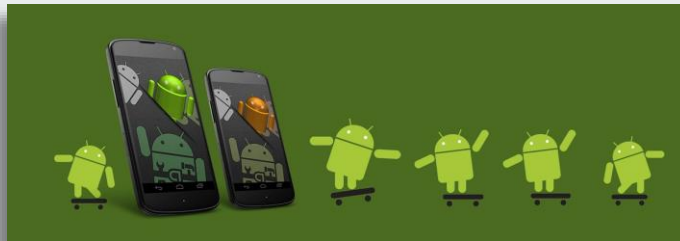




**Code The Future**  
**2019-1-TR01-KA229-074007**  
**Soverato - Italy 9<sup>th</sup>-13<sup>th</sup> December 2019**  
**1st Joint Staff Training**

*Prof. Raffaele Vincenzo Micelotta*

***Apps with App Inventor***

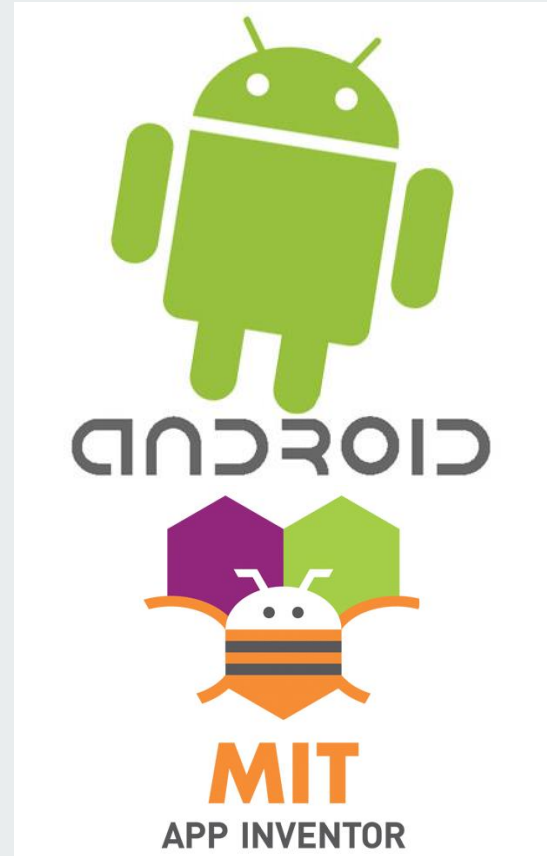


Co-funded by the  
Erasmus+ Programme  
of the European Union



# MIT App Inventor

How to build an Android app easily





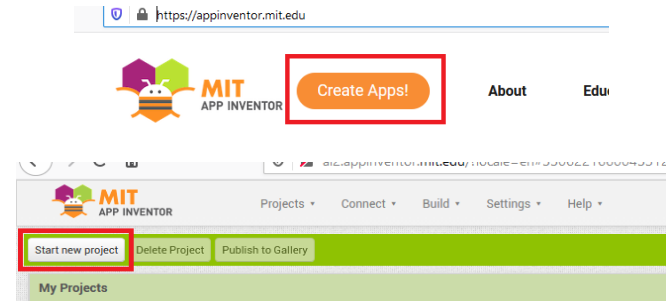
# Introduction to MIT App Inventor



- Originally developed by Google, in 2011 the project came under the management of the MIT (Massachusetts Institute of Technology), taking the name of MIT App Inventor EDU, or more commonly, MIT App Inventor.
- The use of MIT App Inventor lets the creation of an Android app in a much more intuitive process than the traditional methods based on textual languages; in fact, to build an app with App Inventor you use a series of blocks similar to pieces of a puzzle in order to create the code. Starting from December 2013, version 2 of the environment has been released, replacing the previous beta version (renamed App Inventor Classic), introducing a series of improvements and fixing bugs.

# Environment setup and project creation

- MIT App Inventor is a completely online development environment; to use the environment you must have a Google account and connect to the web address <https://appinventor.mit.edu/>
- Click on the “Create apps!” button, select a Google account and password. Once the conditions are met, the home page of the online development environment will be opened.
- Select the "Start New Project" button, located at the top left to create a new project.
- Project files are also kept on a cloud database and are linked to the Google account used for authentication.



# First project

Once the "Start New Project" option is selected, a pop-up will open within which the project name must be entered; after this operation the project will appear in the list of projects ("My Projects") and the screen will open as illustrated below.

The image shows the MIT App Inventor web interface. A red arrow points to the 'Start new project' button in the top navigation bar. A green-bordered dialog box titled 'Create new App Inventor project' is open, with the text 'Project name: FourOperations' and 'OK' and 'Cancel' buttons. Below the dialog, the 'My Projects' section is visible, showing a table with the following data:

	Name	Date Created
<input type="checkbox"/>	FourOperations	Dec 1, 2019, 7:14:41 PM
<input type="checkbox"/>	BricioleDiPane	Nov 26, 2019, 3:02:57 PM

The main interface shows the 'FourOperations' project selected. The central viewer displays a mobile phone screen with a white background. The left sidebar contains a 'User Interface' palette with various components like Button, CheckBox, DatePicker, Image, Label, ListPicker, ListView, Notification, Slider, Spinner, Switch, TextBox, TouchPicker, and WebViewer. The right sidebar shows the 'Properties' panel for the selected 'Screen1' component, with fields for Accessibility, Alignment, AppName, BackgroundColor, BackgroundImage, BlockToolKit, Orientation, Icon, and Opacity.



