of the Components panel, there are "Rename" and "Delete" buttons. A "Media" section is partially visible at the bottom of the interface.

Coding: Telling the device what to do



```
when Button1 .Click
do call TextToSpeech1 .Speak
   message "Congratulations! You've made your first app"
```

```
when Button1 .Click
do if is empty TextBox1 . Text
   then call TextToSpeech1 .Speak
        message "Nothing to read"
   else call TextToSpeech1 .Speak
        message TextBox1 . Text
```

Basic concepts

➤ App

- A type of software that does a certain task.
- Intended for a platform/device.
- Usually need user interaction to function while software does not necessarily have to.

➤ Software

- Global term for all the components (programs) distinct to hardware.
- Programs that tell a device what to do and how to behave.

Programming language

➤ A programming (or coding) language is a set of syntax rules that define how code should be written and formatted.

- Java
- Python
- JavaScript
- SQL
- C++, C#



Designer mode

The screenshot displays the DigitalDoodle application in its Designer mode. The interface is divided into four main sections:

- Palette:** A list of UI components categorized into User Interface (Button, CheckBox, DatePicker, Image, Label, ListPicker, ListView, Notifier, PasswordTextBox, Slider, Spinner, TextBox, TimePicker, WebViewer), Layout, Media, and Drawing and Animation.
- Viewer:** A central workspace showing a preview of the app. It includes a status bar at the top with the time 9:48 and icons for Wi-Fi, signal strength, and battery. The main content area shows a "Paint" screen with two buttons: "Take a picture" and "Upload image". Below the buttons is a large white canvas with a small image icon in the center. At the bottom of the viewer is a color palette with buttons for Red (R), Blue (B), Yellow (Y), Green (G), Purple (P), Cyan (C), Black (B), and a plus sign, along with a "Wipe" button. The Android navigation bar is visible at the very bottom.
- Components:** A tree view showing the hierarchy of components on the screen. It starts with "Screen1", which contains "HorizontalArrangement2" (with Button9 and ImagePicker1) and "HorizontalArrangement1" (with Button2, Button3, Button4, Button5, Button6, Button7, Button13, Button11, Button12, Button1, and Camera1). Buttons are represented by small blue icons.
- Properties:** A panel for configuring the selected component (Screen1). It includes fields for "AboutScreen", "AlignHorizontal" (set to Left), "AlignVertical" (set to Top), "AppName" (DigitalDoodle), "BackgroundColor" (White), "BackgroundImage" (None...), "CloseScreenAnimation" (Default), "Icon" (None...), "OpenScreenAnimation" (Default), "ScreenOrientation" (Unspecified), and "Scrollable".

At the top of the interface, there is a green header bar with the app name "DigitalDoodle", a dropdown menu for "Screen1", and buttons for "Add Screen ..." and "Remove Screen". On the right side of the header bar, there are buttons for "Designer" and "Blocks".

Blocks mode

DigitalDoodle

Screen1 | Add Screen ... | Remove Screen

Designer | Blocks

Blocks

- Built-in
 - Control
 - Logic
 - Math
 - Text
 - Lists
 - Colors
 - Variables
 - Procedures
- Screen1
 - HorizontalArrangement
 - Button9
 - ImagePicker1
 - Canvas1
 - HorizontalArrangement
 - Button2
 - Button3
 - Button4
 - Button5

