Practical 1: Introduction to app inventor

**Introduction**

App Inventor for Android is an open-source web application originally provided by Google, and now maintained by the Massachusetts Institute of Technology (MIT).

It allows newcomers to computer programming to create software applications for the Android operating system (OS).

It uses a graphical interface, very similar to Scratch and the StarLogo TNG user interface, which allows users to drag-and-drop visual objects to create an application that can run on Android devices.

In creating App Inventor, Google drew upon significant prior research in educational computing, as well as work done within Google on online development environments.

App Inventor and the projects on which it is based are informed by constructionist learning theories, which emphasizes that programming can be a vehicle for engaging powerful ideas through active learning.

As such, it is part of an ongoing movement in computers and education that began with the work of Seymour Papert and the MIT Logo Group in the 1960s and has also manifested itself with Mitchel Resnick's work on Lego Mindstorms and StarLogo.

Enables you to develop applications on the android platform, without having to learn complex application development environments and programming languages.

All about priming you before you move onto more complex platform development

**Main components**

Mainly includes the following:

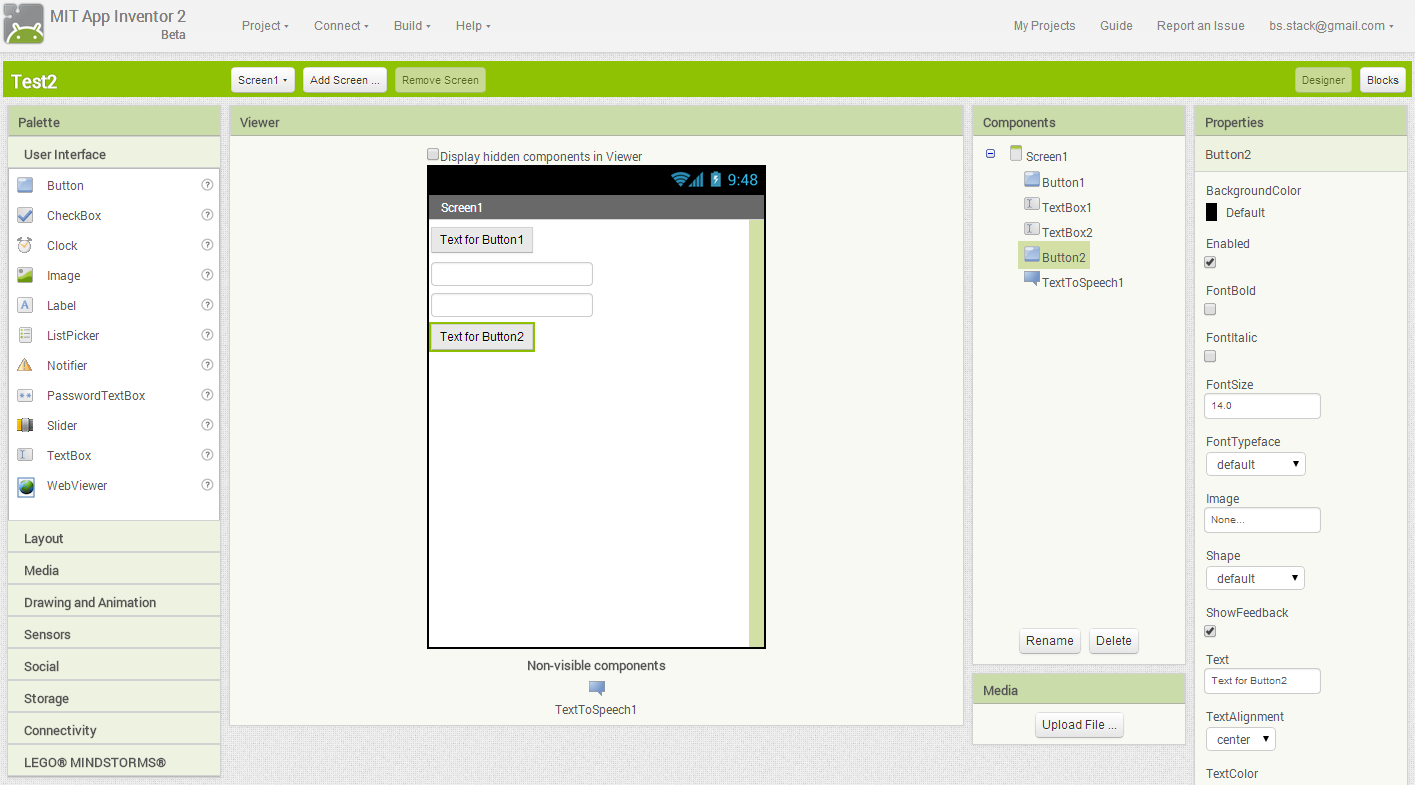
* A designer, in which a program's components are specified. This includes visible components, such as buttons and images, which are placed on a simulated screen, and non-visible components, such as sensors and web connections.
* A blocks editor, in which the program's logic is created.
* A compiler based on the Kawa language framework and Kawa's dialect of the Scheme programming language, developed by Per Bothner and distributed as part of the GNU operating system by the Free Software Foundation.
* An app for real-time debugging on a connected Android device
* Open source