# Debug and Emulation

In order to test and debug the applications you're going to create, you need a system to connect with a test device, or an emulator. App Inventor provides you with three options to test your creations:

- real-time tests on Android devices with WiFi connection (STRONGLY RECOMMENDED);
- real-time test with Emulator on PC (if you do not have an Android device);
- real-time tests on Android devices with USB connection (if you do not have WiFi connection, use this option as resort).

## Option One - RECOMMENDED

Build apps with an Android device and  
WiFi Connection (preferred):  
[Instructions](#)

If you have a computer, an Android device, and a WiFi connection, this is the

easiest way to test your apps.



## Option Two

Dont have an Android device? Use the  
Emulator: [Instructions](#)

If you dont have an Android phone or tablet handy, you can still use App Inventor. Have a class of 30 students? Have them work primarily on emulators and share a few devices.

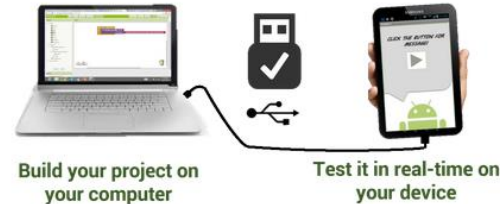


Build your project on  
your computer    Test it in real-time on  
your computer with  
the onscreen  
emulator

## Option Three

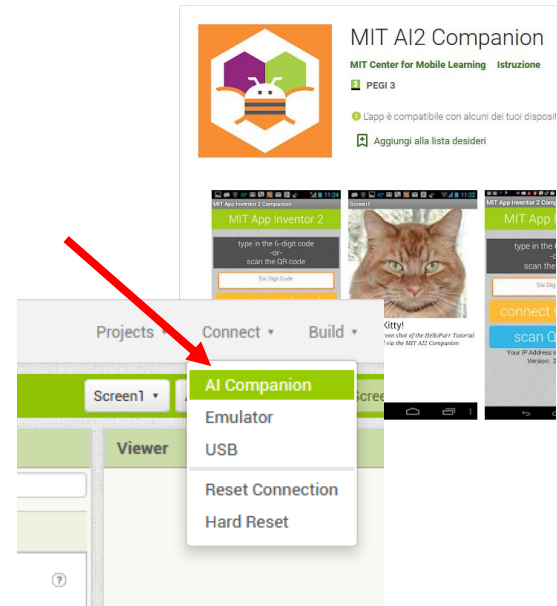
No WiFi? Build apps with an Android  
device and USB Cable: [Instructions](#)

Some firewalls within schools and organizations do not allow the type of  
WiFi connection required. If WiFi doesnt work for you, try USB.



# Real-time testing on Android devices with WiFi

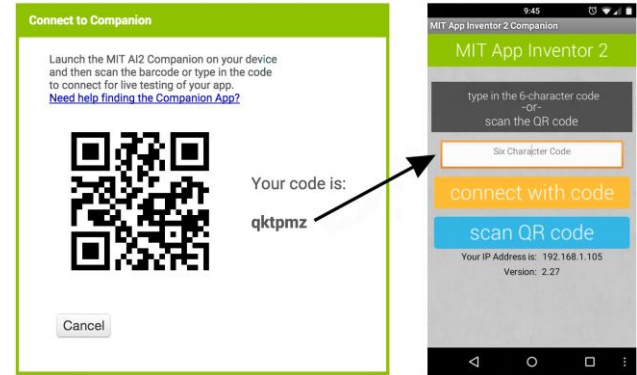
- Install the "MIT AI2 Companion" app on the target device, which can be easily downloaded from the Play Store; once the app has been downloaded, simply follow the various installation steps.
- Connect the development PC and the target device on the same WiFi network.
- Connect the App Inventor project to the device, selecting the "Connect AI Companion" option present on the project bar at the top.





# Real-time testing on Android devices with WiFi

- Once this option has been selected, a dialog box containing a QR code will appear: at this point you can launch the app on the device and connect it to the PC using one of the two available options (scan the QR code or enter the 6-digit code directly).
- After a few seconds the connection will be made and you should see the app you are developing on the screen of your device. The app itself will update each time you make a change from the development environment, thanks to a feature called "live testing".