Rename | Delete

Viewer

```
when Button1 .Click
do
  call Canvas1 .Clear
  set Canvas1 . BackgroundImage to 

when Button2 .Click
do
  set Canvas1 . PaintColor to 

when Button3 .Click
do
  set Canvas1 . PaintColor to 

when Button4 .Click
do
  set Canvas1 . PaintColor to 

when Button5 .Click
do
  set Canvas1 . PaintColor to 

when Button11 .Click
do
  set Canvas1 . LineWidth to  $\text{Canvas1 . LineWidth} + 1$ 

when Button12 .Click

when Button9 .Click
do
  call Camera1 .TakePicture

when Button6 .Click
do
  set Canvas1 . PaintColor to 

when Button7 .Click
do
  set Canvas1 . PaintColor to 

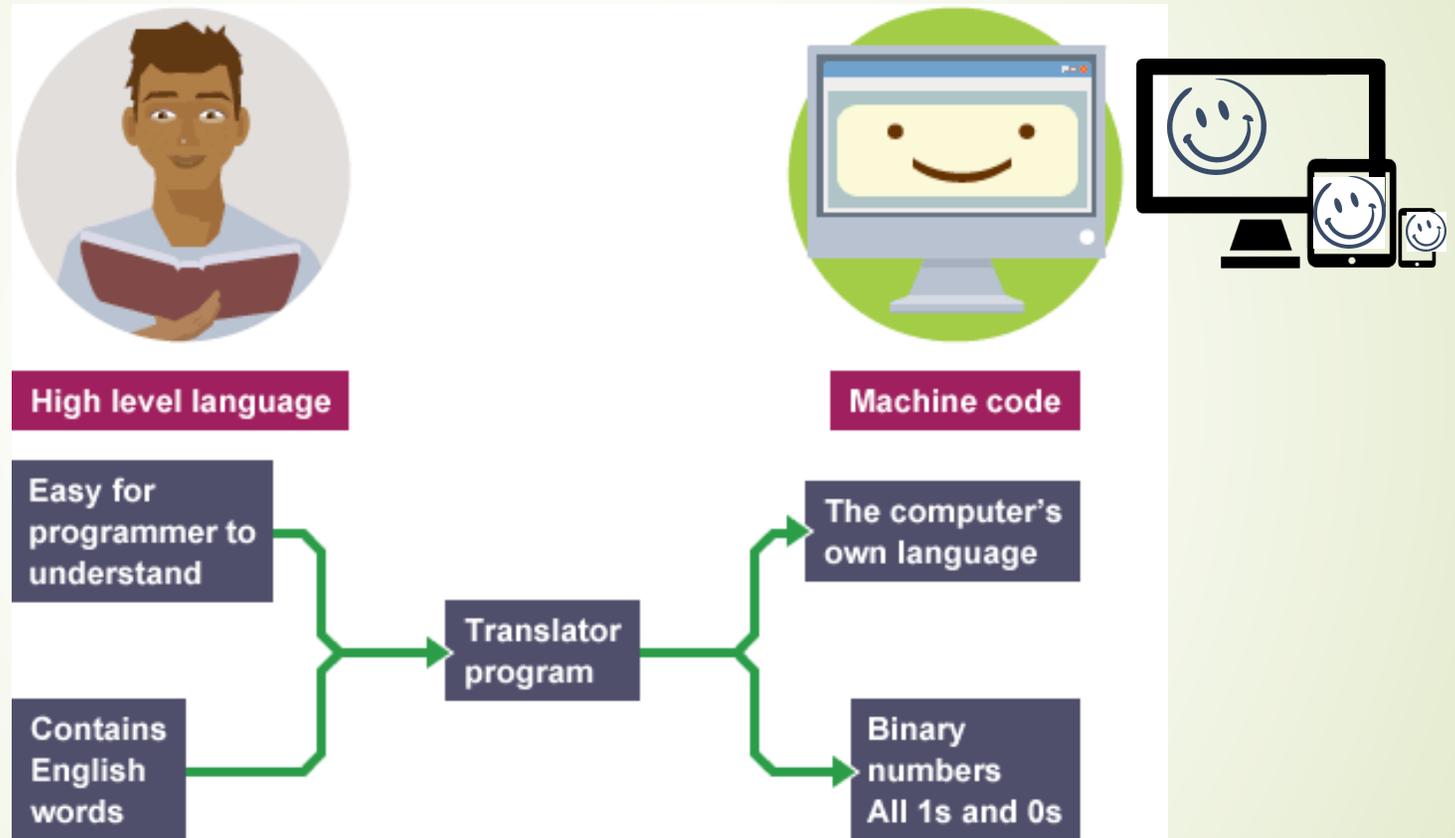
when Screen1 .Initialize
do
  set Button5 . BackgroundColor to 

when Button13 .Click
do
  set Canvas1 . PaintColor to 
```



Blocks mode is like a programming language

14



Basic concepts

➤ Program

- Is a sequence of instructions that comply the rules of a specific programming language, written to perform a specified task with a computer.