**Primary aims**

App inventors primary aims is to:

* Enable an easy way to design graphical user interfaces
* Enable you to write code in a graphical way

**The graphical user interface and the blocks editor**



**How do you run app inventor apps?**

You can run an app inventor app via:

* AI companion
* Emulator
* USB

**What can you do with app inventor?**

You can do the following:

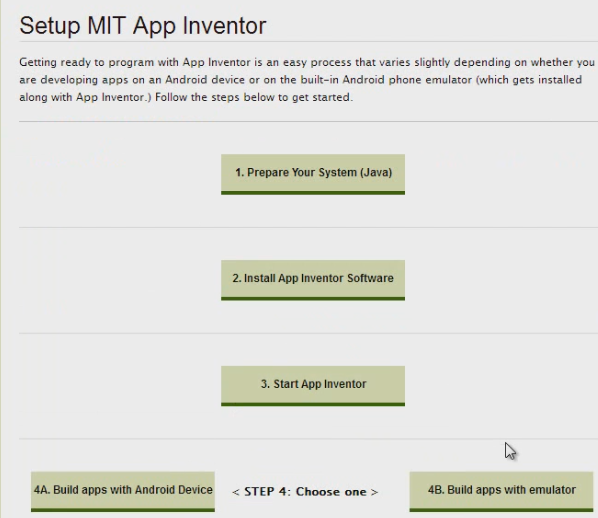
* Build simple Android apps
* Build games
* Access hardware
  + Take pictures
  + Detect movement
  + Get location
* Control Lego MindStorms
* Prototype apps
* More

**What you cant do?**

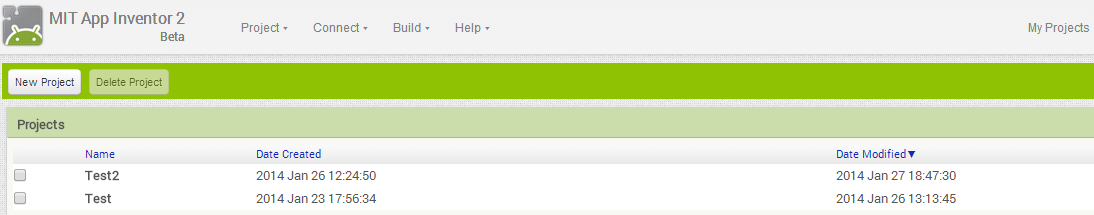
* Create new UI components
* Access all Android APIs

**Installation**

* Java
* App Inventor software
* Start Inventor (Need Google account)
* Connect and Run (Via device or emulator)