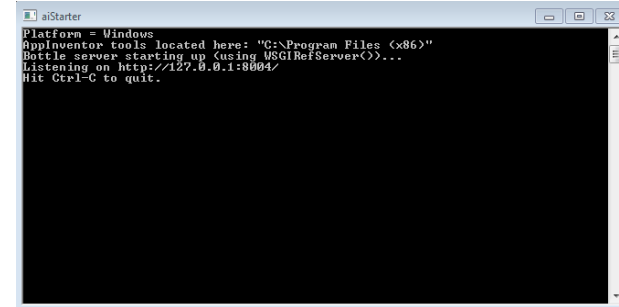


# Real-time test with Emulator

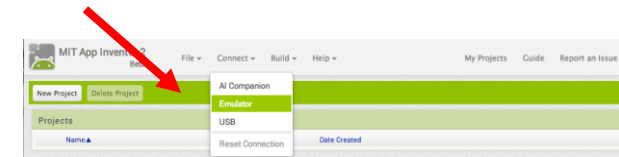
Using the emulator allows you to test your apps without using an external Android device, just using a virtual device that directly runs on the PC.

In order to use the emulator, you need to install it. To do this it is necessary:

1. to download the application (this link is valid for the Windows version) from <https://appinventor.mit.edu/explore/ai2/windows>.
2. after the emulator is installed, it will be launched by double-clicking on the Starter icon that you find on the desktop; a command window will open to display the messages coming from the aStarter on the console;
3. Now it is sufficient to select the "Connect Emulator" option present on the project bar at the top, as shown in the figure.

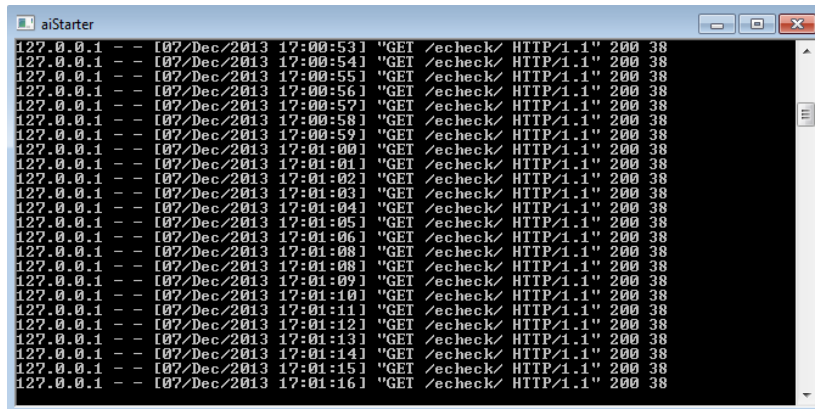


```
aiStarter
Platform = Windows
Appinventor tools located here: "C:\Program Files (x86)..."
Bottle server starting up (using USGIRefServer(<>)...
Listening on http://127.0.0.1:8004/
Hit Ctrl-C to quit.
```

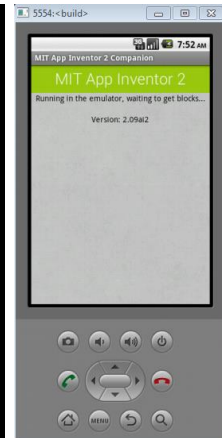


# Real-time test with Emulator

At this point you should notice some activities taking place on the aiStarter side, as shown in the figure on the left, and after a few seconds the device emulator will also run; the app will be eventually synchronized.

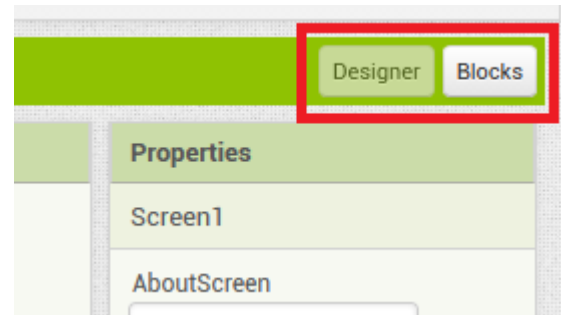


```
aiStarter
127.0.0.1 -- [07/Dec/2013 17:00:53] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:00:54] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:00:55] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:00:56] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:00:57] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:00:58] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:00:59] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:00] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:01] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:02] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:03] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:04] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:05] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:06] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:08] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:08] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:09] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:10] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:11] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:12] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:13] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:14] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:15] "GET /echeck/ HTTP/1.1" 200 38
127.0.0.1 -- [07/Dec/2013 17:01:16] "GET /echeck/ HTTP/1.1" 200 38
```



# Designer view and Block view

Now let's analyze the development environment and in particular let's dwell on the two main screens where we will be working; the App Inventor interface is in fact made up of two main work screens, called Designer View and Block View: in the former the layout of your app is developed, i.e. you build its graphic interface, while in the latter you insert the real code which, as we have earlier seen, is entirely graphic in App Inventor .



# Designer view

In Designer View you find the following main components:

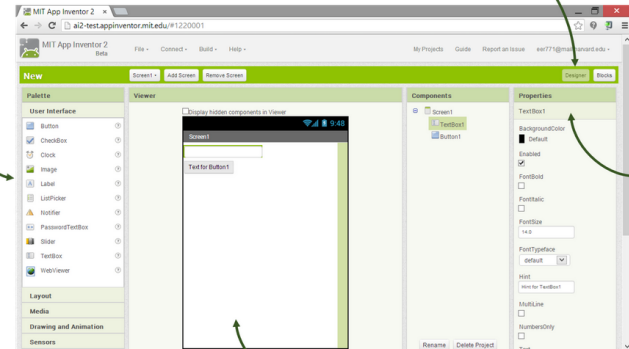
- Main Bar
- Components Palette
- Viewer
- Components View

## App Inventor Designer

Design the App's User Interface by arranging both on- and off-screen components.