➤ Algorithm

- Is a self-contained step-by-step set of operations to be performed to solve a specific problem or a class of problems.

Program vs Algorithm



Algorithm

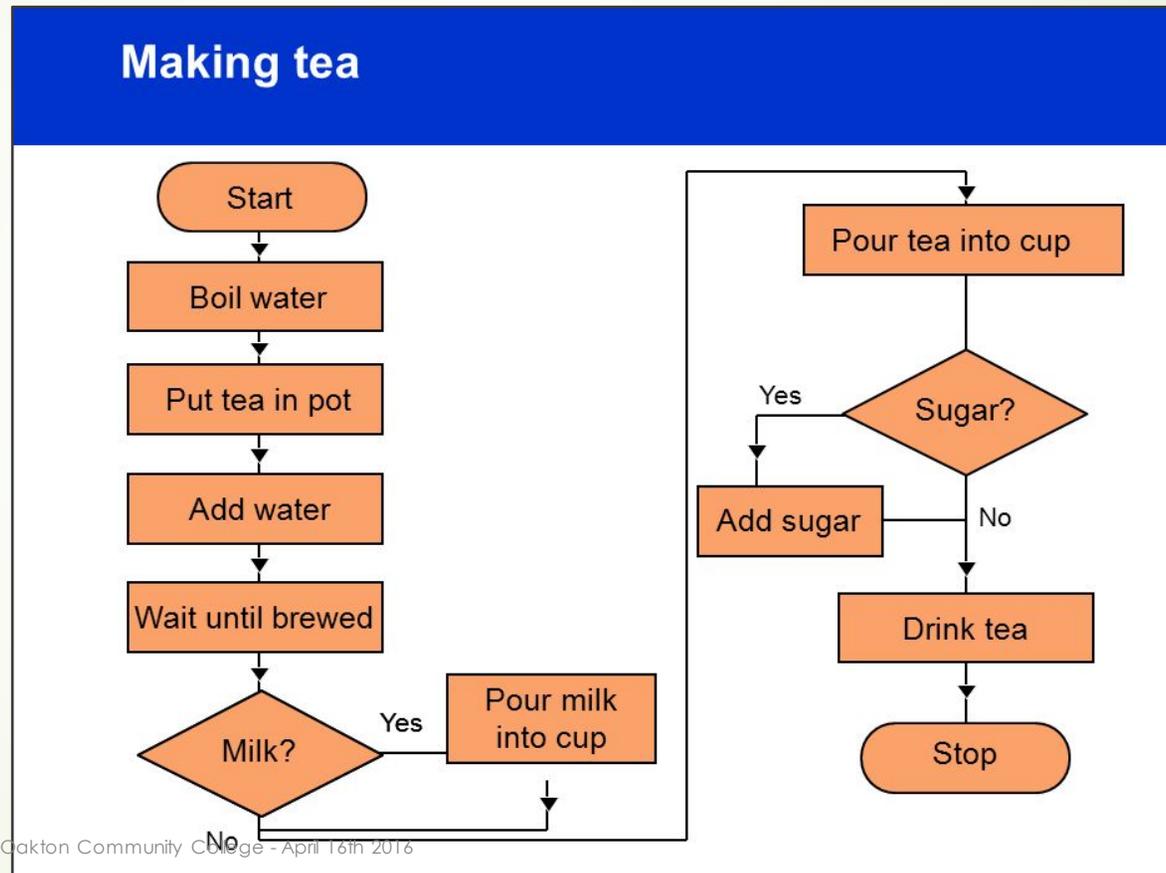
Cup of Tea Algorithm: Set of precise instructions to solve a problem or make something happen.