**Creating a project**



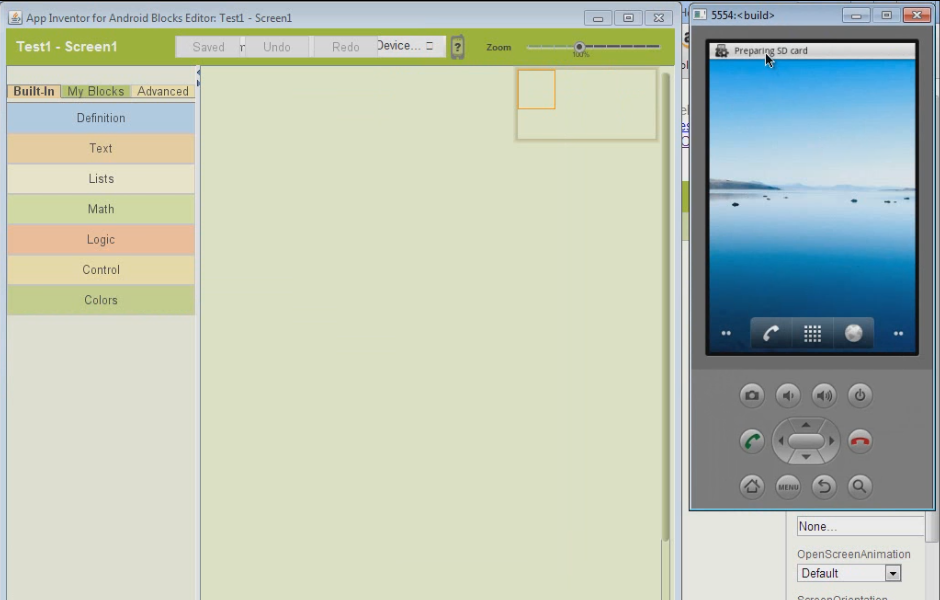
**Demo connecting with WIFI**

Download AI companion app. Scan in QR code when you attempt to connect to the device.

Note sync with device and app inventor

**Demo connecting with Emulator**

Demo using emulator:



**Overview of the designer view**

Describe the following:

* Palette (components and controls)via drag drop
* Components view (Hierarchical View)
* Go through different palette categories
* Describe checkpoint
* Demonstrate the properties section for a component
* Show “Upload Media”
* Go through Project and Build options

**Overview of blocks editor**

Describe the following:

* This is where logic is added
* Logic is represented by blocks that can be connected together
* Note that many blocks exist for a given component
* Note the built in blocks e.g. “color”

**Create simple hello world application**

**Create guess my number game**

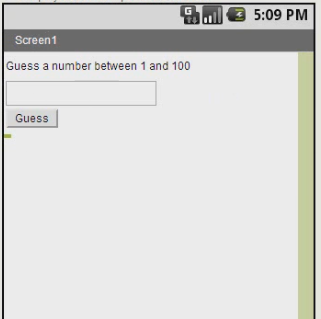
Describe the following:

* Simple game where computer game picks number between 1 and 100
* If guessed correctly, user wins, game starts with new number
* If higher or lower, user is told this and can guess again

Break it down:

* Create the screen
* Pick a random number
* Get the users guess
* Compare guess to value
* Display correct result message
* Reset number when game is over

**Create screen**



**Variables:**

Variables hold a value like a number. We change a variables value by setting it

**Procedures:**

Some pre-packaged logic that does something for us (like methods in Java)

**Parameters:**

Parameters are values that are passed into a procedure

**Define a variable to store a random number between 1 and 100:**



**Test the generation of the random number:**



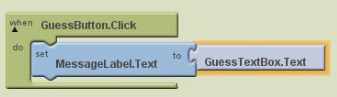
**Properties:**

Properties are variables that belong to a control

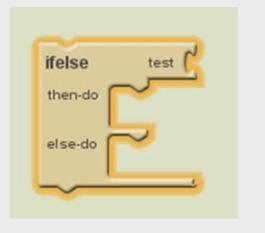
**Events:**

Events let us do something when something happens. When a button is clicked, it fires a click event.

**Getting textbox text**

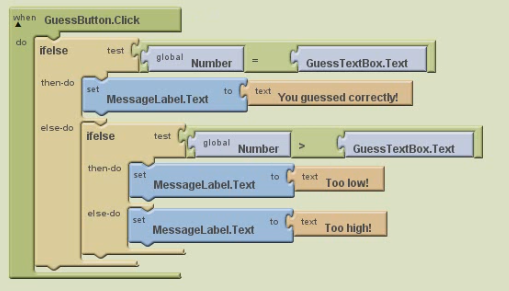


**If-else block**

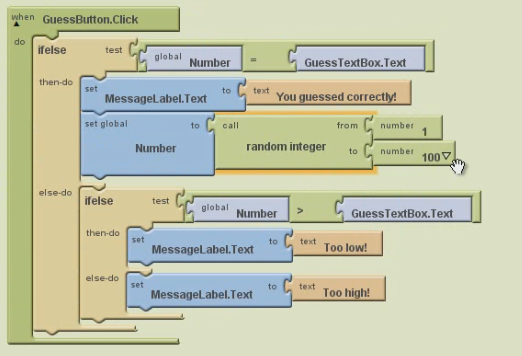


Do the first thing if it is true, do the second if its false.

**Program logic**



**Reset if it is true, try to eradicate duplicate code!**



**Duplicate code removed**