**Palette:** Find your components and drag them to the Viewer to add them to your app.

**Designer Button:** Click from any tab to go to the Designer tab.



**Properties:** Select a Component in the Components List to change its properties (color, size, behavior) here.

**Viewer:** Drag components from the Palette to the Viewer to see what your app will look like.

# Designer view - Main Bar

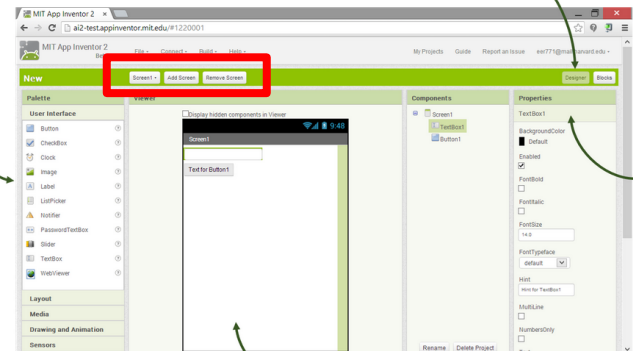
Main Bar is the main project management bar, through which you can manage the screens (adding new views or deleting existing ones). It performs all project management operations, it builds the app and performs many more functions.

## App Inventor Designer

Design the App's User Interface by arranging both on- and off-screen components.

**Palette:** Find your components and drag them to the Viewer to add them to your app.

**Designer Button:** Click from any tab to go to the Designer tab.



**Properties:** Select a Component in the Components List to change its properties (color, size, behavior) here.

**Viewer:** Drag components from the Palette to the Viewer to see what your app will look like.

# Designer view - Component Palette

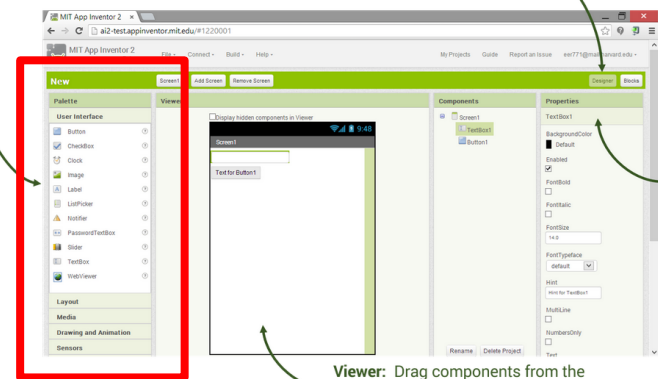
Component Palette is the palette containing the components. It is divided into categories; the components are a very important feature of App Inventor, since in addition to allowing the construction of the graphic interface and the layout, they permit to practically manage all the peripherals present on your Smartphone; they also allow you to perform advanced operations, such as interfacing with social media, storage and connectivity management, etc.

## App Inventor Designer

Design the App's User Interface by arranging both on- and off-screen components.

**Palette:** Find your components and drag them to the Viewer to add them to your app.

**Designer Button:** Click from any tab to go to the Designer tab.



**Properties:** Select a Component in the Components List to change its properties (color, size, behavior) here.

**Viewer:** Drag components from the Palette to the Viewer to see what your app will look like.

# Designer view - Viewer

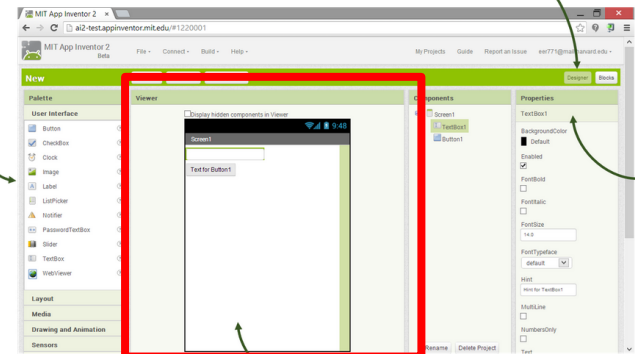
Viewer: this section represents your work area in the design layout phase, in fact it is shaped like the screen of your Smartphone; in it you can drag the components taken from the Palette Components.

## App Inventor Designer

Design the App's User Interface by arranging both on- and off-screen components.