- 1** Fill kettle with enough water to make a cup of tea.
- 2** Plug kettle in and switch on.
- 3** Place a tea bag in a mug.
- 4** When the kettle has boiled pour water into mug to near the top.
- 5** Leave tea bag until tea is strong enough.
- 6** With a spoon take out the tea bag.
- 7** Add milk to the tea.
- 8** Once the tea is cool enough, drink.

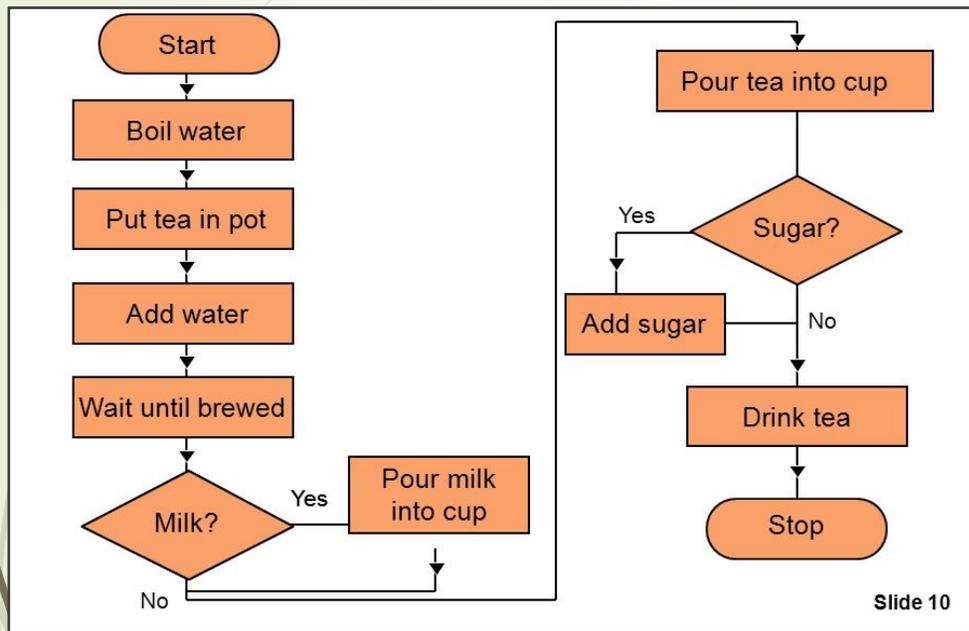
TechSavvy 2016 - Oakton Community College - April 16th 2016

Self-contained
step-by-step set of
operations to be
performed to
solve a specific
problem

Algorithm: flowchart



Algorithm: pseudocode



Start

```

Boil water
Put tea in a pot
Add water
Wait until brewed
If (Milk?)
| Pour milk into cup
EndIf
Pour tea into cup
If (Sugar?)
| Add sugar
EndIf
Drink tea

```

Stop

I have an idea for an app!

- ▶ Algorithms are used to sketch ideas for apps and software
 - ▶ Use your own words
- ▶ Select a programming language
- ▶ Write your app, test it, deploy it and use it!

Let's code!

```
#include <stdio.h>
int main(void)
{
    int count;
    for (count = 1; count <= 500; count++)
        printf("I will not throw paper airplanes in class.");
    return 0;
}
```

WEND 10-3



Talk to me! App

The screenshot displays the MIT App Inventor 2 Beta web interface. At the top, the title bar reads "MIT App Inventor 2 Beta" and includes navigation menus for "Projects", "Connect", "Build", and "Help". On the right side of the title bar, there are links for "My Projects", "Gallery", "Guide", "Report an Issue", "English", and a user profile "jeny.teheran@gmail.com".

The main workspace is titled "Test" and contains several panels:

- Palette:** A list of UI components categorized into "User Interface" (Button, CheckBox, DatePicker, Image, Label, ListPicker, ListView, Notifier, PasswordTextBox, Slider, Spinner, TextBox, TimePicker, WebView), "Layout", "Media", and "Drawing and Animation".
- Viewer:** A central area showing a mobile device preview. It includes checkboxes for "Display hidden components in Viewer" and "Check to see Preview on Tablet size.". The preview shows a screen with a status bar at the top displaying "9:48" and a bottom navigation bar.
- Components:** A panel showing a list of components currently on the screen, including "Screen1". It has "Rename" and "Delete" buttons.
- Properties:** A panel for configuring the selected component, "Screen1". It includes fields for "AboutScreen", "AlignHorizontal" (set to "Left"), "AlignVertical" (set to "Top"), "AppName" (set to "Test"), "BackgroundColor" (set to "White"), "BackgroundImage" (set to "None..."), "CloseScreenAnimation" (set to "Default"), "Icon" (set to "None..."), "OpenScreenAnimation" (set to "Default"), "ScreenOrientation" (set to "Unspecified"), and "Scrollable".

Talk to me! App

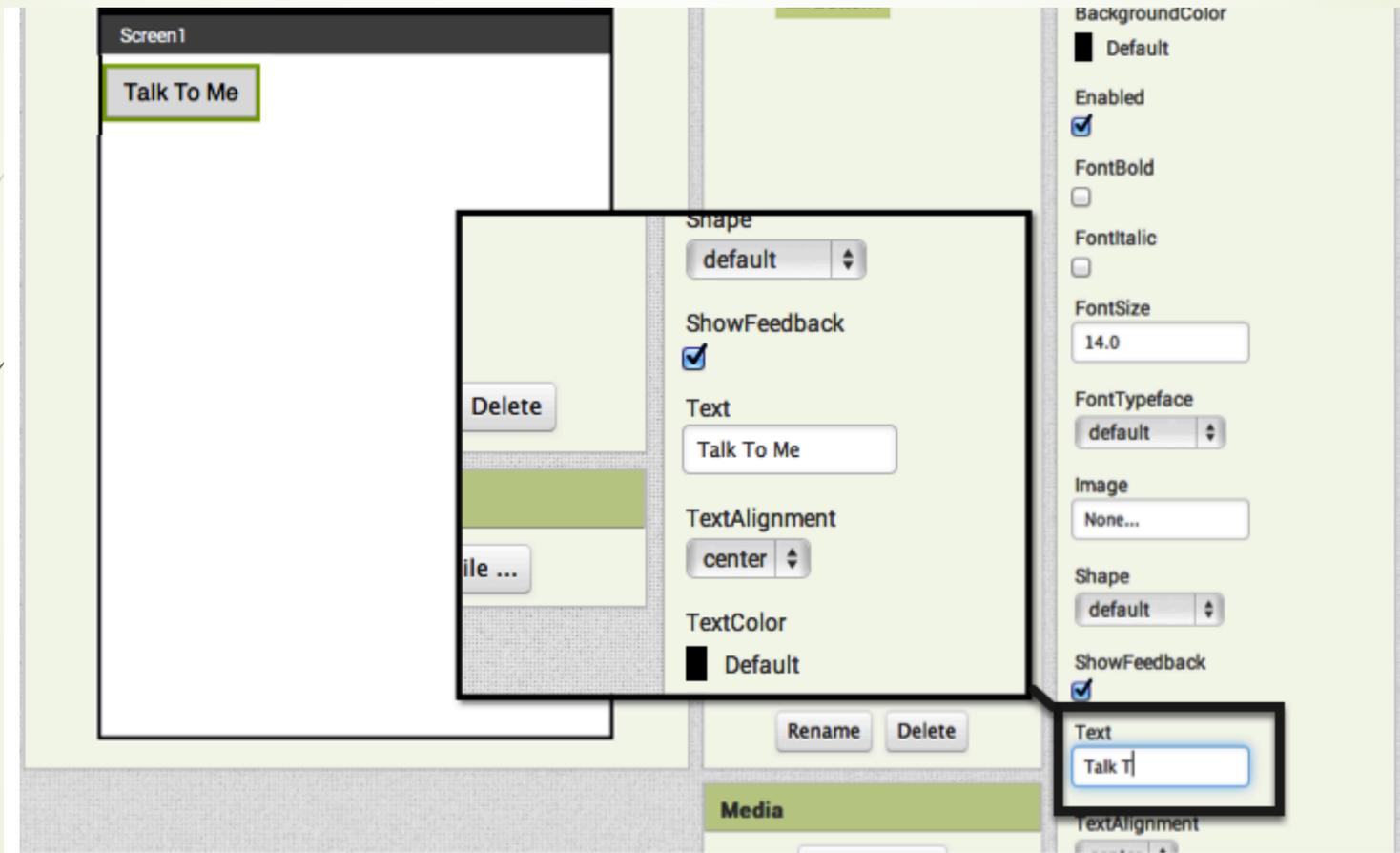
Add a Button

Click and hold on the word "Button" in the Palette. Drag your mouse over to the Viewer. Release the mouse. A new button will appear on the Viewer.

Button