**Palette:** Find your components and drag them to the Viewer to add them to your app.

**Designer Button:** Click from any tab to go to the Designer tab.



**Properties:** Select a Component in the Components List to change its properties (color, size, behavior) here.

**Viewer:** Drag components from the Palette to the Viewer to see what your app will look like.



# Designer view - Component View

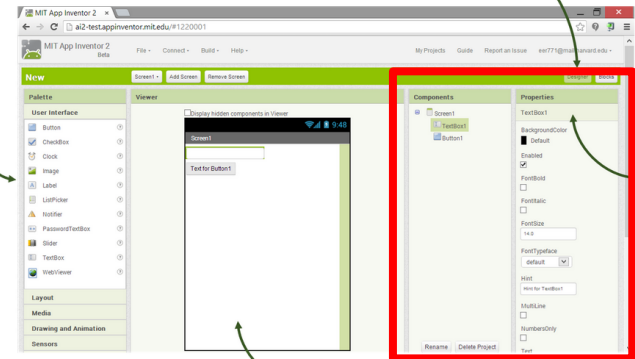
Through Components View you can manage the properties of the components as well as rename or delete them from the layout; this view is divided into two sections called Components (from which you can select one of the components present within a given screen) and Properties, from which you can access the properties of the selected component.

## App Inventor Designer

Design the App's User Interface by arranging both on- and off-screen components.

**Palette:** Find your components and drag them to the Viewer to add them to your app.

**Designer Button:** Click from any tab to go to the Designer tab.



**Properties:** Select a Component in the Components List to change its properties (color, size, behavior) here.

**Viewer:** Drag components from the Palette to the Viewer to see what your app will look like.

# Block View

The Block View is the screen within which the development of the logic of your app takes place.

It is composed of the following elements:

- Main Bar
- Blocks Palette
- Blocks Viewer

## App Inventor Blocks Editor

Program the app's behavior by putting blocks together.

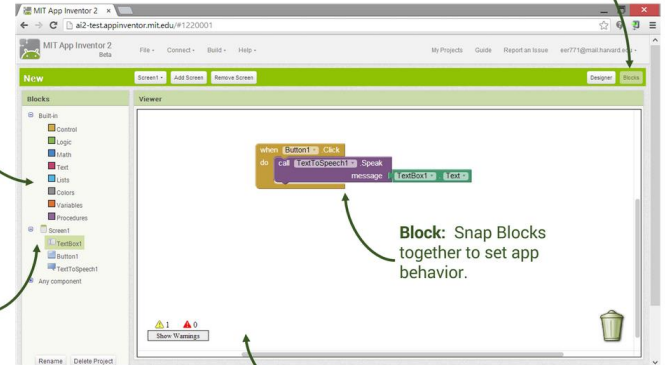
**Built-In Drawers:** Find Blocks for general behaviors you may want to add to your app and drag them to the Blocks Viewer.

**Blocks Button:** Click from any tab to go to the Blocks tab.

**Component-Specific Drawers:** Find Blocks for behaviors for specific Components and drag them to the Blocks Viewer.

**Block:** Snap Blocks together to set app behavior.

**Viewer:** Drag Blocks from the Drawers to the Blocks Viewer to build relationships and behavior.



# Block View - Main Bar

Main Bar is in all respects identical to the Main Bar examined in the case of Designer View.

App Inventor Blocks Editor

Program the app's behavior by putting blocks together.

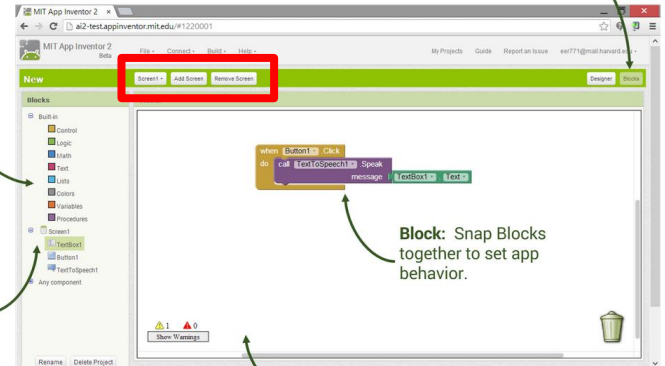
**Built-In Drawers:** Find Blocks for general behaviors you may want to add to your app and drag them to the Blocks Viewer.

**Blocks Button:** Click from any tab to go to the Blocks tab.

**Component-Specific Drawers:** Find Blocks for behaviors for specific Components and drag them to the Blocks Viewer.

**Block:** Snap Blocks together to set app behavior.

**Viewer:** Drag Blocks from the Drawers to the Blocks Viewer to build relationships and behavior.



# Block View - Blocks Palette

Blocks Palette contains the blocks for the implementation of your application; this palette is divided into two parts: the former (called Built-in) contains all the basic blocks such as the logical/mathematical blocks, the blocks for the execution control, the various constants and the various basic operations that can be performed on the blocks, while the latter contains the specific blocks of the various components of the project and this section is occupied as soon as the components are added to the View on the design side.

## App Inventor Blocks Editor

Program the app's behavior by putting blocks together.

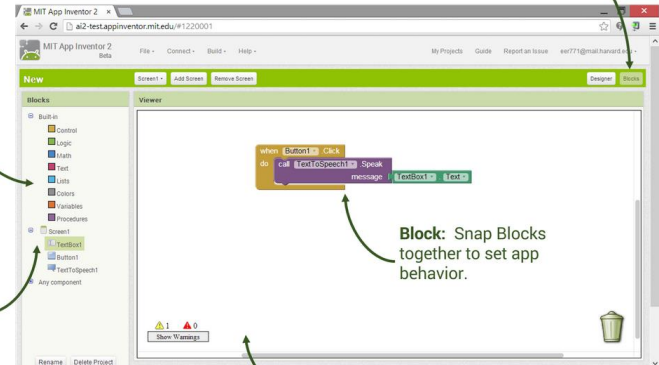
**Built-In Drawers:** Find Blocks for general behaviors you may want to add to your app and drag them to the Blocks Viewer.