The screenshot displays the Visual Studio IDE interface for the 'TalkToMe' application. At the top, there is a title bar with 'TalkToMe' and three buttons: 'Screen1', 'Add Screen ...', and 'Remove Screen'. Below the title bar are three main panes: 'Palette', 'Viewer', and 'Components'. The 'Palette' pane is titled 'User Interface' and lists several controls: Button, TextBox, ListView, DatePicker, TimePicker, and CheckBox. An orange arrow points from the 'Button' control in the Palette to the 'Viewer' pane. The 'Viewer' pane shows a preview of the application with a button labeled 'Text for Button1'. The 'Components' pane shows a tree view with 'Screen1' containing 'Button1'.

Talk to me! App



Talk to me! App

TextToSpeech

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

- Palette:** The 'Media' category is selected and circled in red. The 'TextToSpeech' component is also circled in red, with an orange arrow pointing from it to the 'Non-visible components' area.
- Viewer:** Shows a mobile app preview with a 'Talk To Me' button. A large grey box with the text 'Drop here. Component will automatically show up in Non-visible components area below' is overlaid on the screen. Below the viewer, the 'Non-visible components' area contains a 'TextToSpeech1' component.
- Components:** Shows a tree view with 'Screen1' containing 'Button1' and 'TextToSpeech1'. Below the tree are 'Rename' and 'Delete' buttons.
- Properties:** Shows properties for 'TextToSpeech1', including 'Country' and 'Language'.

TechSavvy

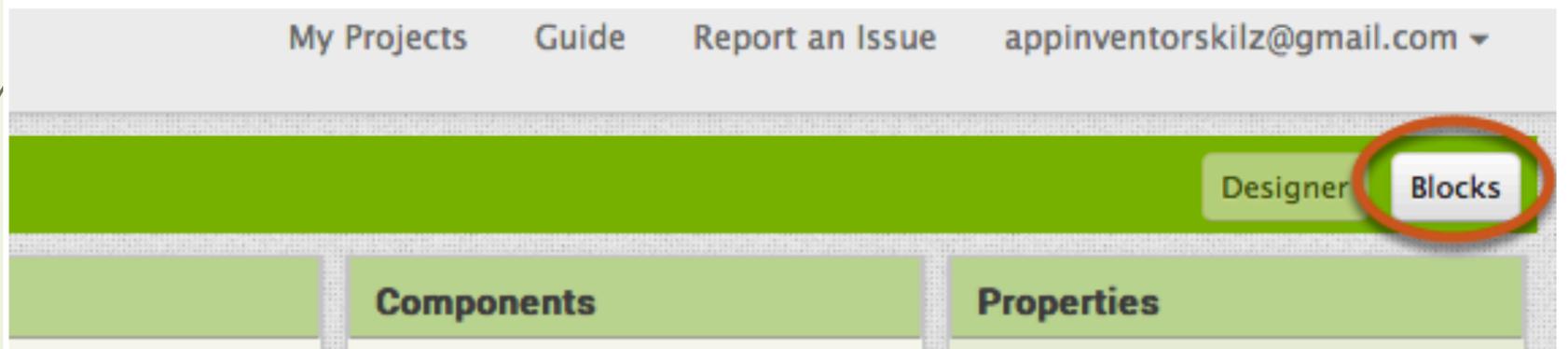
Let's build the app!

Switch over to the Blocks Editor

It's time to tell your app what to do. The Blocks Editor is where you program the behavior of your app.

Click the button "Blocks" to move over to the Blocks Editor.

You will often toggle between the Designer and Blocks Editor as you develop apps.



Blocks mode

MIT App Inventor 2
Beta

Project ▾ Connect ▾ Build ▾ Help ▾

My Projects Guide Report an Issue appinventorskilz@gmail.com ▾

TalkToMe Screen1 ▾ Add Screen ... Remove Screen Designer Blocks

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.

Workspace where you assemble the blocks into a program.

Trash for deleting unneeded blocks.

⚠ 0 ⚠ 0
Show Warnings

Rename Delete

Event Handler

Make a button click event

Click on the Button1 drawer.

Click and hold the ***when Button1.Click do*** event block.

Drag it over to the Viewer and drop it there.

This block will launch when the button on your app is clicked.

It is called an "Event Handler".

TalkToMe Screen1 Add Screen ... Remove Screen

Blocks

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

Viewer

when Button1 .Click
do

when Button1 .GotFocus
do

when Button1 .LongClick
do

when Button1 .LostFocus
do

Button1 . BackgroundColor

set Button1 . BackgroundColor to

Button1 . Enabled

set Button1 . Enabled to

when Button1 .Click
do

TextToSpeech

Program the TextToSpeech action

Click on the TextToSpeech drawer.

Click and hold the ***call TextToSpeech1.Speak*** block.

Drag it over to the Viewer and drop it there.

This is the block that will make the phone speak.

Because it is inside the Button.Click, it will run when the button on your app is clicked.

MIT App Inventor 2 Beta

Project ▾ Connect ▾ Build ▾ Help ▾

My Projects Guide Report an Issue

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 ▾
```

Code

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

1

2

3

Fill in the message socket on TextToSpeech.Speak Block

Now you need to tell the TextToSpeech.Speak block what to say.

Click on the Text drawer, drag out a **text** block and plug it into the socket labeled "message".

The screenshot shows the Scratch IDE interface. The title bar reads "TalkToMe" and includes buttons for "Screen1", "Add Screen ...", and "Remove Screen". The "Blocks" panel on the left is expanded to show "Built-in" categories, with the "Text" category highlighted by an orange circle. The "Viewer" panel on the right shows a script starting with a "when Button1 Click" event, followed by a "do call TextToSpeech1 .Speak" block. The "message" socket on the ".Speak" block is highlighted by an orange circle. An orange arrow points from a "text" block in the "Text" drawer to the "message" socket. Other blocks visible in the "Viewer" panel include "join", "length", "is empty", "compare texts", and "trim".

TextToSpeech

Specify what the app should say when the button is clicked

Click on the text block and type in "Congratulations! You've made your first app."

(Feel free to use any phrase you like.)

```
when Button1 .Click
do call TextToSpeech1 .Speak
  message "Congratulations! You've made your first app."
```

Let's test the first app!