**Blocks Button:** Click from any tab to go to the Blocks tab.

**Component-Specific Drawers:** Find Blocks for behaviors for specific Components and drag them to the Blocks Viewer.

**Block:** Snap Blocks together to set app behavior.

**Viewer:** Drag Blocks from the Drawers to the Blocks Viewer to build relationships and behavior.



# Block View - Blocks Viewer

Blocks Viewer is the section in which the graphic code is created, by dragging inside it the various blocks present in the Blocks Palette. To prevent errors in connections, the blocks in App Inventor are shaped like a puzzle and can be fitted together only if they are compatible.

## App Inventor Blocks Editor

Program the app's behavior by putting blocks together.

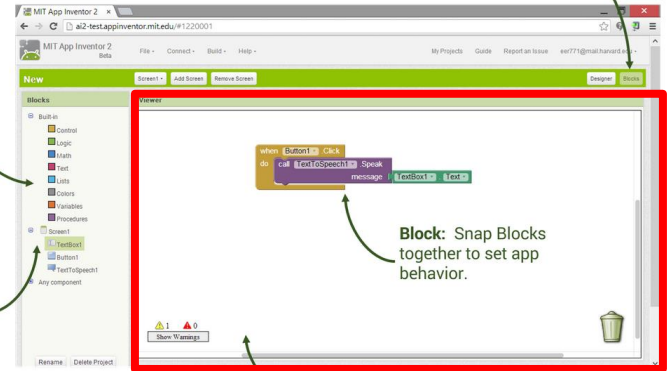
**Built-In Drawers:** Find Blocks for general behaviors you may want to add to your app and drag them to the Blocks Viewer.

**Blocks Button:** Click from any tab to go to the Blocks tab.

**Component-Specific Drawers:** Find Blocks for behaviors for specific Components and drag them to the Blocks Viewer.

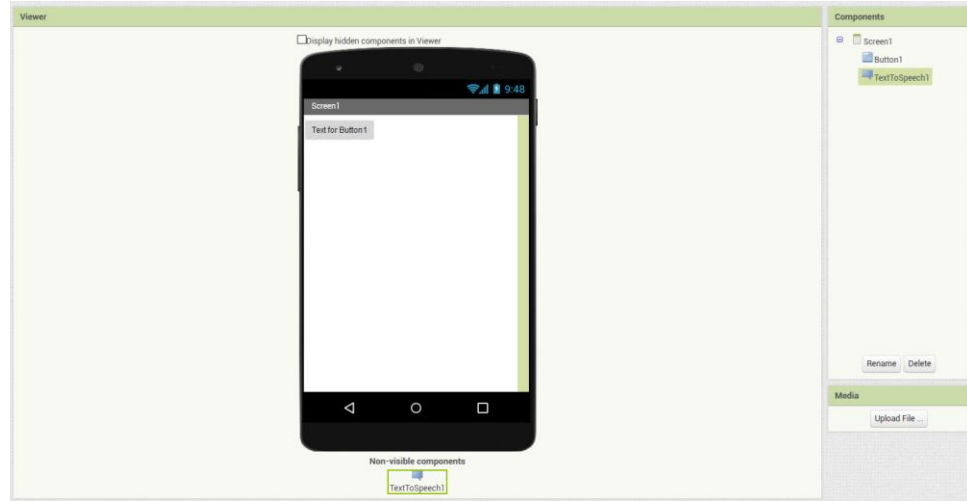
**Block:** Snap Blocks together to set app behavior.

**Viewer:** Drag Blocks from the Drawers to the Blocks Viewer to build relationships and behavior.



# You build your first App

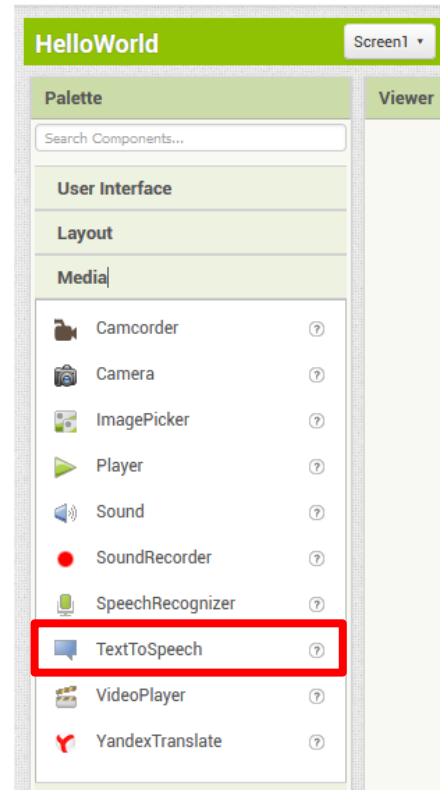
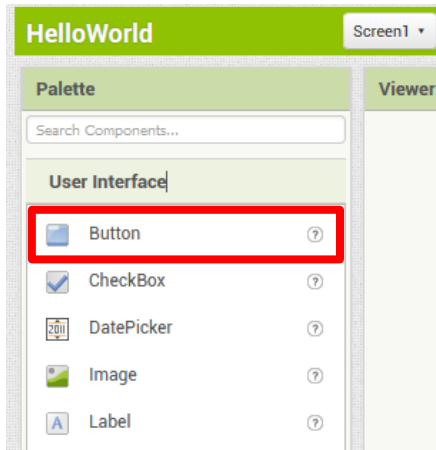
Once got acquainted with the development environment, you can carry out a simple practical example that allows you to immediately understand the power of this programming tool: you will build the famous "Hello World" program, but instead of visualizing the classic string "Hello World" on video, you will use the TextToSPeech component to synthesize vocally your string "Hello World".



# Adding components

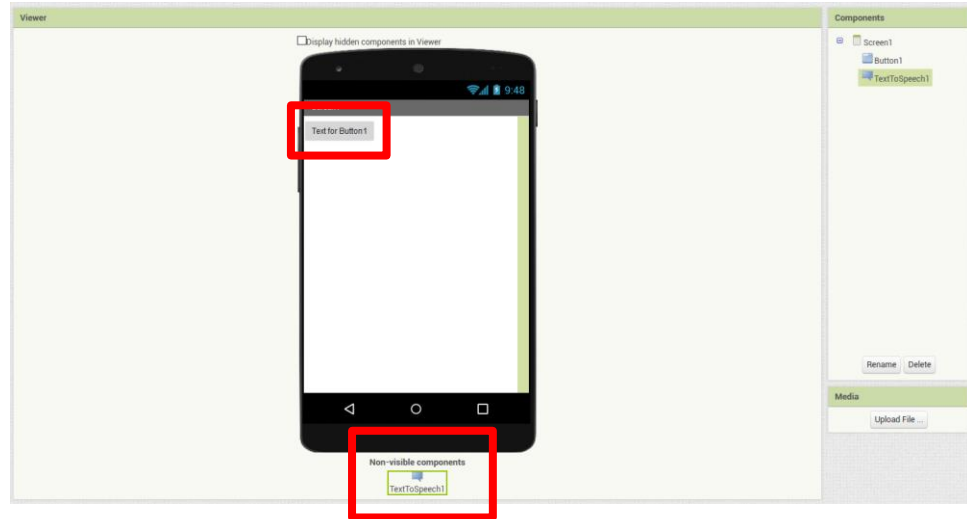
Let's start with the Designer View by inserting the following components into your layout:

- User Interface - Button
- Media - TextToSpeech



# Adding components

- To insert the components on the layout, select them from the palette and then drag them onto the Design Viewer. Once this operation is completed, your layout should be completely similar to the one shown in the figure.
- Unlike the button, the TextToSpeech component is a non-visible component, so its presence within the project is indicated by an icon at the bottom under the layout.



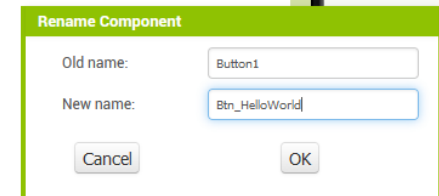
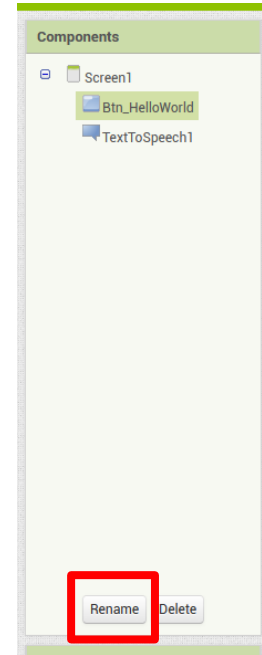


# Rename components

To maintain a minimum order and coding style rename the components as follows:

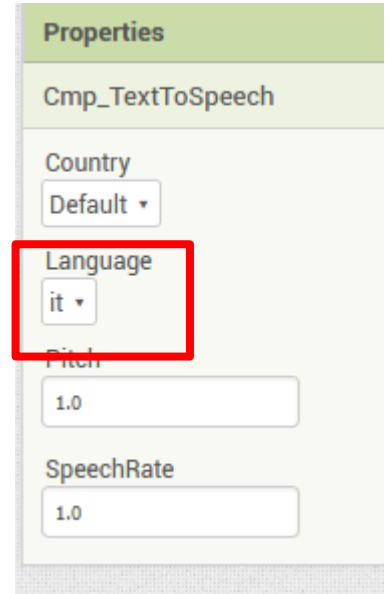
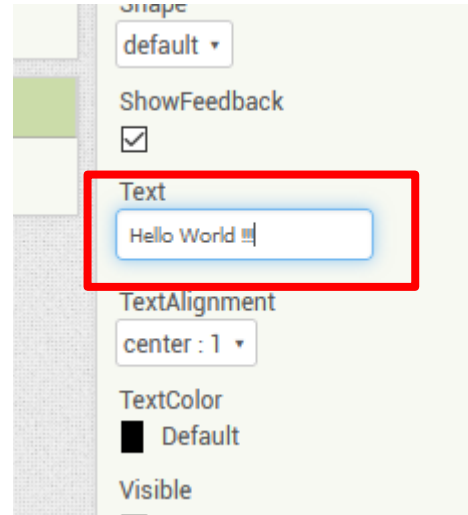
- Button1 to Btn\_HelloWorld;
- TextToSpeech1 to Cmp\_TextToSpeech.

The components can be renamed by selecting them in the component view and clicking on the Rename button, present at the bottom of the same view. A dialog box will appear from which you can rename the component.



# Change properties

Modify the Text property of the Button, changing it to "Hello World !!!", as shown in the figure. Then set the default language for the TextToSpeech component to "It", through the Language drop-down menu accessible from the component properties.

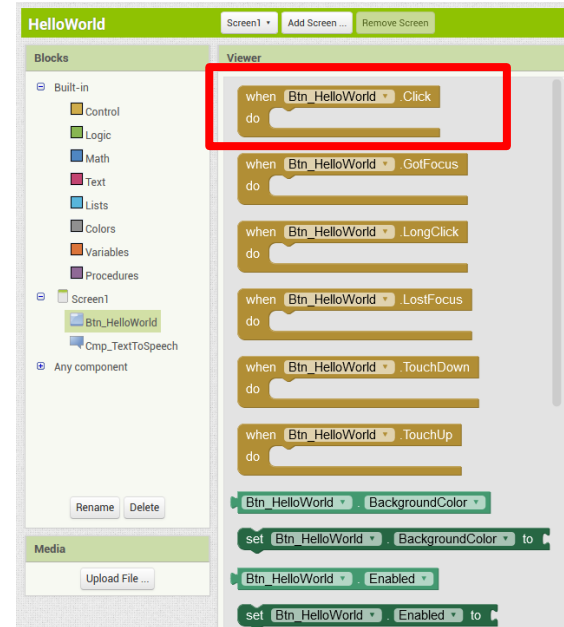


# Add the code

At this point you can switch to Block View to add the graphic code to our app.

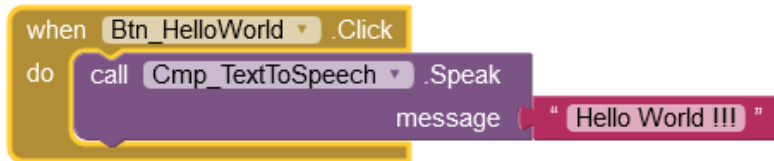
Proceed as follows:

- first place on the Viewer the "when Btn\_HelloWorld.Click...do" block, present between the specific blocks of the Btn\_HelloWorld component.



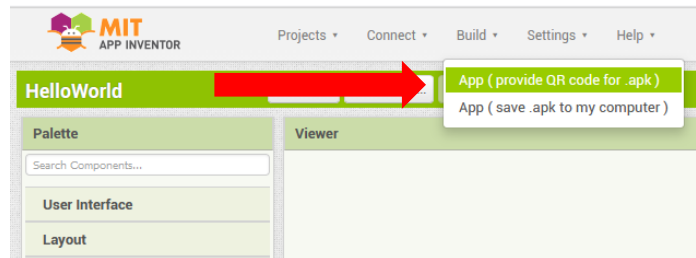
# Add the code

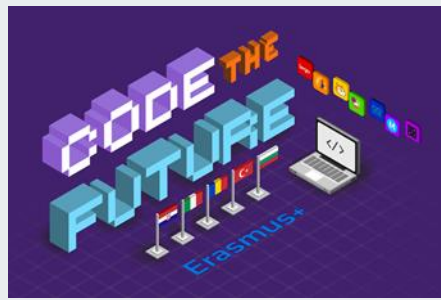
- Then by inserting in the latter the block "call Cmp\_TextToSpeech.Speak" of the component Cmp\_TextToSpeech. Finally you connect a text string block to the latter's "message" connection.
- Finally write the string "Hello World !!!" in the text string block.
- If you did everything correctly the result should be visible as shown.



# Test and build the App

- To test your app created through these steps, you can use one of the two methods described above (device test via WiFi connection or emulator use).
- When you are sure that everything works properly you can also build your app in .apk format and install it on your mobile device. To do this, simply select the BuildApp option (save. Apk on my computer) in the Main Bar, as shown in the figure.
- Once the Build phase is complete, the app will be downloaded to the default browser download folder you are using and you can install it on your mobile device.





**Thanks for watching**

***Prof. Raffaele Vincenzo Micelotta***

**“Giovanni Malafarina” Technic Technological Institute Digital Facilitator**



Co-funded by the  
Erasmus+ Programme  
of the European Union

**it** Giovanni Malafarina  
Istituto Tecnico Tecnologico

"This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein."