PROVOLA
APP INVENTOR

di

NiktorTheNat

Realizzato nell'aprile del 2012

INDICE

PREFAZIONE

Questo libro vuole essere di aiuto, per quanti hanno poche capacità con computer e smartphone e voglio utilizzare un semplice linguaggio di programmazione, qual'è App Inventor, per realizzare semplici programmi da usare su cellulari con sistema operativo Android.

Per creare programmi per smartphone Android, bisogna conoscere il linguaggio di programmazione Java, ma Google, ha realizzato un software, che si chiama App Inventor, che permette di realizzare semplici programmi personali, con un linguaggio davvero semplice e visuale, che non necessita studi complessi.

In questo libro verrà spiegato, partendo da zero, come ottenere tutto il software necessario, rigorosamente gratuito, perchè così è distribuito da Google, e come realizzare semplici programmi, imparando a capire la logica della programmazione e le istruzioni che compongono App Inventor.

Le spiegazioni saranno poco tecniche, per non annoiare il lettore meno appassionato di informatica, ma saranno anche utili a chi già sa usare altri linguaggi di programmazione. Inoltre tutto sarà spiegato in modo informale, con qualche intermezzo di libera ispirazione dell'autore, che cercherà di tenere viva l'attenzione di chi legge.

Il titolo stesso del libro "Provola App Inventor", indica come l'autore voglia sdrammatizzare un argomento che potrà sembrare complesso.

L'autore stesso, non è un programmatore professionista né uno scrittore, per questo, l'esplicazione della materia, non sarà paragonabile ad alcun testo tecnico informatico, ma piuttosto ad un dialogo informale, che sicuramente sarà apprezzato dalla maggior parte di voi, considerando che chi legge, potrà trovarsi in difficoltà in determinate spiegazioni o considerazioni più complesse o anche solo quelle iniziali.

UN PO' DI NOTIZIE UTILI

I cellulari Android sono programmabili in linguaggio Java, Se qualcuno vuole realizzare un programma da installare su un cellulare Android, dovrà creare il programma, appunto, in Java.

App Inventor è un linguaggio alternativo, che permette di realizzare i programmi usando un'interfaccia grafica molto intuitiva ed una tecnica di programmazione tipo a puzzle, e che al termine permette di avere un risultato identico a quello che si creerebbe con il linguaggio Java.

La differenza principale tra i due metodi di programmazione, è data dal fatto che mentre con Java è possibile realizzare qualsiasi programma, con App Inventor si hanno istruzioni limitate, che non permetteranno di realizzare le cose complesse realizzabili con Java.

App Inventor è stato creato dai laboratori Google ed è stato reso disponibile gratuitamente sui server Google per molto tempo. Alla fine del 2011, Google ha abbandonato il progetto, rendendolo open source. A questo punto il MIT (Massachusetts Institute if Technology), una prestigiosa università americana, nota anche per lo sviluppo di numerose tecnologie, ha ospitato App Inventor sui suoi server, preoccupandosi di mantenere ed aggiornare questo semplice, ma straordinario linguaggio di programmazione visuale.

Il progetto, attualmente in fase Beta, continua a rimanere gratuito e disponibile a tutti gli utenti che hanno un account Google.

Tutti i programmi realizzati con App Inventor, a differenza di quelli realizzati con Java, non sono attualmente pubblicabili ufficialmente sul Google Play (ex Market Android).

COSA SERVE PER PROGRAMMARE CON APP INVENTOR

- Prima di tutto bisogna avere un personal **computer** desktop o notebook (computer portatile) o netbook. Non è possibile utilizzare gli attuali tablet, per difficoltà pratica di gestione e per l'impossibilità di installare determinato software;

- Un **sistema operativo** Mac OS X 10.5 o 10.6 o successivi, oppure Windows XP, Windows Vista o Windows 7, oppure Linux Ubuntu 8 o successivi oppure Linux Debian 5 o successivi;

- Un **browser** Mozilla Firefox 3.6 o successivo, Google Chrome 4 o successivo; Microsoft Internet Explorer 7 o successivo; Apple Safari 5 o successivo;

- E' necessaria una **connessione Internet**, in quanto App Inventor è un software che funziona online, quindi ha bisogno di una connessione Internet sempre attiva. Inoltre da Internet si potranno scaricare alcuni software necessari per predisporre il computer da usare per programmare;

- E' necessario avere **Java** installato sul computer (nelle prossime pagina trovate le informazioni su dove scaricare Java e come testarlo);

- E' consigliabile avere **un cellulare** con sistema operativo Android, per testare i programmi che si realizzeranno.

  Anche se è vero che si possono testare i programmi anche su un emulatore virtuale, cioè su un cellulare "finto" che viene creato virtualmente sul computer, in realtà questo cellulare virtuale, non può sfruttare tutte le potenzialità ed istruzioni di cui App Inventor è dotato;

- Infine serve un **cavetto USB**, che di solito è fornito insieme al cellulare Android e che serve a collegare il computer al cellulare.

IMPORTANTE: App Inventor è sempre in evoluzione, pertanto eventuali novità e soluzioni ai problemi che potreste riscontrare sulla compatibilità degli hardware e impostazioni necessarie, li trovate a questo indirizzo Internet (pagina in lingua inglese): http://beta.appinventor.mit.edu/learn/setup/index.html#setupComputer

PREPARARE IL COMPUTER

Prima di poter iniziare a programmare con App Inventor è necessario verificare se sul vostro computer sia installato il Java. Per verificarlo, connettete il computer ad Internet, quindi andate su questa pagina: http://www.java.com/en/download/testjava.jsp

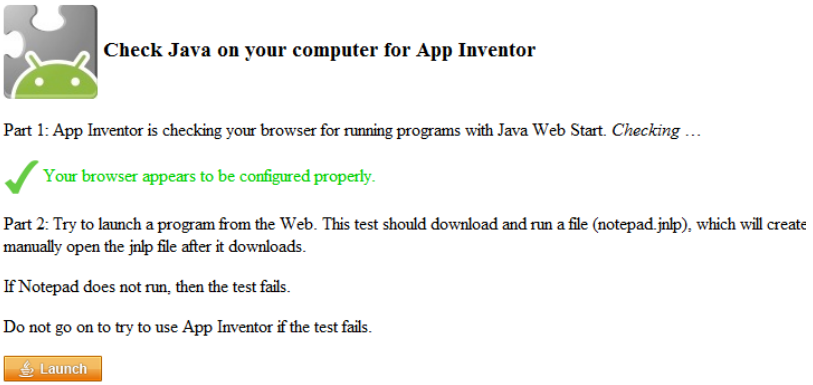
Se la pagina che si apre, ha un messaggio simile:

allora significa che Java è correttamente installato. Se invece non avete alcun messaggio simile, allora dovete installare Java.

Per farlo, andate alla pagina http://www.java.com/it/ e scaricate Java.

Infine, testate che App Inventor possa funzionare con il vostro browser, andando a questo indirizzo: http://beta.appinventor.mit.edu/learn/setup/misc/JWSTest/AppInvJWSTest.html

Apparirà questa pagina:



Se appare in verde la scritta *Your browser appears to be configured properly,* allora Java è davvero installato e configurato per funzionare perfettamente con il vostro computer.

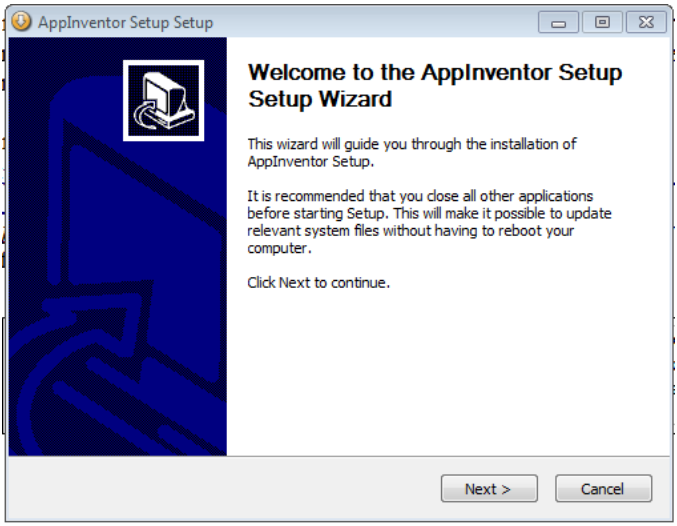
Per poter programmare con App Inventor, dobbiamo preparare il computer con il software necessario per capire le istruzioni che gli daremo.

Per fare questo, bisogna andare sul sito http://dl.google.com/dl/appinventor/installers/windows/appinventor_setup_installer_v_1_2.exe che farà scaricare immediatamente il file *appinventor_setup_installer_v_1_2.exe* (al momento in cui vi scrivo, il software ha una dimensione di 90 mb)

IMPORTANTE: data la continua evoluzione di App Inventor, è possibile che l'indirizzo qui sopra sia variato, pertanto andate all'indirizzo http://beta.appinventor.mit.edu/learn/setup/setupwindows.html per trovare la procedura attuale per scaricare il software necessario (pagina in inglese).

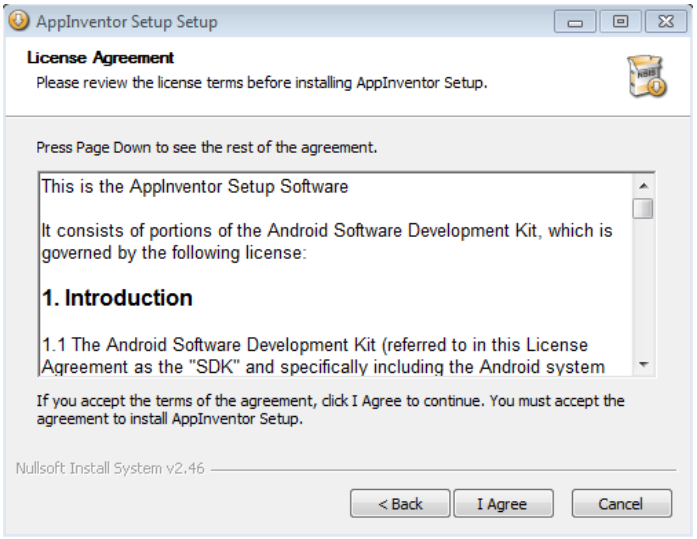
Dopo aver scaricato il file, andate nella cartella dove si trova il file ed avviate la sua installazione.

Avrete una prima finestra che vi da il benvenuto.



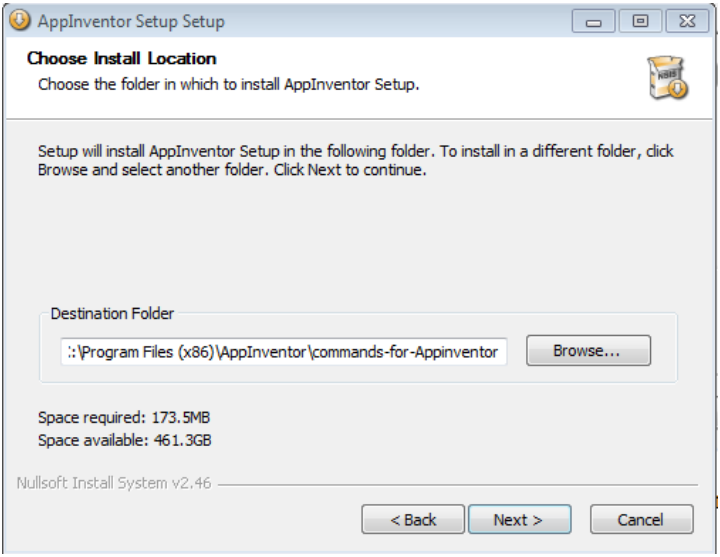
Premete il tasto sinistro del mouse sul pulsante **Next**.

Apparirà la seguente finestra:



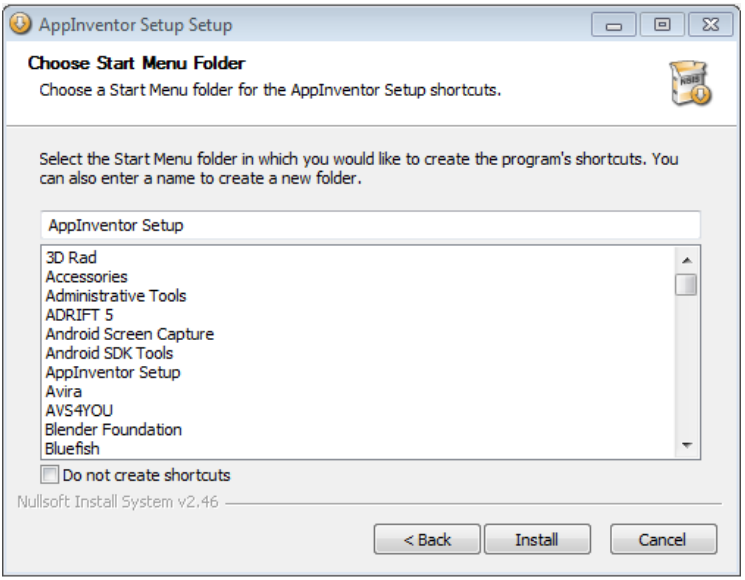
Questa è la licenza d'uso, in inglese. Premete su **I Agree** per accettarla.

Apparirà la seguente finestra che vi informa della cartella dove verranno estratti i file e vi informa dello spazio che avete sul vostro computer (space available) e dello spazio che è richiesto per l'installazione del programma (space required).



Ora premete il tasto sinistro del mouse sul pulsante **Next** per continuare.

Apparirà la seguente finestra che vi indicherà quale nome di file avrete sullo start menu di Windows.



Infine premete su **Install** per iniziare l'installazione vera e propria.

Per i sistemi operativi Linux andate a vedere le istruzioni, in inglese, su questa pagina: http://beta.appinventor.mit.edu/learn/setup/setuplinux.html

Per i sistemi operativi Mac andate a vedere le istruzioni, in inglese, su questa pagina: http://beta.appinventor.mit.edu/learn/setup/setupmac.html

PREPARARE IL CELLULARE

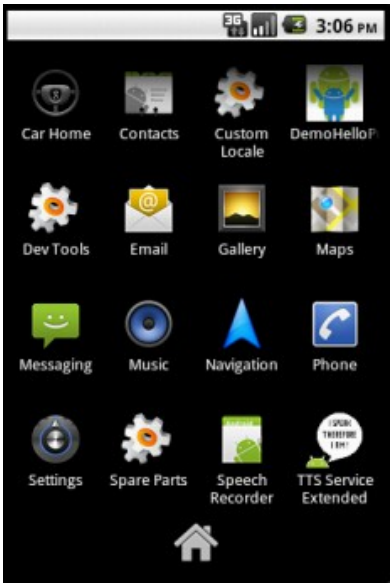
Sul telefono cellulare bisogna eseguire alcune impostazioni. Come sapete, esistono varie versioni di cellulari con Android, alcuni con personalizzazioni dell'azienda produttrice, quindi possono esserci differenze tra quello che verrà indicato in questo testo, rispetto alle impostazioni che vi trovate sul vostro cellulare.

Il fatto che esistano tante versioni, non mi permette di darvi indicazioni specifiche per tutti, quindi mi limiterò ad indicare quelle presenti nell'emultore di un cellulare Android su computer, sperando possiate trovare agevolmente le stesse impostazioni sul vostro.

Inoltre vi ricordo che l pagina, in inglese, che spiega come procedere alla configurazione, si trova a questo indirizzo: http://beta.appinventor.mit.edu/learn/setup/phone.html

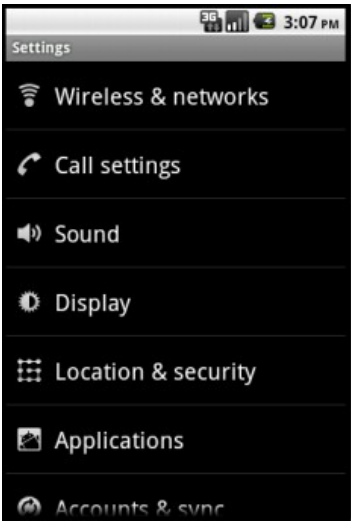
Il vostro cellulare Android ha una schermata principale, chiamata **Home**. Come già detto prima, ognuno può avere una grafica diversa a causa dei vari produttori e a causa anche delle varie versioni Android esistenti fino ad ora. Quella sotto è una di queste:

Premete sul pulsante che vi permette di accedere ai programmi del vostro cellulare. Avrete la visualizzazione delle icone dei vari programmi.



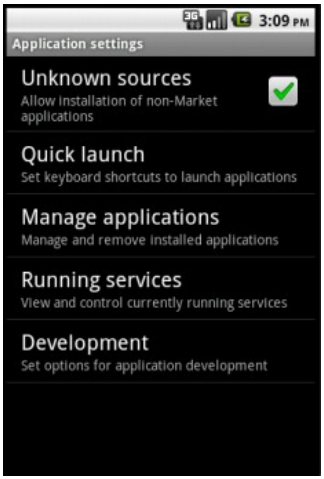
Ora premete sull'icona **Impostazioni**. (Nel caso dell'immagine qui sopra, in inglese, verrà premuto su **Setting**)

Si aprirà un'altra pagina che mostra tutte le impostazioni possibili.



Premete sull'impostazione **Applicazioni** (nell'immagine qui sopra, in inglese, verrà premuto **Application**)

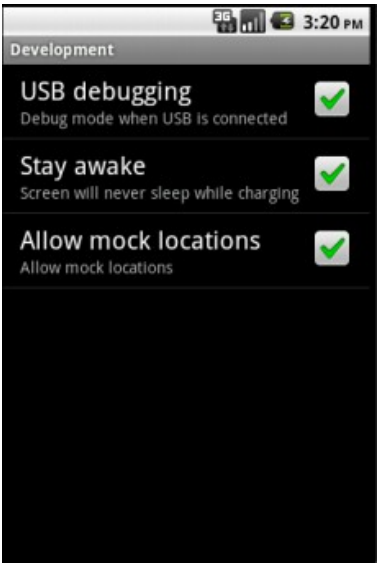
Apparirà la seguente finestra, che come già detto, potrà differire anche di molto, per numero di impostazioni, dalla vostra.



Premete su **Origini sconosciute** (nell'esempio qui sopra, in inglese, si premerà su **Unknow sources**) in modo che il relativo checkbox sia selezionato. In pratica dovete fare in modo che sul quadratino a destra della scritta, ci sia la V verde, o comunque che quel quadratino sia selezionato.

Ora bisogna premere sull'opzione **Sviluppo** (nell'esempio qui sopra, in inglese, si premerà su **Development**).

Apparirà la seguente finestra (o simile).



Qui bisogna selezionare tutte e tre le impostazioni mostrate nella foto sopra, cioè bisogna fare in modo che tutte e tre abbiano le V verdi all'interno dei rispettivi quadratini.

A questo punto il cellulare è pronto per essere utilizzato per essere programmato con App Inventor.

IMPORTANTE: quando programmeremo con App Inventor, ricordatevi che il cellulare va collegato con il cavetto USB al computer e ricordatevi anche di sbloccare il cellulare, cioè di non avere il display spento o bloccato.

Ricordate anche che molti dei programmi possono essere anche testati con l'emulatore, di cui vedremo il funzionamento ed installazione più avanti.

LA PAGINA DEI NOSTRI PROGETTI APP INVENTOR

Ora che abbiamo tutto il necessario per programmare con App Inventor, apriamo il browser che preferiamo (Internet Explorer, Google Chrome, Safari, Opera, Mozilla Firefox, ecc...) e colleghiamoci al sito:

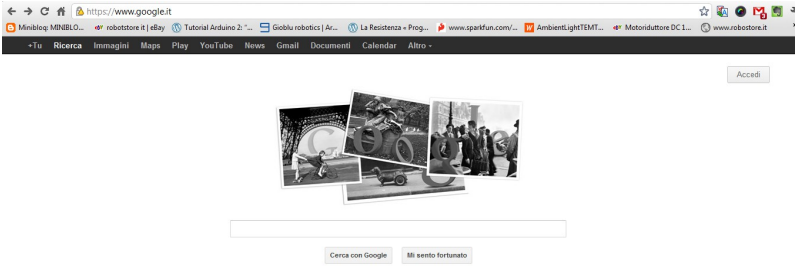
http://beta.appinventor.mit.edu

Se non avete un account Google, vi verrà chiesto di crearlo.

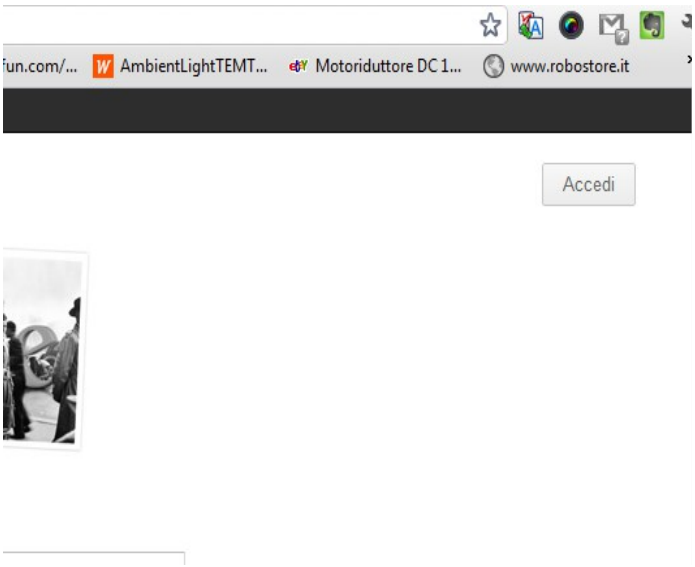
Cos'è un account Google? Google, come sapete, è una società che offre numerosi servizi Internet, di cui moltissimi gratuiti, e che per essere utilizzati necessitano di un account, cioè di una registrazione gratuita.

Se non siete registrati a Google (ndr questo è necessario alla data che sto scrivendo questo testo) dovrete registrarvi andando alla pagina: https://www.google.it/

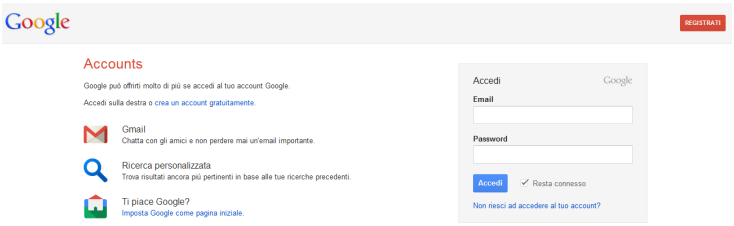
In alto a destra troverete la scritta **Accedi**, come mostrato nell'immagine qui sotto, e ancora meglio nell'ingrandimento più sotto:



(ingrandimento)



Premete su **Accedi**. Si aprirà la seguente pagina:



Dove in alto a destra, troverete un pulsante rosso con scritto **Registrati** o **Crea account**.

Premete su quel pulsante per creare un nuovo account Google.

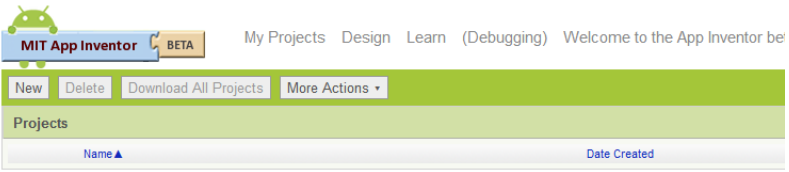
Apparirà così una nuova pagina che vi chiederà i vostri dati personali per procedere alla registrazione gratuita.

Terminata la registrazione, avrete il vostro account Google, con l'username (nome utente) e password che avrete scelto in fase di registrazione.

Quell'username e password vi serviranno per poter usare App Inventor.

Ma torniamo alla pagina di App Inventor che era http://beta.appinventor.mit.edu/ .

Se vi chiede di accedere con l'account di Google, mettete i dati richiesti, quindi si aprirà la pagina dei progetti di App Inventor.



Questa pagina mostrerà l'elenco dei progetti, cioè dei programmi, che avete realizzato fino ad ora.

Chiaramente, trattandosi del nostro primo accesso, questa pagina sarà vuota, senza alcun progetto.

Vi ricordo che tutti i programmi che creeremo con App Inventor non risiederanno sul nostro PC ma saranno tutti salvati online. Questo ci permetterà di aggiustarli, modificarli, cancellarli, ovunque saremo nel mondo, accedendo da un qualsiasi computer, alla pagina di App Inventor, e mettendo l'username e password della nostra iscrizione a Google.

IL NOSTRO PRIMO PROGRAMMA

E' finalmente giunta l'ora di realizzare il nostro primo banalissimo programma con App Inventor, che ci darà modo di vedere come si fa a muoversi tra opzioni e schermate.

La programmazione con App Inventor è davvero semplice, ma allo stesso tempo necessita di capire la logica della programmazione dei computer, perchè App Inventor è comunque un linguaggio di programmazione.

Questo non deve spaventarvi, perchè la logica è alla base del ragionamento di tutti i giorni che fa ognuno di noi.

Se vogliamo aprire una porta, abbassiamo la maniglia e spingiamo la porta. Gesto semplice, ma che nasconde numerosi ragionamenti che ormai abbiamo immagazzinato nel nostro cervello e che il nostro corpo esegue istintivamente.

Se dovessimo creare un programma con App Inventor (ma il ragionamento vale anche per tutti gli altri linguaggi di programmazione) che apra una porta, dovremmo fargli fare tutti i ragionamenti logici che il nostro cervello fa talmente velocemente ed istintivamente che neppure ce ne accorgiamo. Vediamo quali sono:

- verifica se sei abbastanza vicino alla porta per poter afferrare la maniglia
- se la porta è ancora troppo lontana, allora avvicinati
- se sei abbastanza vicino, alza il braccio verso l'alto
- afferra la maniglia
- abbassa la mano verso il basso in modo da abbassare la maniglia

- se la maniglia è totalmente abbassata, allora la porta è sbloccata, quindi spingi la porta

Come avete potuto notare, quelle elencate sopra, sono azioni banali ed ovvie, ma noi le abbiamo memorizzate nel tempo, da quando eravamo infatti. Il computer o il cellulare Android (che non è altro che un computer in miniatura) non sa cosa significa aprire una porta, quindi siamo noi che dobbiamo dirgli tutte le azioni da fare.

Tutte queste azioni insegnate al computer, costituiscono un programma (ndr software).

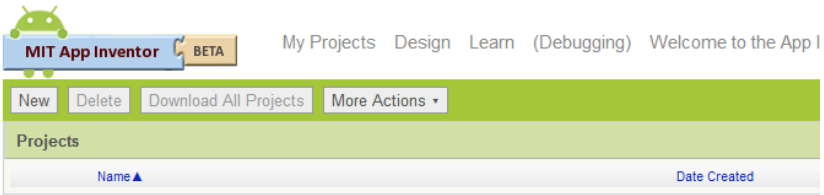
Un programma serve a dare le direttive alla macchina per eseguire determinati ragionamenti.

Se si riesce a scindere queste cose, cioè a ragionare in modo logico e non ovvio, allora si sarà capaci di realizzare programmi. Se non si riesce a ragionare in questo modo, si faranno spesso errori e soprattutto si rimarrà bloccati in malfunzionamenti dei programmi che realizzeremo, senza saper andare avanti. Non sarà solo la logica, l'essenza della programmazione, ma è una parte fondamentale.

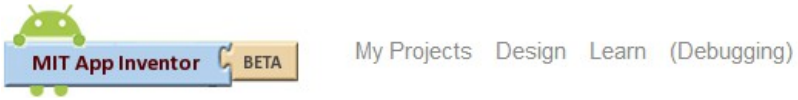
Quindi, tornate bambini, e preparatevi a programmare ;-)

Andiamo alla pagina http://beta.appinventor.mit.edu/ ed eseguiamo l'accesso, se non è già memorizzato.

Si aprirà la seguente pagina con l'elenco dei programmi, che per ora non ci sono:



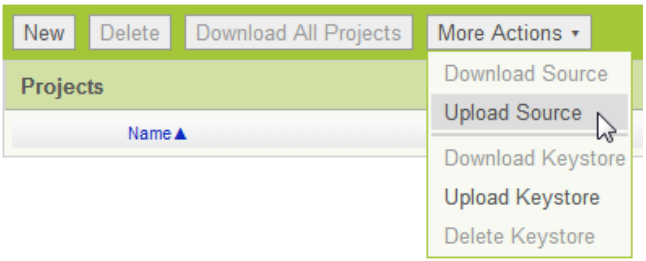
In questa pagina, nella parte alta, c'è questa porzione di istruzioni:



In particolare, **My Projects** permette di tornare a questa pagina, ovvero all'elenco dei programmi che abbiamo realizzato e **Learn** permette di accedere alla pagina, in inglese, con tutti i tutorial ed altre informazioni di apprendimento su App Inventor.

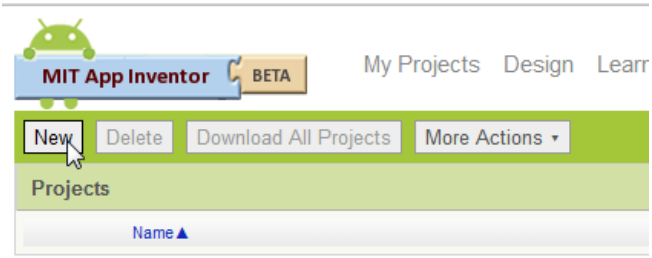
Più in basso a questa porzione che abbiamo appena esaminato, c'è la seguente porzione di istruzioni:

Premendo su **New** si crea un nuovo programma; premendo su **More Actions** si aprono altre opzioni che sono mostrate qui sotto:

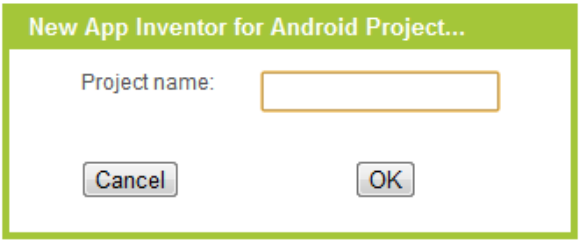


Quella più importante è l'opzione **Upload Source** che permette di caricare il progetto (cioè il programma) che è stato creato da un altro utente. In questo caso, i programmi che si vogliono caricare, sono in formato **zip**.

Ma dopo queste doverose precisazioni e spiegazioni, premiamo sul pulsante **New** per iniziare la creazione del nostro primo programma.



Si aprirà la seguente finestra:

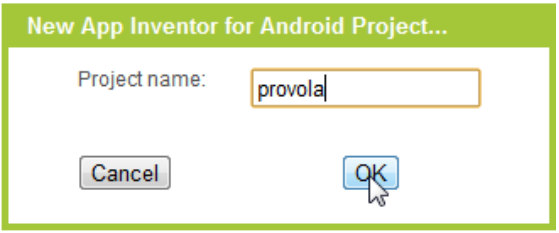


In questa finestra, nella casella di testo **Project name** (cioè Nome del progetto), dobbiamo scrivere il nome che vogliamo dare al nostro programma.

Per questo esempio usiamo il nome **provola**.

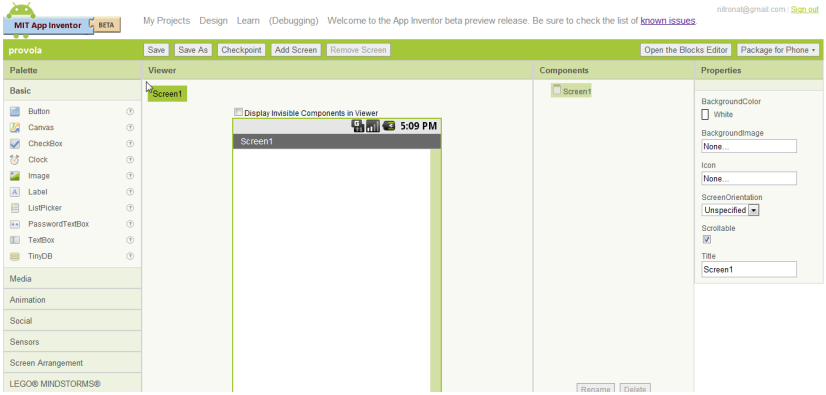
Come vi ho accennato nella prefazione, non essendo né uno scrittore né un tecnico/programmatore, né un insegnante, non seguirò ciò che viene spiegato normalmente nei testi informatici, con la realizzazione del classico "Hello World", ma seguirò una mia personalissima metodologia di spiegazione e nomenclatura.

Quindi scriviamo **provola** dentro la casella di testo, così come mostrato nell'immagine sotto:

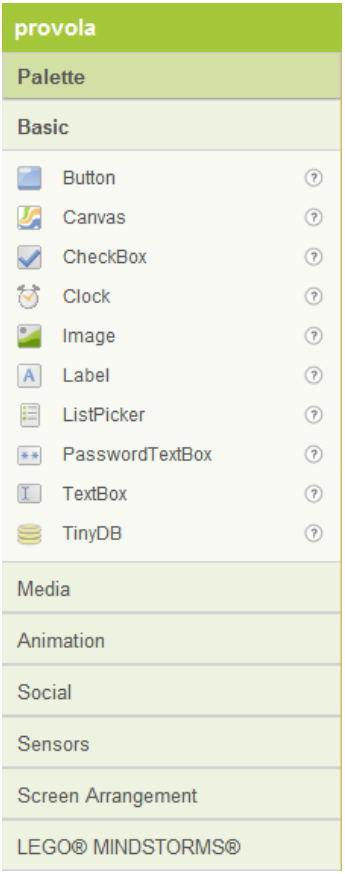


Poi premiamo il pulsante **OK** per confermare.

Dopo pochi istanti, si aprirà la seguente pagina:



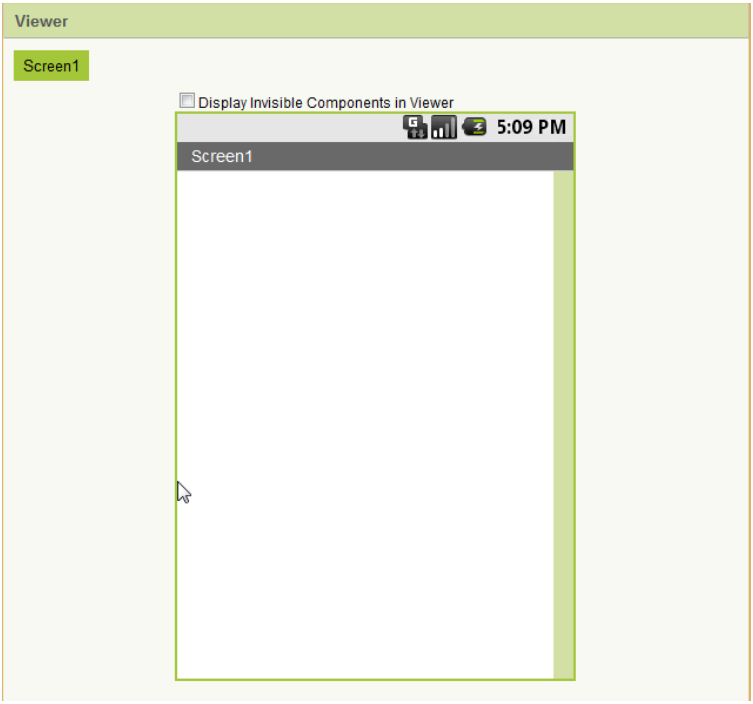
Esaminiamola sommariamente. Nella parte sinistra troviamo la seguente visualizzazione:



La prima cosa che si nota è il nome **provola** che è il nome del nostro programma. Poi troviamo le **Palette**, che si dividono in varie sottosezioni: **Basic, Media, Animation, Social, Sensor, Screen Arrangement, LEGO MINDSTORMS, Other Stuff, Not ready for prime time, Old stuff.**

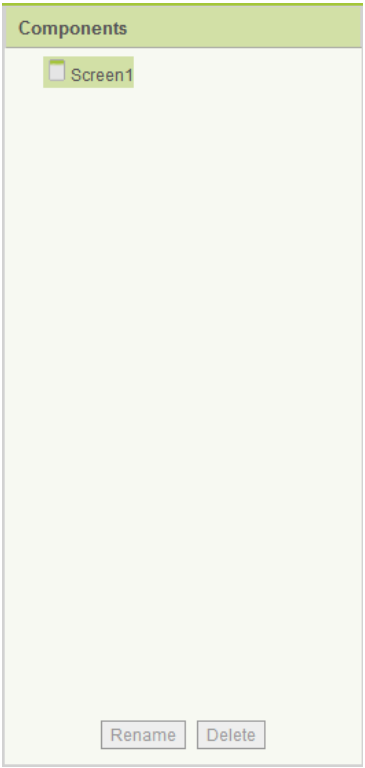
Ogni **Palette** ha i suoi strumenti correlati. Ad esempio, la **Palette** chiamata **Basic**, che si vede nell'immagine precedente, ha i suoi strumenti **Button**, **Canvas**, **CheckBox**, **Clock**, ecc... che possono essere inseriti all'interno del programma.

Nella parte centrale della finestra abbiamo la seguente visualizzazione:



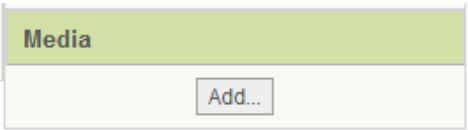
Questa visualizzazione mostra quello che è il display del cellulare. Qui potremo inserire i vari strumenti visti prima e creare così l'interfaccia grafica della nostra applicazione.

Subito a destra di quella visualizzazione, ne troviamo un'altra:



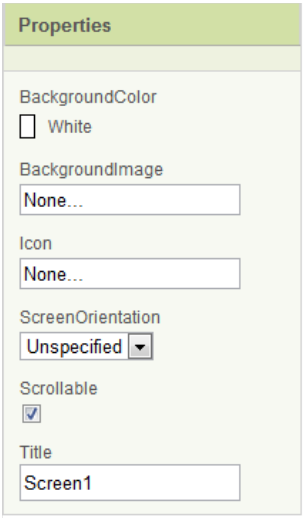
In questa parte avremo l'elenco di tutti gli strumenti che avremo inserito per disegnare l'interfaccia grafica. Potete notare che c'è un solo strumento già inserito; è lo **Screen1**, cioè la prima schermata del programma, che viene creata automaticamente da App Inventor, in quanto è la prima finestra che verrà visualizzata dal programma.

Più in basso c'è anche questa visualizzazione:



In questa parte potremo aggiungere suoni ed immagini che saranno collegati al programma.

Nell'estrema destra dello schermo avremo quest'ultima visualizzazione:

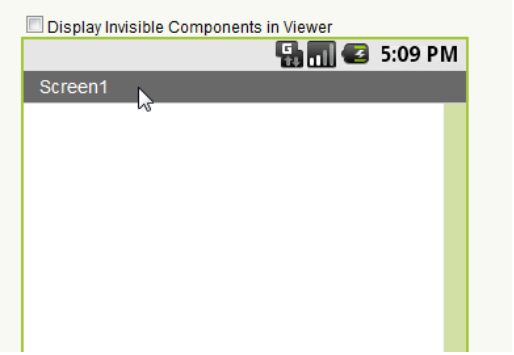


Questa parte ci permette di regolare determinate proprietà di ogni strumento che inseriremo nell'interfaccia grafica. In particolare, quella che vediamo nell'immagine, sono le proprietà dello **Screen1**, che è l'unico strumento attualmente presente nel nostro programma.

Il primo programma che creeremo sarà un pulsante, che premuto, farà apparire un messaggio.

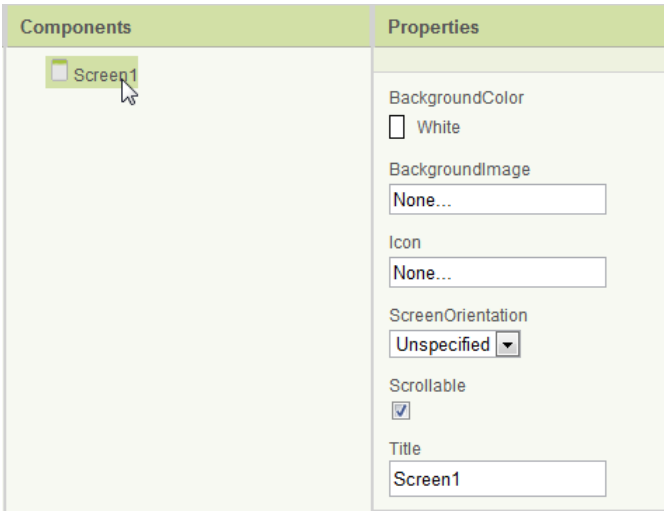
Iniziamo a creare l'interfaccia grafica.

Possiamo notare, nell'anteprima del display del cellulare, che il nome che sarà visibile nella barra alta del programma sarà **Screen1**, come mostrato nell'immagine sotto:

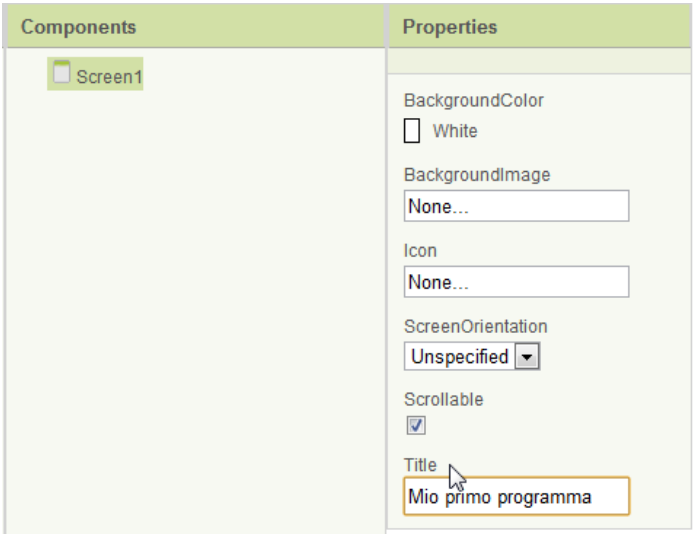


Ora cambieremo questo nome, con il testo *Mio primo programma*

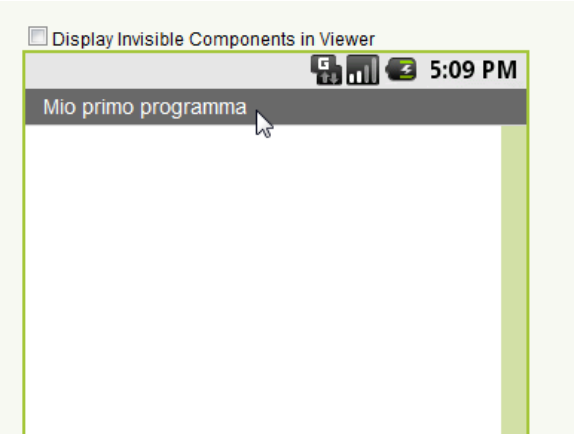
Premiamo il tasto sinistro del mouse sullo strumento **Screen1** nella sezione *Components*:



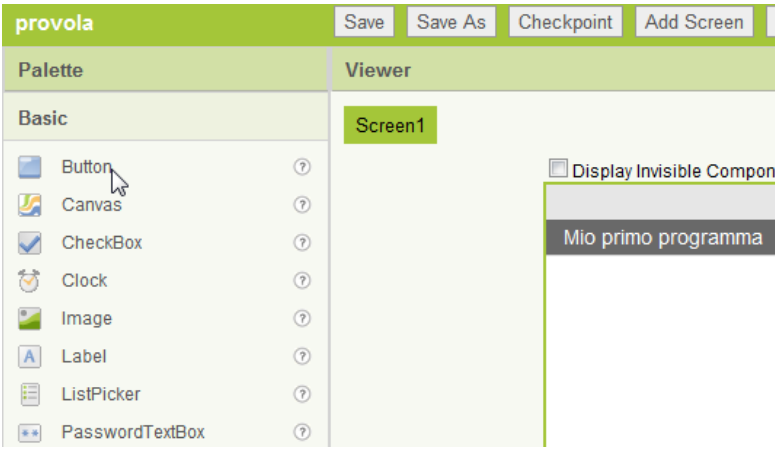
A destra, nella sezione *Properties,* abbiamo le proprietà dello Screen1. Premete il tasto sinistro del mouse su **Title** e cambiate la parola *Screen1* con la frase *Mio primo programma*:



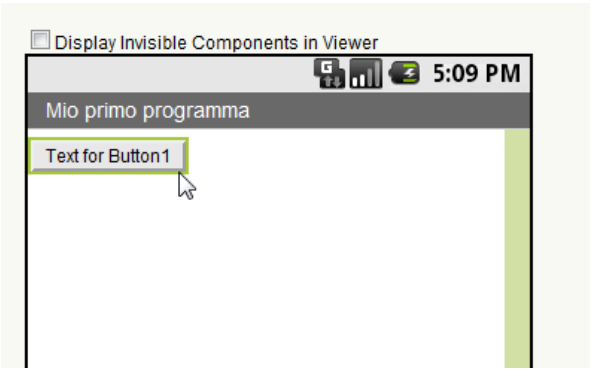
Ecco che anche nell'interfaccia grafica del display del cellulare, troveremo cambiato il nome visualizzato:



Ora inseriamo un pulsante nel display. Per fare questo, premete il tasto sinistro del mouse sullo strumento **Button** e...



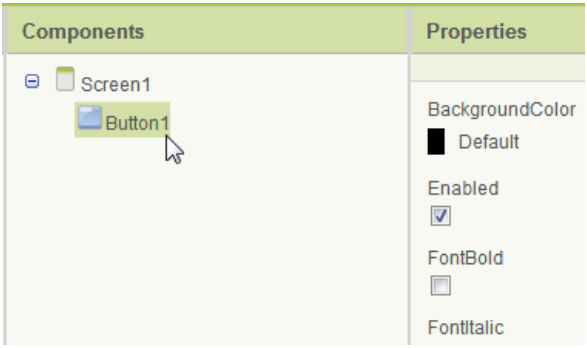
...sempre tenendo premuto il tasto sinistro del mouse, spostate il **Button** dentro l'interfaccia grafica del display:



Il pulsante è stato inserito correttamente nell'interfaccia grafica.

Esaminiamo alcune cose importanti. Sul pulsante c'è la scritta *Text for Button1* che cambieremo tra poco.

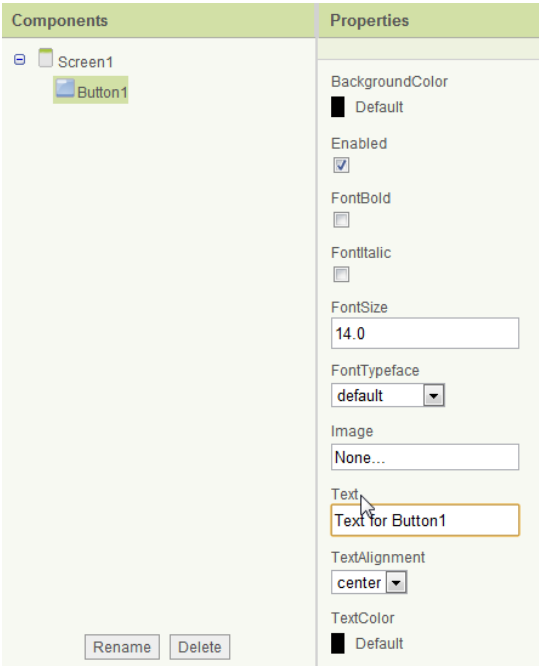
Questo pulsante, ha un nome con il quale verrà identificato all'interno del programma ed il suo nome è **Button1,** come possiamo notare nella sezione *Components*:



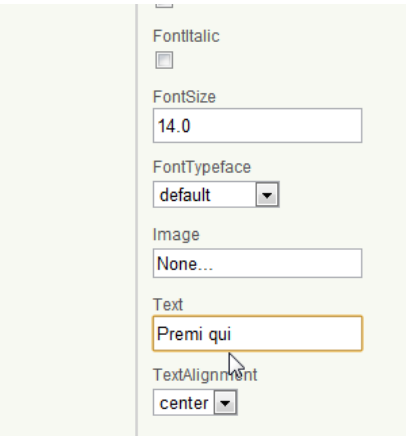
Possiamo anche notare che il **Button1** è, per così dire, figlio dello **Screen1**.

Premiamo su questo **Button1** in modo da avere, nella sezione *Properties*, le proprietà del pulsante.

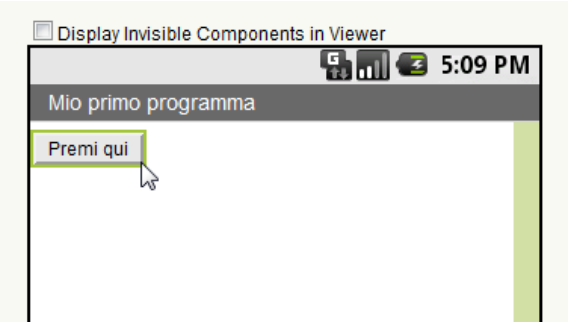
Tra le proprietà del pulsante, selezionate la proprietà **Text** che permette di cambiare il testo all'interno del pulsante:



Lì scriviamo *Premi qui*.

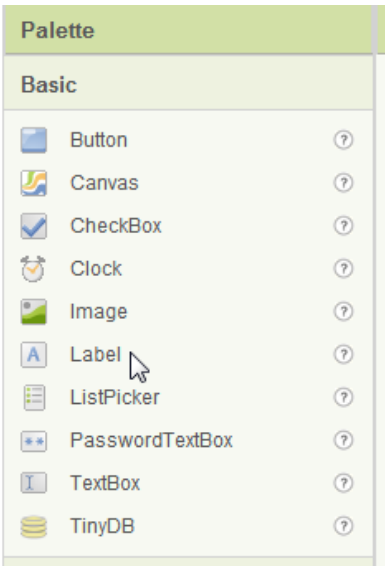


Ecco che anche nell'interfaccia grafica, il pulsante avrà cambiato il suo testo:



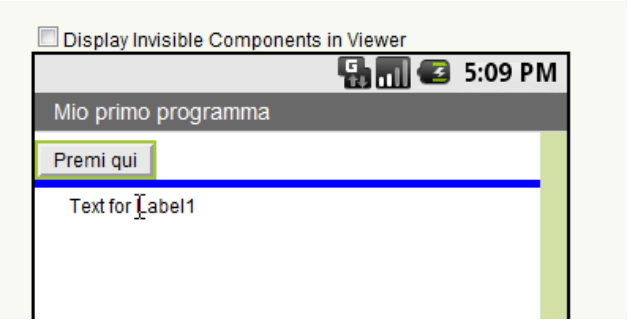
Ora aggiungiamo un'etichetta sotto il pulsante. Sarà lì che faremo mostrare il messaggio. Per aggiungere un'etichetta, osserviamo la sezione *Palette*.

Tra gli strumenti del **Basic** troviamo **Label**, come mostrato in figura.



Questa è l'etichetta. Ora inseriamola nell'interfaccia del programma come abbiamo già fatto per il pulsante, ovvero premendo il tasto sinistro del mouse su **Label** e spostandolo nell'interfaccia.

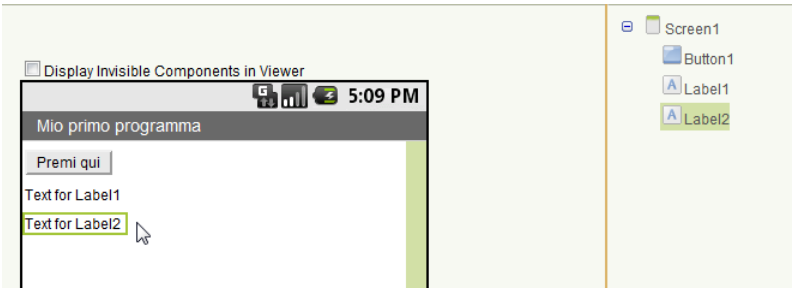
Ora fate attenzione a quando spostate un nuovo strumento, come in questo caso, dentro l'interfaccia grafica. Se appare una linea blu sotto il pulsante che avevamo creato prima, allora l'etichetta verrà inserita sotto il pulsante, come mostrato nella figura sotto:



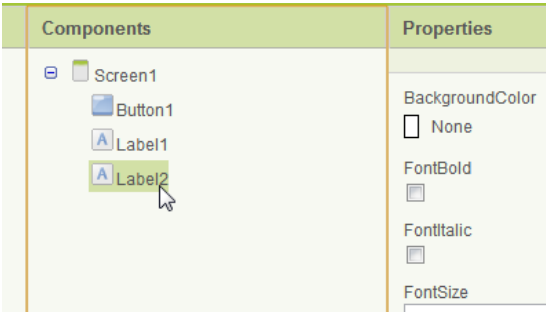
Se invece, quando spostate la **Label** nell'interfaccia, la linea appare sopra il pulsante, allora l'etichetta verrà posizionata sopra il pulsante, così come mostrato nella figura sotto:

Scegliete voi dove preferite inserirla, infondo non avrà alcuna differenza per l'esecuzione del nostro programma. Sarà solo una questione di aspetto grafico.

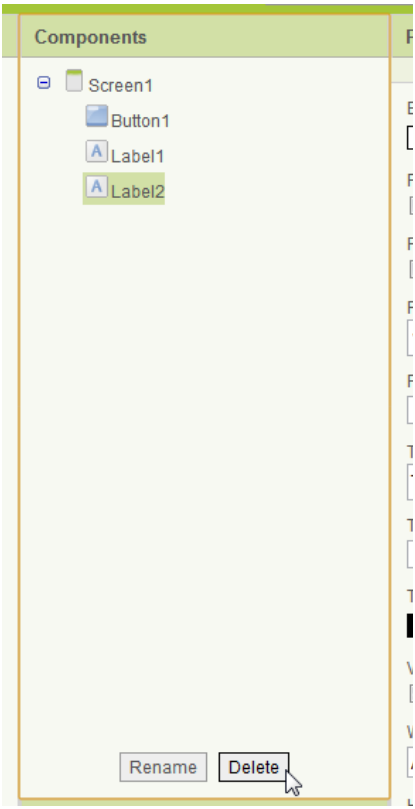
Prima di andare avanti, se ad esempio abbiamo inserito uno strumento di troppo, come ad esempio un'etichetta in più che non ci serve, come mostrato nella figura sotto:



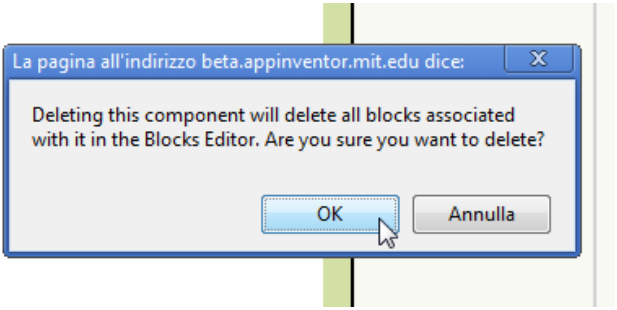
è sufficiente premere il tasto sinistro del mouse sull'etichetta che vogliamo eliminare, ad esempio sulla **Label2**:



Quindi, nella parte bassa della sezione *Components*, premete sul pulsante **Delete**.



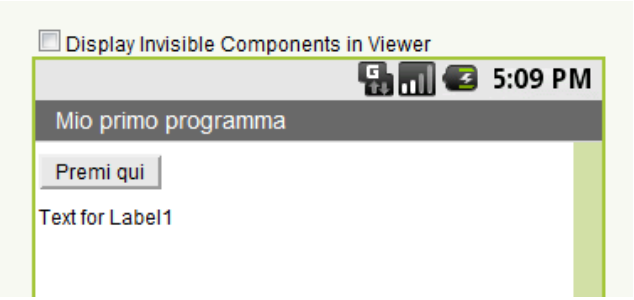
Apparirà la seguente finestra che ci avvisa che verranno anche cancellate le istruzioni di programmazione a lui associati, ma non avendone ancora scritte, non è una nostra preoccupazione, quindi premiamo sul pulsante **OK**.



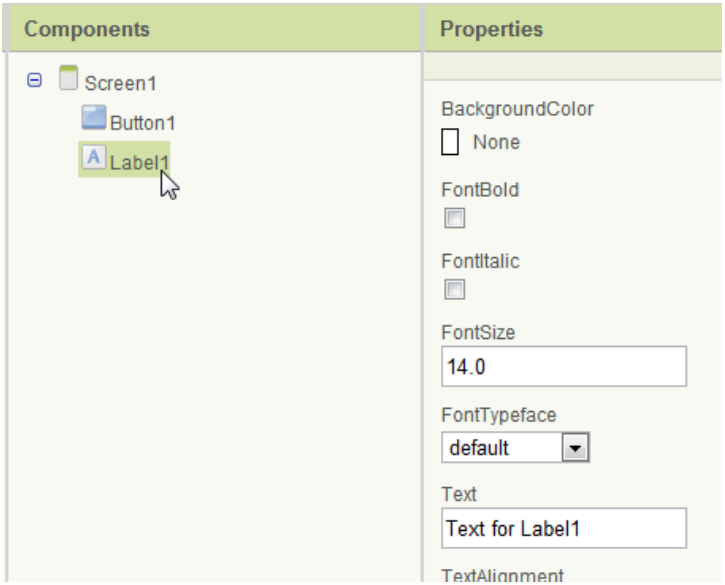
Ma torniamo alla nostra interfaccia, che come abbiamo detto prima, non fa differenza se l'etichetta (cioè la Label) l'abbiamo posizionata sopra o sotto il pulsante.

Nel caso del mio esempio, l'etichetta l'ho posizionata sotto il pulsante.

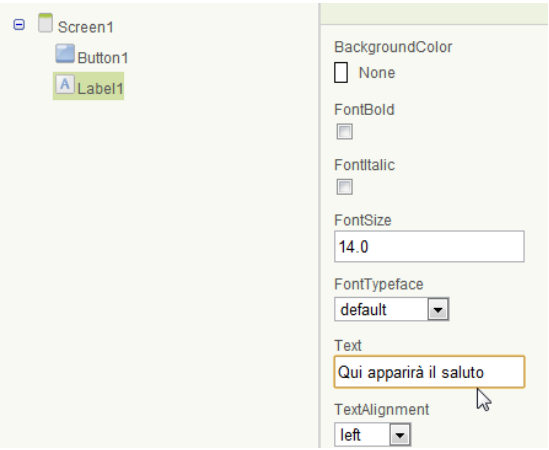
(Volendo potete spostarla anche dopo averla già piazzata)



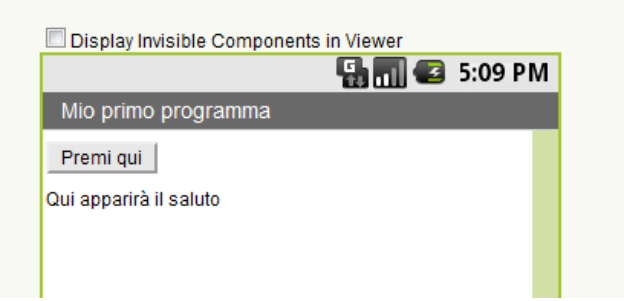
Ora premiamo sull'etichetta, per selezionarla, in modo che nella sezione *Properties* appaiano le proprietà dell'etichetta.



Nella sezione *Properties* premiamo sulla proprietà **Text** che è la proprietà dove c'è il testo mostrato dall'etichetta. Al momento è impostato su *Text for Label1*; noi cambiamolo con la frase *Qui apparirà il saluto*:



Ecco che sull'interfaccia grafica apparirà il testo cambiato, sull'etichetta:



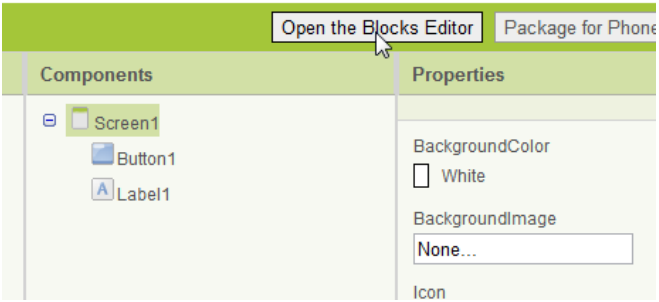
Quello che abbiamo fatto fino ad ora, è stato creare la grafica della nostra applicazione ed inserire, eventualmente, ulteriori strumenti da utilizzare con il nostro programma.

Ricordiamo l'obiettivo del nostro primo programma, cioè permettere all'utente di premere sul pulsante **Premi qui** ed avere un saluto nell'etichetta sotto.

Tutti gli strumenti che abbiamo inserito non sanno cosa devono fare e saremo noi, con la programmazione vera e propria, a dirgli cosa dovranno fare e quando dovranno farlo.

Il programma non dovremo scriverlo qui in questa finestra, ma dovremo accedere ad un'altra finestra di App Invetor che è il **Blocks Editor**.

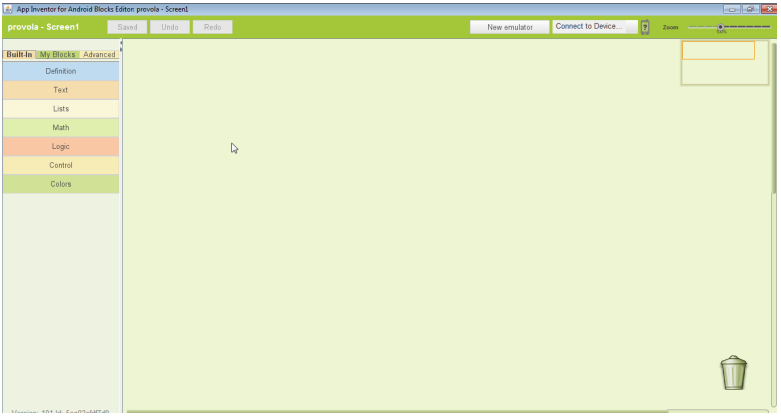
Il **Blocks Editor** può essere aperto premendo sul pulsante **Open the block editor**, come mostrato nella figura qui sotto:



Appena premeremo su questo pulsante, in relazione a quale browser stiamo utilizzando, potrebbe aprirsi una finestra che ci permetterà di scaricare un file (nel caso, ad esempio, di Internet Explorer), oppure potremo vedere il file da scaricare, infondo al browser (nel caso, ad esempio, di Google Chrome).

Scaricate il file sul vostro computer. Il file, che si chiama **AppInventorForAndroidCodeblocks.jnlp**, è molto piccolo, quindi il download avverrà in pochi istanti.

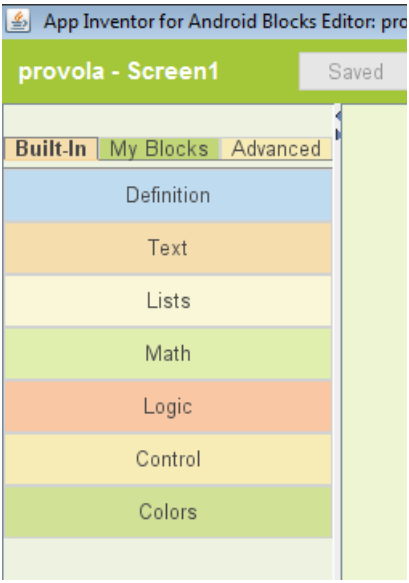
Ora avviate quel file e dopo altri istanti, avrete questa finestra:



Questo è il **Blocks Editor**, ovvero l'editor dei blocchi. Infatti la programmazione del software avverrà mediante dei blocchi, tipo puzzle, che si incastreranno tra loro.

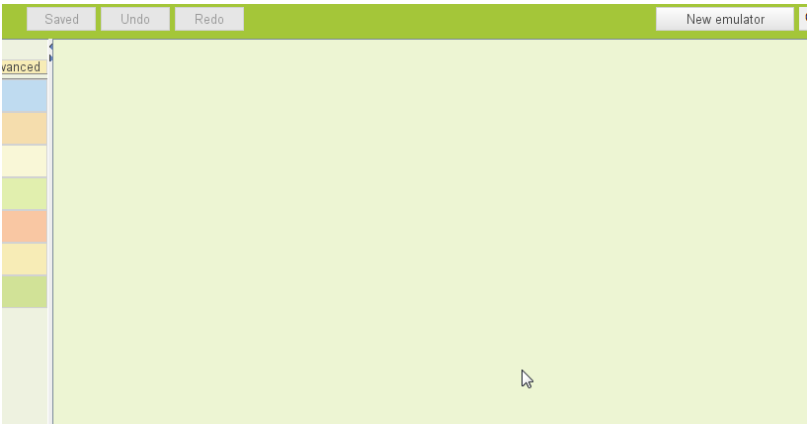
Prima di tutto esaminiamo questa finestra.

Sulla sinistra avremo questa sezione:



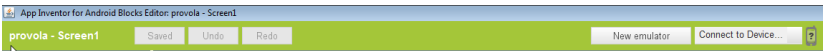
Qui ci sono tutte le istruzioni di programmazione utilizzabili in App Inventor, nonché le istruzioni relative agli strumenti che abbiamo inserito nell'interfaccia. Li vedremo tra poco, perchè dovremo utilizzarli.

Al centro dello schermo, su tutto lo spazio, avremo l'area dove inseriremo i blocchi e quindi dove scriveremo il programma:



In basso a destra avremo il cestino, che ci permetterà di eliminare le istruzioni sbagliate o comunque quelle che vogliamo eliminare:

In alto avremo la barra principale dove ci sono i comandi gestionali:

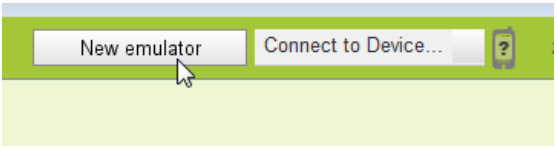


Questi comprendono il pulsante **Save** (per salvare il nostro programma. Vi ricordo che il programma verrà salvato online e non sul

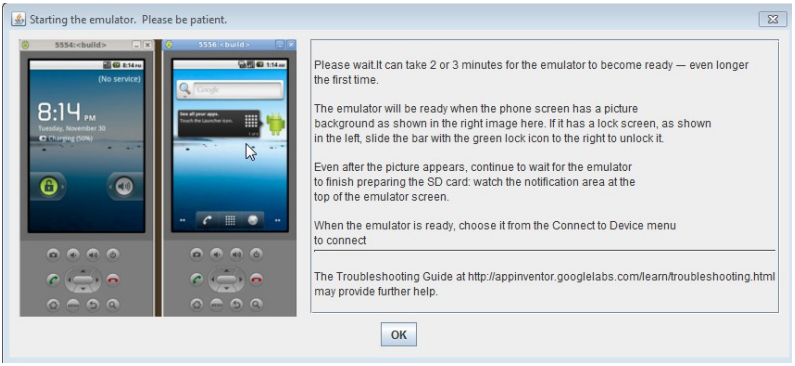
vostro computer), il pulsante **Undo** (che ci permetterà di tornarie indietro all'ultima istruzione scritta), il pulsante **Redo** (che ci permetterà di reimpostare l'eventuale istruzione appena cancellata), il pulsante **New emulator** (che ci permetterà di usare l'emulatore virtuale Android, cioè visualizzare, sul computer, un cellulare Android virtuale per testare i nostri programmi), il pulsante **Connect to device** (che ci permetterà di inviare il programma fatto fin'ora al cellulare o all'emulatore per essere testato).

Prima di iniziare a programmare, cioè a scrivere i comandi che permetteranno di definire il nostro primo programma, vediamo quello che abbiamo creato fino ad ora sull'emulatore Android, cioè sul cellulare virtuale Android.

Per fare questo, premete sul pulsante **New emulator** che vi ho mostrato poco sopra.

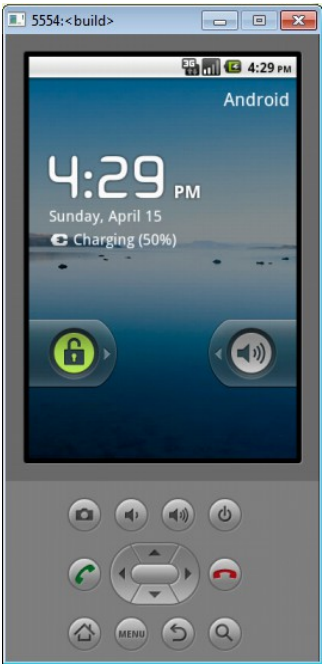


Si apriranno due finestre. La prima è un avviso con alcune indicazioni in inglese, che in pratica ci dicono di aspettare qualche minuto che appare il cellulare virtuale e ci indica come procedere.



Se volete, potete premere sul suo pulsante **OK** per far sparire quell'avviso.

Poi si aprirà la seguente finestra, che non p altro che il nostro emulatore, cioè la riproduzione virtuale di un cellulare Android.



Il cellulare ha la lingua inglese impostata, inoltre non è un vero e proprio cellulare con il quale si può telefonare, ma solo una rappresentazione limitata di un cellulare.

Per usarlo si usa il mouse come se usassimo le dita.

Come sapete, ad esempio, per sbloccare lo schermo, dobbiamo far scorrere l'immagine del lucchetto da sinistra verso destra con le dita. In questo caso, trattandosi dell'emulatore, premete con il puntatore del mouse sul lucchetto e sempre tenendo premuto il tasto sinistro del mouse, spostate il lucchetto verso destra.

Ecco che il display dell'emulatore verrà sbloccato:

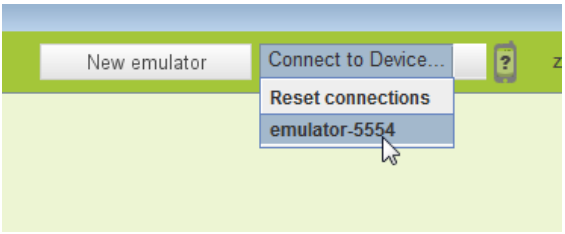
Ora che l'emulatore è attivo, proviamo ad inviargli il nostro programma, per vedere come sarà visualizzato.

Non abbiamo ancora scritto le istruzioni per far funzionare il programma, ma l'interfaccia grafica l'abbiamo già disegnata, quindi possiamo vederla sul display.

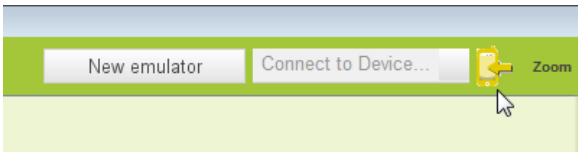
Per poterla vedere, premiamo sul pulsante del **Blocks editor**, che si chiama **Connect to device** e che significa "Connetti al dispositivo" (cioè Connetti al cellulare). Si aprirà un elenco di opzioni, come mostrato nella figura sotto:

Tra queste opzioni c'è **emulator** seguito da un numero. E' possibile che il numero che segue sia differente da quello che vedete in figura, ma questo non pregiudica il funzionamento del sistema.

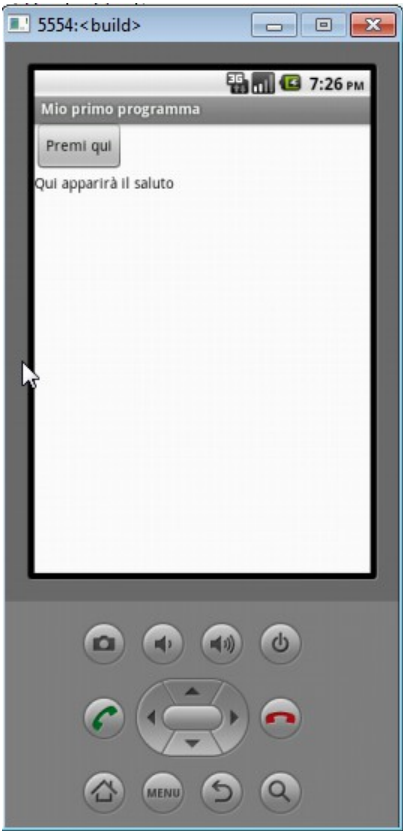
Premete ora il tasto sinistro del mouse sull'opzione **emulator**:



Vedrete lampeggiare il disegno del piccolo cellulare, colorato di giallo. Questo significa che il programma si sta caricando sull'emulatore.



Dopo pochi istanti, vedrete il programma visibile sul display dell'emulatore:



Questo è il metodo che dobbiamo usare per testare i nostri programmi sull'emulatore.

Ma ancora non abbiamo programmato il nostro progetto, quindi torniamo al **Blocks editor** e proviamo a ragionare insieme.

Il nostro programma, come vi ricordo per l'ennesima volta, dovrà scrivere un saluto nell'etichetta, quando l'utente premerà sul pulsante.

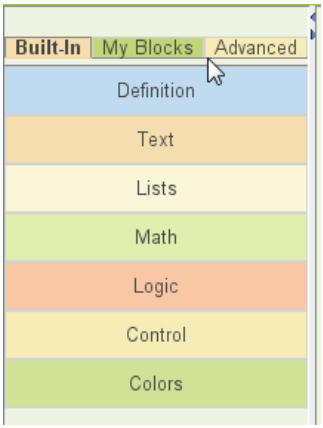
Quindi dobbiamo ragionare così:

- quando si preme sul pulsante, nell'etichetta dovrà apparire un saluto

La prima azione è: "quando si preme sul pulsante".

Andiamo a cercare l'istruzione giusta per questa azione.

Guardiamo la sezione di sinistra del **Blocks editor:**

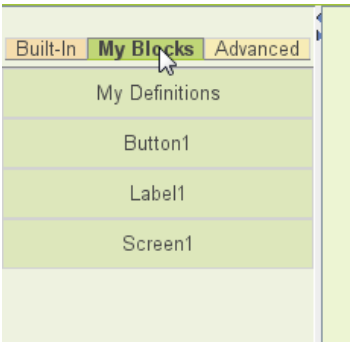


Questa sezione ha tre schede: la scheda **Built-In**, la scheda **My Blocks** e la scheda **Advanced**. Le due schede principali sono:

- La scheda **Built-In** che ha le istruzioni di base del linguaggio di App Inventor
- La scheda **My Blocks** che ha le istruzioni relative agli strumenti che abbiamo inserito nell'interfaccia grafica.

Quella che ci interessa in questo momento, è la scheda **My Blocks**, perchè contiene le istruzioni relative agli strumenti che abbiamo inserito nell'interfaccia grafica, e dato che la prima azione che stiamo

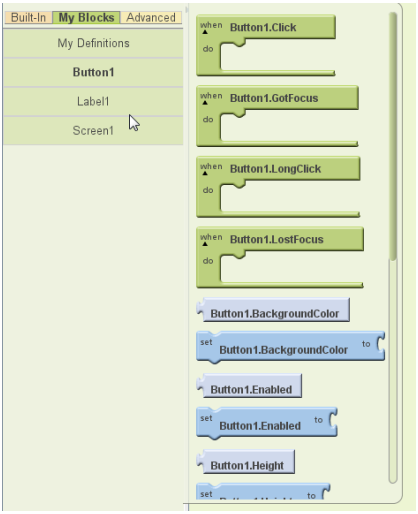
esaminando nel nostro ragionamento è "quando si preme sul pulsante", allora dobbiamo esaminare proprio la scheda **My Blocks**. Premiamo su quella scheda:



Ecco che apparirà l'elenco degli strumenti che abbiamo inserito nell'interfaccia, infatti ci sono **Button1** (il pulsante), **Label1** (l'etichetta) e **Screen1** (lo schermo). In più c'è **My Definitions** che conterrà le variabili che creeremo. Ma questo lo spiegheremo in altri esempi.

Per ora premiamo su **Button1**, perchè il ragionamento che stavamo facendo è "quando si preme il pulsante", quindi **Button1** è l'opzione che conterrà le istruzioni relative al pulsante.

Appariranno, a destra, tutte le possibili istruzioni legate al pulsante:



Le istruzioni sono di più di quelle che si vedono in figura, e si possono far scorrere mediante la barra laterale grigia.

L'istruzione che ci interessa per il nostro progetto è la prima in alto, cioè questa:

Questa istruzione ha un significato ben preciso, che è più facile comprendere se si conosce l'inglese. Ma non spaventiamoci se abbiamo problemi con i termini stranieri, perchè vedremo che non sarà per nulla difficile.

Questa istruzione ha le parole **When Button1.Click do**.

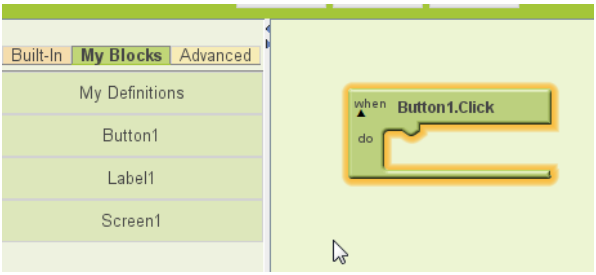
When significa "quando"; **Button1.click** significa "si fa click sul Button1" ed infine **do** significa "fai".

Leggendo l'istruzione tutta attaccata, avremo: "quando si fa click sul Button1 fai", che tradotto in italiano corrente sarebbe: "quando l'utente fa click sul Button1 (il pulsante) allora fai..."

Come vedete non c'è nulla di difficile da capire, anche se chi non ha mai programmato prima d'ora potrà trovarsi ancora in difficoltà.

Sicuramente, continuando a leggere il libro, scoprirete che quello che prima sembrava difficile, dopo diventerà semplice ed ovvio.

Quell'istruzione che abbiamo appena esaminato, che come potete notare, è simile ad un blocco di un puzzle. Ora lo dobbiamo spostare all'interno dello spazio vuoto della finestra del **Blocks editor**, quindi premeteci sopra e spostatelo in un punto qualsiasi dello spazio vuoto, così come mostrato nella figura sotto:



Da questo momento, abbiamo inserito la prima istruzione vera e propria del linguaggio App Inventor.

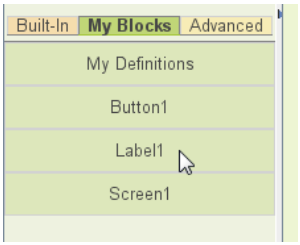
Abbiamo appena detto al programma che quando verrà premuto il pulsante dovrà fare qualcosa. Ma cosa?

Il nostro progetto prevedeva queste azioni:

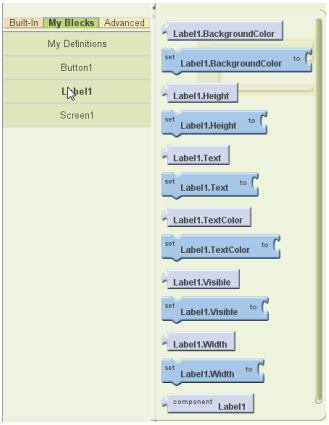
- quando verrà premuto il pulsante, nell'etichetta dovrà apparire un saluto

L'azione "quando verrà premuto un pulsante" l'abbiamo inserita, quindi ora dobbiamo inserire l'azione "nell'etichetta dovrà apparire un saluto".

Rimaniamo nel **Blocks editor.** E sempre nella scheda **My blocks**, troviamo **Label1** che non è altro che l'etichetta che avevamo inserito nell'interfaccia grafica.



Premiamo su **Label1**, in modo da far apparire le sue istruzioni di programmazione sulla sua destra:



Tra queste opzioni troviamo questo blocco:

Questo blocco riporta la seguente istruzione: **set Label1.Text to** che tradotto significa: "imposta il testo della Label1 su...".

Questa istruzione ci permette di impostare un testo all'interno della **Label1** (cioè l'etichetta)

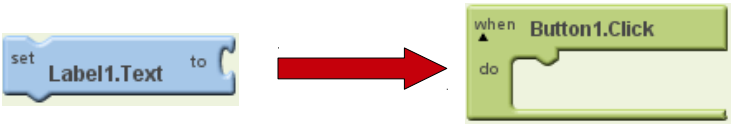
E' sicuramente l'istruzione giusta per noi. Considerato che le azioni del nostro progetto sono:

- quando verrà premuto il pulsante, nell'etichetta dovrà apparire un saluto

L'azione del pulsante premuto lo abbiamo già impostato, ci rimaneva "nell'etichetta dovrà apparire un saluto", quindi l'istruzione che abbiamo appena visto ci permetterà di far apparire il saluto, dato che quell'istruzione si occupa d cambiare il testo dell'etichetta.

Inseriamo questa istruzione dentro l'istruzione che abbiamo inserito precedentemente:

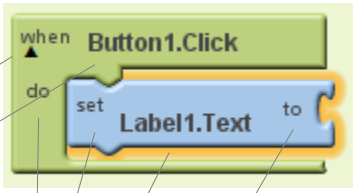
In pratica dobbiamo trascinare, premendoci sopra con il tasto sinistro del mouse, l'istruzione dell'etichetta nell'istruzione già inserita, del pulsante, come mostrato nella figura qui sotto:



Notate attentamente come i due blocchi si possono incastrare perfettamente. Infatti, dopo che abbiamo spostato la **Label1.Text** nel **Button1.Click** avremo questo incastro perfetto:

Avrete sentito anche un click sonoro che conferma l'aggancio.

Questo blocco nel blocco, può sembrare strano, a chi non ha mai programmato, ma se lo proviamo a leggere, vedremo che ha un senso:



When Button1.Click do set Label1.Text to

Tutto questo tradotto letteralmente significa:

- *When* (quando) *Button1.Click* (si fa click sul Button1) *do* (fai) *set* (imposta) *Label1.Text* (il testo dell'etichetta) *to* (su)

Tradotto ancora in italiano corretto, significa:

- *Quando viene fatto click sul Button1, fai che impostare il testo dell'etichetta su*

L'istruzione è quasi completa. Abbiamo appena programmato che il cellulare, quando gli verrà fatto click sul pulsante, dovrà impostare un testo nell'etichetta.

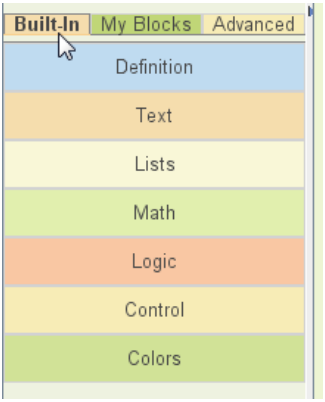
Nel nostro obiettivo, il testo che vogliamo scrivere nell'etichetta, quando verrà premuto il pulsante, era un semplice saluto.

Supponiamo che il saluto sia: *Ciao sono Android*

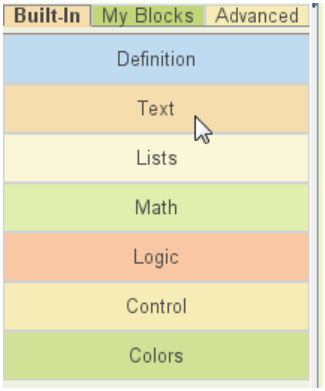
Dobbiamo quindi dire a quel blocco di codice che il testo da scrivere sarà *Ciao da Android*

Esiste un blocco di codice che ci permette di contenere del testo.

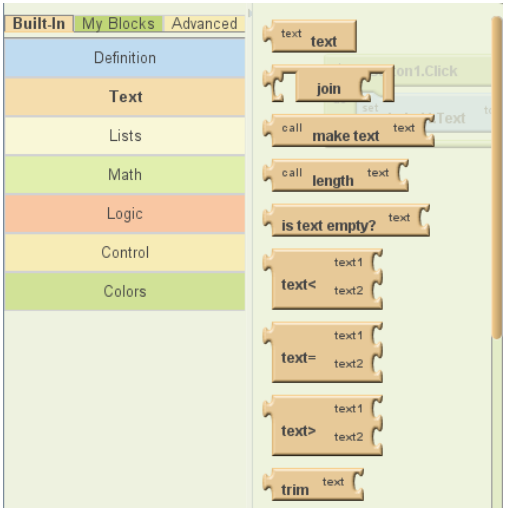
Questo blocco di codice lo troviamo nella scheda **Buit-In**, quind premiamo su questa scheda, come mostrato nella figura sotto:



Tra le istruzioni che fanno parte di Android c'è **Text**. Premiamo su **Text**, come mostrato nella figura sotto:

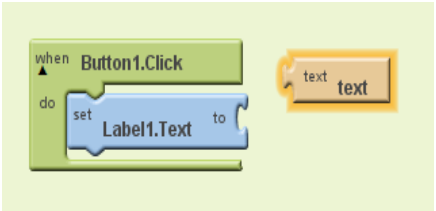


A destra appariranno le istruzioni legate a **Text**, come mostrato nella figura sotto:



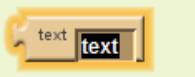
Premiamo sulla sua istruzione che ha nome t**ext text** mostrata in particolare qui sotto:

Trasciniamo, come abbiamo già imparato a fare per gli altri blocchi di codice, questa istruzione **text text** dentro l'area del codice di programmazione:



Ora, premiamo dentro la seconda parola **text** del blocco **text text.**

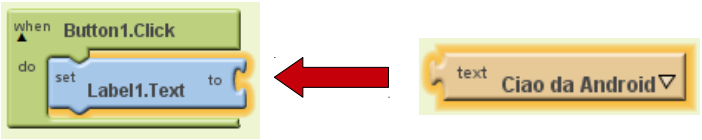
Il blocco si presenterà così:



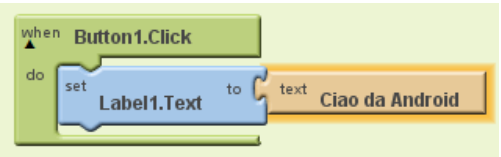
In pratica, il testo **text** è stato evidenziato. Questo ci permetterà di cambiare quella parola con quello che vogliamo noi. Quindi scriviamo la frase che avevamo deciso prima, cioè *Ciao da Android*, quindi premete il tasto INVIO sulla testiera, come mostrato nella figura qui sotto:

Abbiamo appena impostato il blocco **text** con un testo predefinito. Ma l'istruzione **text** qui creata, separata da tutto il codice, non serve a nulla. Dovremo incastrarla al resto del codice.

Trasciniamo questo blocco nel resto del blocco creato precedentemente, come mostrato nella figura sotto:



Come potete notare, i due blocchi si incastrano perfettamente, come avverrebbe per un puzzle. Ecco, finalmente, il nostro codice completo:



Ora, l'istruzione, prende questo significato:

- When Button1.Click do set Label1.Text to text Ciao da Android

che tradotto nella nostra lingua equivale a:

- Quando viene fatto click sul Button1 fai che impostare il testo della Label1 sul testo Ciao da Android

Il modo di programmare Android con App Inventor, è sicuramente facilitato per chi conosce un po' di parole di inglese e ancor di più per chi ha già programmato in altri linguaggi, anche di base.

Ma anche ad un profano, non sarà difficile capire i concetti della programmazione e l'uso, visuale, di questo straordinario linguaggio.

Il nostro primo programma è finalmente pronto per essere testato.

Se avete ancora l'emulatore Android aperto, potremo testare immediatamente l'applicazione, perchè ogni modifica che facciamo al programma, viene trasferita automaticamente all'emulatore.

Se invece avevate chiuso l'emulatore, procedete come già spiegato nei primi capitoli di questo video.

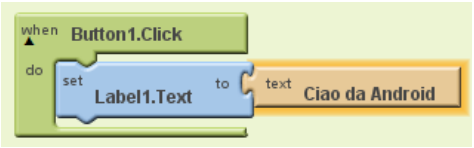
Testiamo la nostra applicazione. Quello che ci troviamo di fronte, prima di premere sul pulsante, è questo:

Nell'etichetta c'è ancora scritto *Qui apparirà il saluto.*

Ora posizioniamo il puntatore del mouse (cioè la freccetta) sul pulsante **Premi qui** e premiamoci sopra il tasto sinistro del mouse.

Ecco cosa accade:

L'etichetta ha cambiato il suo testo ed è apparsa la scritta *Ciao da Android*. Tutto questo, grazie alla semplice istruzione:



Ora potete divertirvi a fare alcuni cambiamenti ed esperimenti.

Ad esempio potete cambiare la frase da far apparire con *Ahia, mi hai fatto male al display*

In modo da far apparire quella frase al posto di *Ciao da Android*.

Oppure potete spostare la posizione del pulsante rispetto all'etichetta, nel disegno dell'interfaccia.

Tutto quello che modificherete, verrà trasmesso immediatamente all'emulatore.

Ma il programma che abbiamo appena creato, è ancora in fase di test, cioè non è un programma chiuso da poter essere usato autonomamente sul cellulare.

Come trasferire il programma al cellulare lo vedremo più avanti. Ora ritengo importante che si capisca veramente quello che abbiamo fatto fino ad ora, cioè che ci si sappia muovere all'interno di App Inventor per rifare un programma da zero leggermente modificato rispetto quello che abbiamo fatto fino ad ora.

Solo in questo modo potremo ritenerci capaci di proseguire con la lettura del libro e l'apprendimento delle ulteriori istruzioni e tecniche di programmazione con App Inventor.

Quindi prepariamoci a ricreare un programma da zero, simile a questo appena sviluppato.

IL NOSTRO SECONDO PROGRAMMA

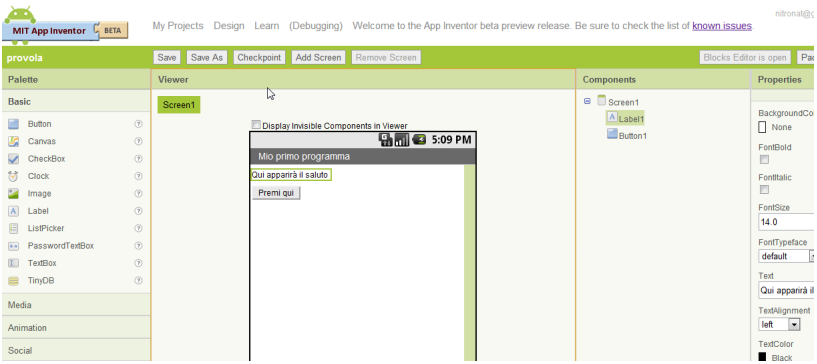
Ora proveremo a creare un programma, del tutto simile a quello precedente, ma con alcune modifiche, senza dare più le istruzioni dettagliatissime che sono state date per il primo programma, in modo da capire, autonomamente, se siamo in grado di continuare nella lettura di questo libro, dato che verranno tralasciati alcuni passaggi fondamentali già spiegati con la realizzazione del primo programma.

Se si trova difficoltà in quei passaggi, sarà opportuno tornare alla lettura di come realizzare il primo programma, dove è spiegato tutto nel dettaglio.

Le eventuali nuove istruzioni e nuovi passaggi invece saranno spiegati sempre dettagliatamente.

Creeremo un programma con due pulsanti ed un'etichetta. Se si preme sul primo pulsante, nell'etichetta verrà scritto *Hai premuto il pulsante uno* se si preme il secondo pulsante, verrà scritto *Hai premuto il pulsante due*.

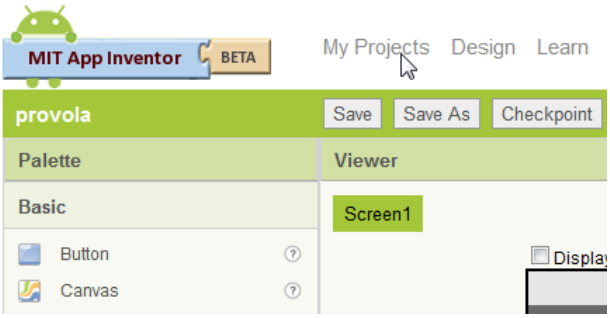
Per prima cosa, andiamo nella finestra di disegno dell'interfaccia grafica:



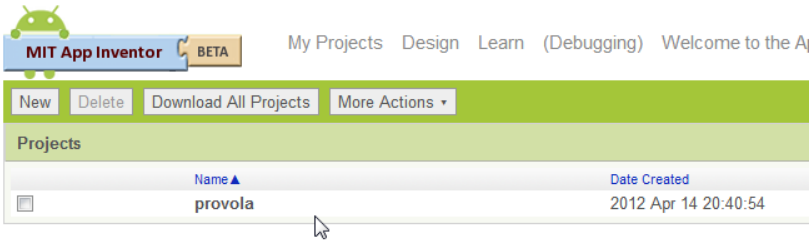
Qui abbiamo ancora il disegno dell'interfaccia del primo programma che abbiamo creato.

Ora ipotizziamo che vogliamo tenercelo per il futuro e quindi creiamo un nuovo programma che abbia un altro nome.

Premiamo sul link **My Projects** (Miei Progetti), che si trova in alto a sinistra, e che ci permetterà di tornare all'elenco dei programmi realizzati fino ad ora:



Questa è la pagina che apparirà:



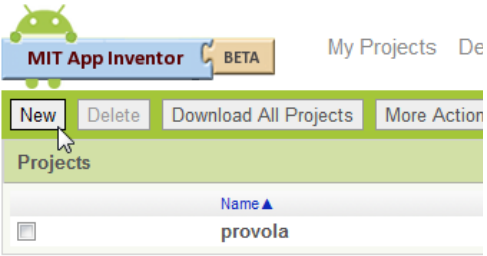
Come potete notare, nell'elenco abbiamo il primo programma che abbiamo realizzato, che si chiama **provola**, e che riporta, sulla sua destra, la data ed ora in cui è stato da noi creato.

Ricordo ancora una volta, che questa pagina, è visibile utilizzando qualsiasi computer connesso ad Internet, in qualsiasi parte del Mondo ci troviamo. Chiaramente potremo accedere a questa pagina solo

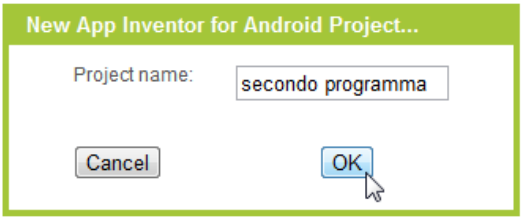
inserendo il nostro username e password che abbiamo creato quando ci siamo registrati a Google. Quindi tutto è visibile solo a noi.

Se vogliamo riaprire il programma **provola**, sarà sufficiente premere sulla parola **provola**, e verremo proiettati direttamente alla pagina di disegno dell'interfaccia grafica di **provola**.

Ma noi vogliamo creare un nuovo programma, a cui vogliamo dare il nome **secondo programma**. Per fare questo, premiamo sul pulsante **New** (Nuovo)**.**

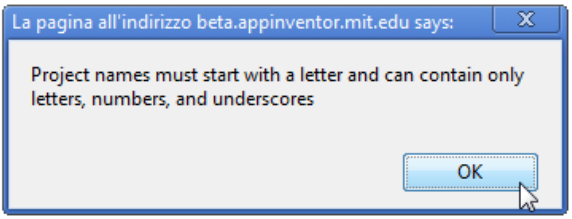


Apparirà la finestra dove dovremo scrivere il nome del programma che vogliamo creare. Scriviamo **secondo programma,** poi premiamo sul pulsante **OK** per confermare:



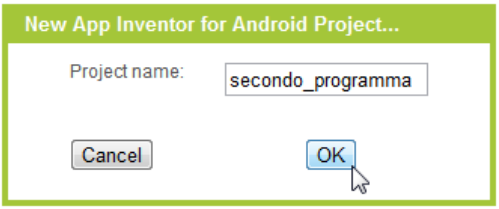
Ma ecco che avremo un errore, a cui vi ho portato volutamente, per farvi notare cosa accade in certe situazioni.

Ecco l'errore che appare:

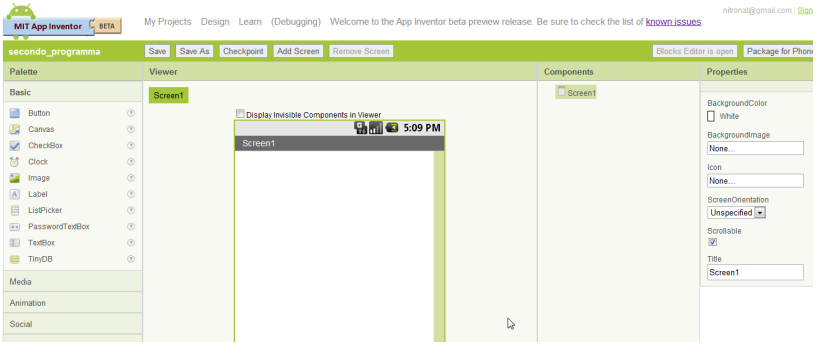


Questo messaggio ci informa che abbiamo scritto un nome di programma che non è conforme alle regole, perchè, nel nostro caso, abbiamo messo uno spazio tra due parole, cioè tra **secondo** e **programma**. App Inventor, al momento, non accetta gli spazi, quindi, se vogliamo comunque tenere quel nome di programma, dobbiamo cambiarlo in **secondo_programma**, oppure in **secondoprogramma,** senza spazi.

Quindi, nella finestra di errore, premiamo sul pulsante **OK**, e ci verrà riproposta la finestra in cui dobbiamo scrivere il nome del programma. Questa volta scriviamo **secondo_programma** (mettendo il segno di underscore tra le due parole) e poi premiamo sul pulsante **OK**.

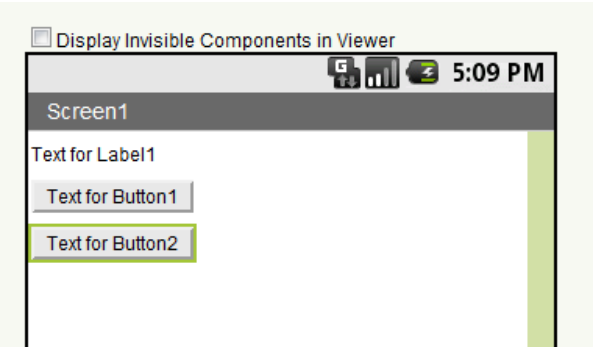


Dopo pochi istanti, si aprirà la finestra che ci permetterà di disegnare l'interfaccia da zero.



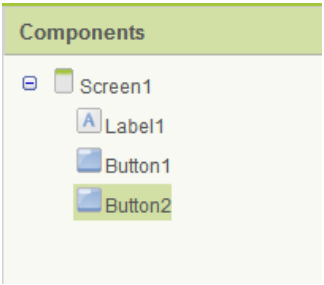
Come già spiegato per il primo programma, inseriamo nell'interfaccia, una **Label** (etichetta) e due **Button** (pulsanti). Posizionateli nell'ordine come preferite.

Ecco come potrebbe apparire la vostra interfaccia:



Nell'immagine qui sopra, ho inserito prima una Label, poi i due Button.

Sulla sua destra, nella sezione **Components** (componenti), troviamo la struttura della grafica del programma:



Sono elencati la **Label1**, il **Button1** ed il **Button2**. Tutti sono "figli" dello **Screen1** (schermo).

Da questa struttura siamo in grado di imparare che, se inseriamo più **Palette**, queste saranno nominate con il nome della Palette seguita da un numero progressivo. In pratica, se inserissimo un'altra etichetta, questa verrebbe chiamata Label2. Se ne inserissimo un'altra ancora, questa verrebbe chiamata Label3, ecc...

Come abbiamo fatto nella realizzazione del primo programma, cambiamo il testo del **Button1** con la frase *Pulsante uno* ed il testo del **Button2** con la frase *Pulsante due*.

Per cambiare il testo dei **Button** dovete selezionarli singolarmente e cambiare le loro proprietà **Text**. Vi ricordo che la sezione delle **Properties** (proprietà) si trovano sulla destra.

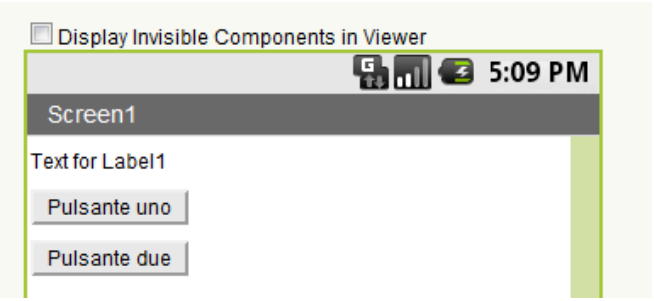
Per rendere più schematico ed immediata la comprensione delle modifiche che dovete fare, userò questo modo di scrivere:

Properties da modificare per ogni **Palette**

Palette	Properties	
Button1	Text	Pulsante uno
Button2	Text	Pulsante due

Per capire la tabella qui sopra, che ritroveremo anche nella spiegazione dei prossimi programmi, esaminiamo, ad esempio, la riga uno (quella sotto i titoli).

Nella prima riga è indicata, sotto la colonna **Palette**, lo strumento **Button1**, che è quello su cui vogliamo eseguire la modifica, quindi, nella colonna **Properties**, troviamo la parola **Text**, che è la proprietà che vogliamo modificare, infine nell'ultima colonna, troviamo la frase *Pulsante uno*, che è la modifica che vogliamo apportare a quella proprietà. Ecco come appare la nostra interfaccia dopo queste prime modifiche.



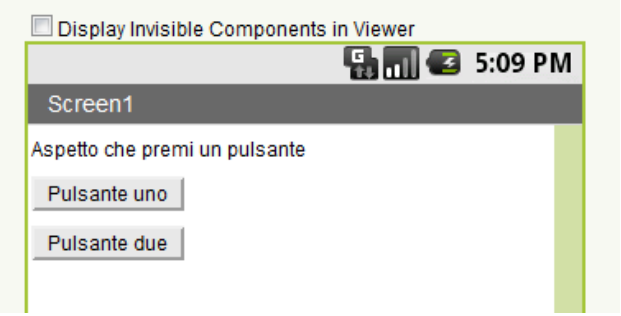
Inoltre cambiamo il testo della **Label1** con la scritta *Aspetto che premi un pulsante*.

Anche qui vi metto la spiegazione schematizzata in tabella, che d'ora in poi verrà usata come unica spiegazione:

Properties da modificare per ogni **Palette**

Palette	Properties	
Label1	Text	Aspetto che premi un pulsante

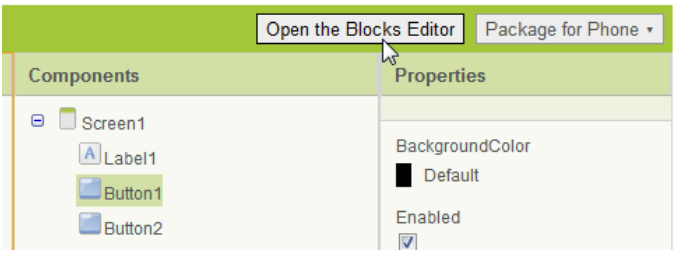
Ecco come apparirà nell'interfaccia grafica:



La nostra interfaccia è pronta per essere programmata.

Se avete creato questo nuovo programma, senza chiudere il **Blocks editor** del programma precedente, vi troverete la finestra del **Blocks editor** sempre attivo e senza alcuna istruzione al suo interno, in quanto App Inventor ha cancellato tutte le istruzioni del programma precedente, per dar spazio al nuovo programma.

Se invece avevate chiuso il **Blocks editor,** allora dovrete premere su **Open the Blocks Editor**, come mostrato in figura sotto:



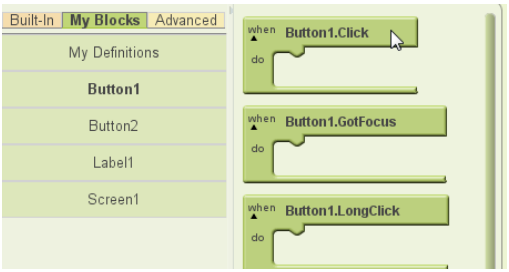
Per i prossimi programmi che creeremo, vi verrà detto solo di aprire il **Blocks Editor**, senza mostrarvi più alcuna figura, quindi dovrete essere in grado da soli di fare questo, sia che il **Blocks editor** risulti già aperto o meno. Per eventuali dimenticanze su come funziona l'apertura del **Blocks editor**, vi consiglio di tornare alla spiegazione del Primo programma.

Ecco che abbiamo il nostro **Blocks editor** completamente pulito:

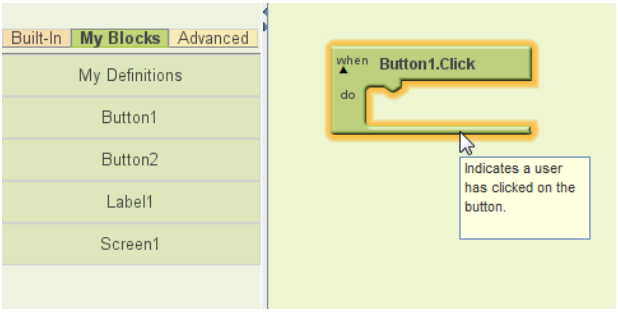
Ricordandoci che l'obiettivo di questo nostro secondo programma è quello di far scrivere, all'interno dell'etichetta, la frase *Hai premuto il pulsante 1*, oppure *Hai premuto il pulsante due* in base a quale pulsante avremo premuto sul display, dovremo programmare i due pulsanti, cioè i due Button.

Partiamo programmando il **Button1**.

Nella scheda **My Blocks**, premete su **Button1** e trascinate il blocco **Button1.Click** dentro l'area di programmazione:



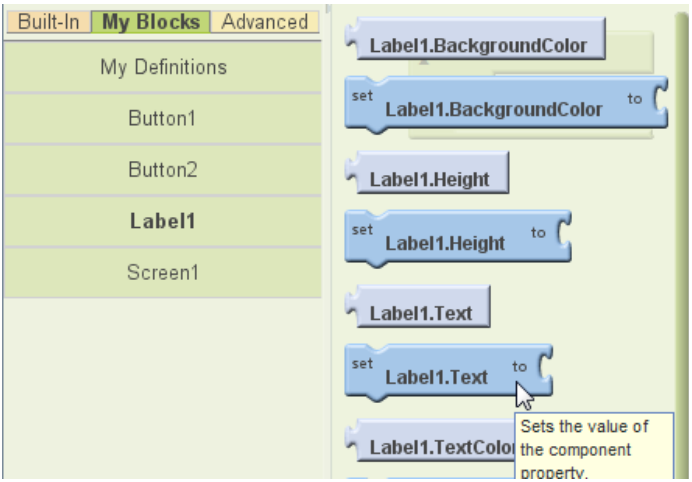
Ecco l'istruzione internamente all'area di programmazione:



Come abbiamo imparato nella realizzazione del primo programma, questa istruzione indica "quando si fa click sul Button1 allora fai".

Noi vogliamo far scrivere una frase nella **Label1** (etichetta), quindi ora dobbiamo prendere l'istruzione che ci permette di inserire/modificare la frase all'interno della **Label1**.

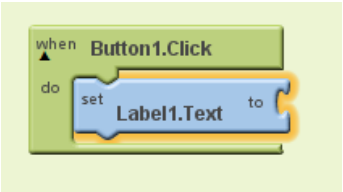
Per farlo, dobbiamo premere nella scheda **MyBlocks**, quindi su **Label1**, quindi sull'istruzione **set Label1.Text to:**



D'ora in poi, per indicarvi l'istruzione del **Blocks editor** da usare, troverete una tabella, come quella qui sotto, che non fa altro che ripetere quello che ho appena detto sopra:

Scheda	Palette	Istruzione
My Blocks	Label1	Set Label1.Text to

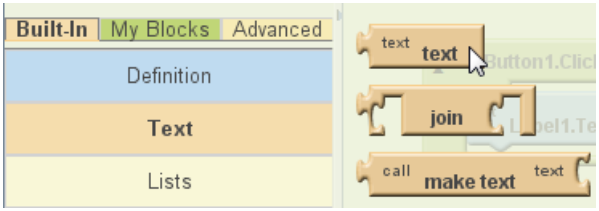
Trasciniamo questa istruzione all'interno dell'area di programmazione e incastriamola con quella del Button1.



A questo punto, abbiamo programmato che quando verrà fatto click sul Button1, il testo della Label1 verrà modificato. Non ci resta che dire con quale frase dovrà essere modificato. Useremo l'istruzione **text**.

Come spiegato poche righe più su, userò una tabella per indicare quale istruzione inserire:

Scheda	Palette	Istruzione
Built-In	Text	text text



Trasciniamo ed incastriamo l'istruzione **text text** al resto del programma fino ad ora realizzato.

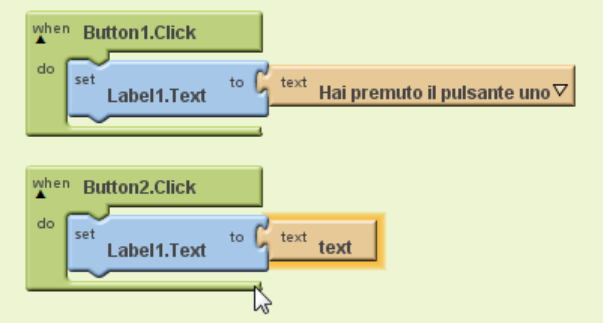
L'istruzione **text text** permette di essere impostata con qualsiasi testo che vogliamo noi. Per impostare il testo, premiamo sulla parola **text**, quindi scriviamo la frase *Hai premuto il pulsante uno*

Abbiamo terminato di programmare la prima istruzione, cioè quella che ci permette di far eseguire l'azione voluta, alla pressione del **Button1**.

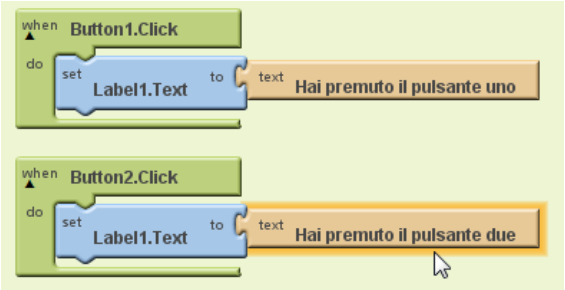
Ora facciamo la stessa cosa per il **Button2**. Ecco la tabella che vi indica quali istruzioni andranno incastrate tra loro, in successione:

Scheda	Palette	Istruzione
My Blocks	Button2	Button2.Click
My Blocks	Label1	Set Label1.Text to
Built-In	Text	text text

Ecco l'istruzione all'interno dell'area di programmazione:

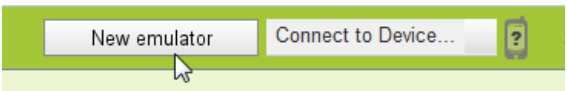


Anche per quest'ultima istruzione, dobbiamo modificare la parola **text** con la frase *Hai premuto il pulsante due*

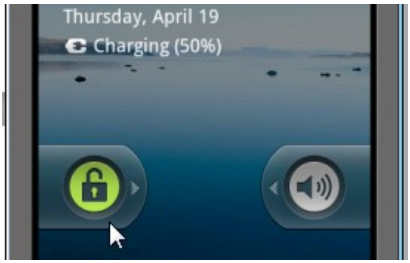


Il nostro secondo programma è terminato. Nell'area di programmazione ci sono le due istruzioni, ognuna delle quali permette di far apparire, nella **Label1**, una frase diversa, in relazione a quale pulsante verrà premuto.

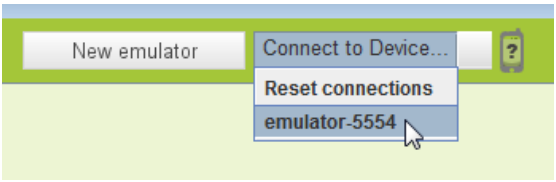
Possiamo testare l'applicazione, premendo su **New emulator** per avviare l'emulatore:



Quando l'emulatore si sarà avviato, ricordatevi di sbloccarlo, trascinando a destra il selettore di blocco schermo:

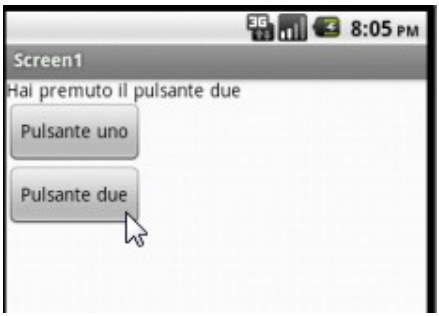


Ora, nel **Blocks editor**, premete su **Connect to device**, quindi su **emulator-5554** (o quello che apparirà a voi).



Ecco che il programma verrà avviato. Testatelo e vedrete che svolgerà le funzioni che gli abbiamo programmato.

Per qualsiasi dubbio o problema relativo all'avvio del programma con l'emulatore, vi invito a rivedere le istruzioni dettagliate scritte per il Primo programma.

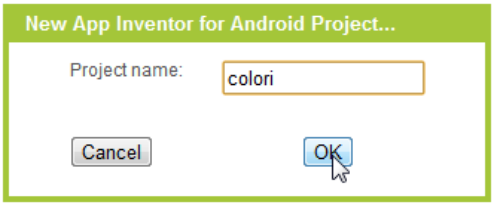


IL NOSTRO TERZO PROGRAMMA

Con questo terzo programma, che ancora una volta non farà nulla di particolarmente interessante, prenderemo sempre più confidenza con App Inventor e creeremo delle scritte a cui cambieremo colore attraverso i pulsanti.

Nonostante, come ripeto, non sarà un programma utile; questo terzo programma ci permetterà di approfondire l'uso delle proprietà delle **Palette.**

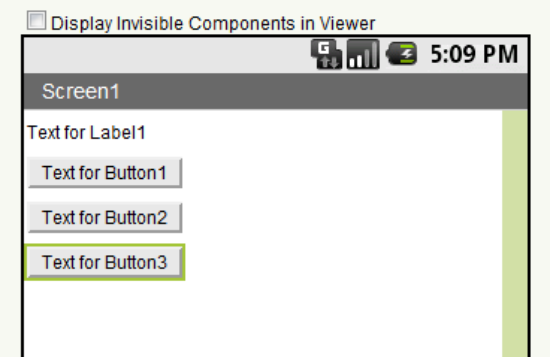
Creiamo un nuovo programma e chiamiamolo *colori*, quindi premiamo sul pulsante **OK**.



(Ricordatevi che se non ricordate come procedere a creare un nuovo programma, dovete ripassare le pagine dove abbiamo spiegato come creare il Primo programma oppure Secondo Programma)

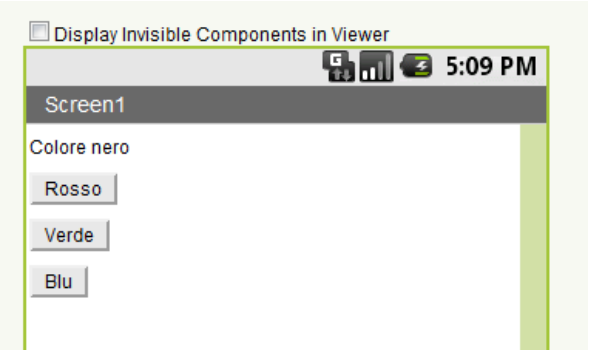
Nella pagina di App Inventor dove si disegna l'interfaccia, inseriamo le seguenti **Palette**:

- **Label**
- **Button**
- **Button**
- **Button**



Properties da modificare per ogni **Palette**

Palette	**Properties**	
Label1	Text	Colore nero
Button1	Text	Rosso
Button2	Text	Verde
Button3	Text	Blu



Ora cerchiamo di aggiustare le dimensioni dei pulsanti, dato che come sono visualizzati ora, non sono belli da vedere.

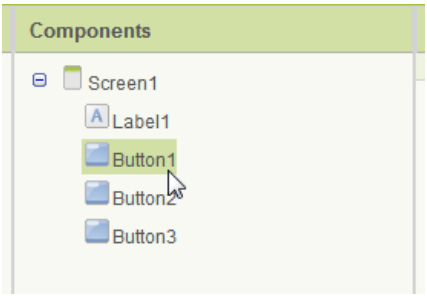
Se volessi dirvi, per il **Button1**, in modo schematico, come procedere, scriverei:

Properties da modificare per ogni **Palette**

Palette	Properties	
Button1	Width	300
Button1	Height	30

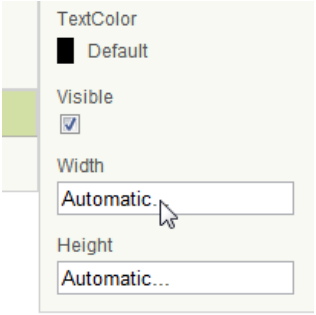
Ma vediamolo in dettaglio, in modo da capire meglio, almeno per questa volta.

Premiamo sul **Button1** nei **Components**.



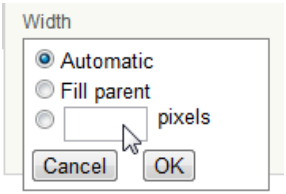
In questo modo si seleziona la **Palette** che vogliamo modificare. Sulla destra verranno visualizzate le sue proprietà.

Tra le proprietà visualizzate (nelle **Properties**), cercate tra quelle verso il basso (fatele scorrere se non le vedete tutte), e troverete la proprietà **Width**



Al momento è preimpostata **Automatic**, che significa *Automatico*, ovvero che il pulsante si ridimensiona in base al testo che c'è scritto al suo interno. Ecco perchè i tre pulsanti hanno tutti dimensioni diverse. Si sono adattati ognuno al testo.

Cambiamo questa proprietà premendoci sopra. Apparirà un rettangolo di opzioni, come mostrato in figura:

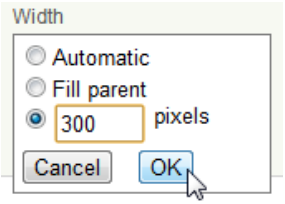


Le opzioni possibili sono tre. Ecco la loro spiegazione:

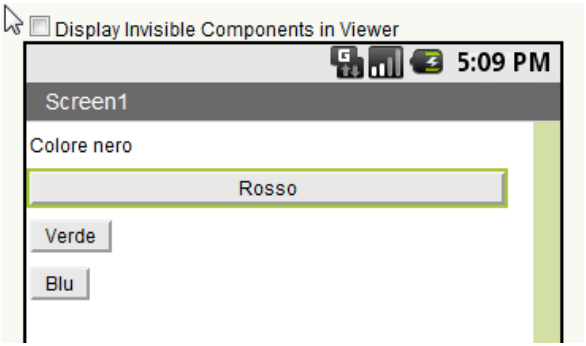
- **Automatic**: adatta il pulsante alla lunghezza del testo scritto dentro il pulsante stesso;
- **Fill parent**: allarga il pulsante dall'estremità del margine sinistro all'estremità del margine destro;
- **Pixels**: accetta un numero che sarà la larghezza in pixel.

L'opzione che interessa a noi è **Pixel** ed il numero che viogliamo inserire è **300**.

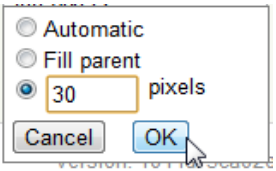
Scriviamo quel numero e premiamo sul pulsante **OK**.



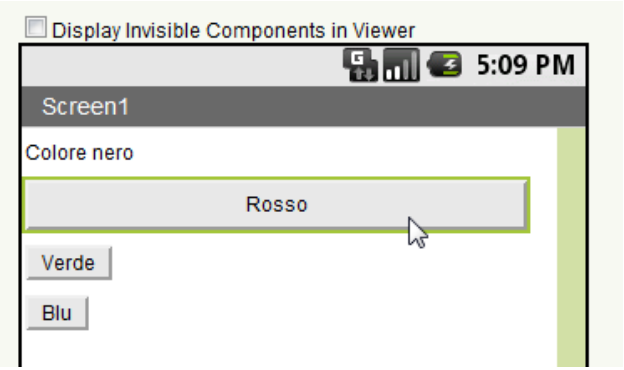
Ecco che il **Button1** verrà allargato a 300 pixel, come mostrato nella figura sotto:



Ora allarghiamo questo pulsante a **30** pixel. Non vi sarà difficile farlo. Selezionate il **Button1** e tra le sue proprietà modificate i pixel di **Height**:



Ora premete sul pulsante **OK** ed ecco il **Button1** che avrà le dimensioni da noi scelte:

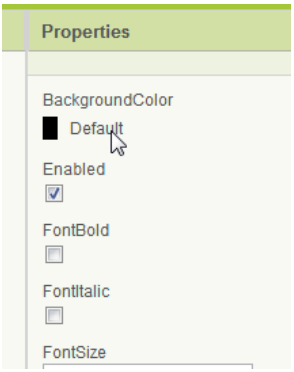


Cambiamo le impostazioni anche degli altri due pulsanti, con le stesse dimensioni del **Button1**.

Properties da modificare per ogni **Palette**

Palette	**Properties**	
Button2	Width	300
Button2	Height	30
Button3	Width	300
Button3	Height	30

Ora selezioniamo nuovamente il **Button1** e tra le sue proprietà cerchiamo **BackgroudColor**. Questa proprietà significa "colore di sfondo" e si riferisce al colore di sfondo del pulsante.

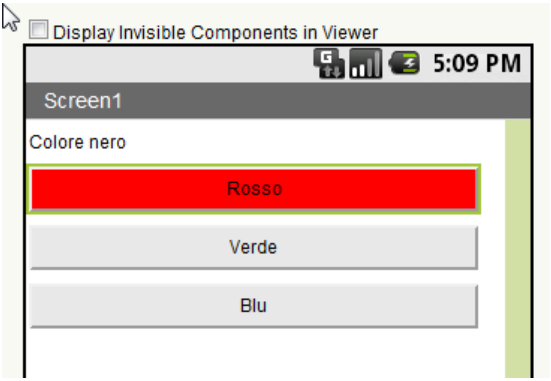


Al momento è preimpostato il colore di **default**. In considerazione che il nostro pulsante ha al suo interno la parola *Rosso* e che questo pulsante dovrà colorare di rosso il testo della **Label1**, agevoliamo l'utente che userà il nostro programma, colorando lo sfondo del pulsante di colore rosso. Quindi premiamo sulla proprietà **BackgroundColor** e vedremo apparire diversi colori selezionabili:

I colori sono scritti in inglese e mi sembra superfluo tradurveli in italiano, dato che a fianco di ognuno c'è un quadrato con il colore.

Selezioniamo quindi il colore **red**.

Ecco come apparirà il pulsante nella nostra interfaccia:

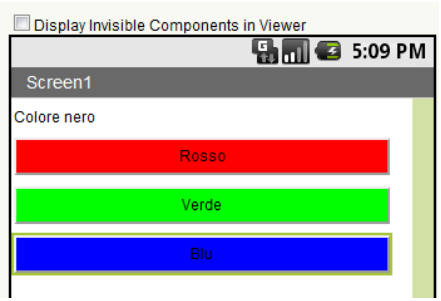


Ora cambiamo anche gli altri colori degli altri pulsanti. Qui sotto lo schema delle proprietà da cambiare:

Properties da modificare per ogni **Palette**

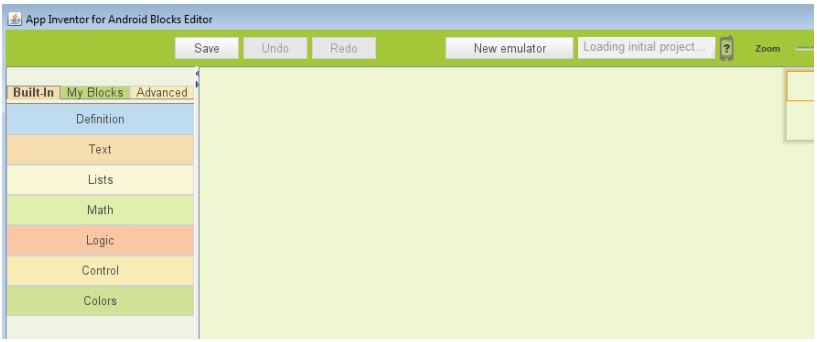
Palette	Properties	
Button2	BackgroundColor	Green
Button3	BackgroundColor	Blue

Ecco come si presenterà l'interfaccia della nostra applicazione:



Ora passiamo alla programmazione. Apriamo il **Blocks editor**.

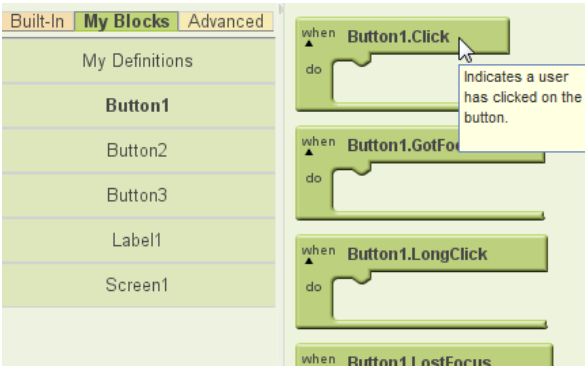
Ecco che avremo l'area di programmazione completamente vuota:



Il nostro obiettivo è far cambiare la scritta ed il colore della **Label1** in relazione al pulsante premuto.

Iniziamo con il programmare il **Button1**.

Dalla scheda **My blocks** selezioniamo il **Button1**, quindi prendiamo la sua istruzione **when Button1.Click do**



trasciniamola dentro l'area di programmazione.

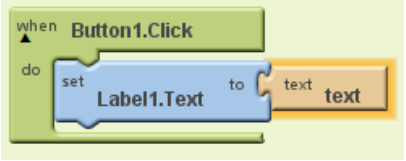
Questa è la nota istruzione che ci permette di programmare, quando verrà fatto click sul **Button1**.

Il **Button1** dovrà cambiare la scritta della **Label1** e dovrà anche cambiare il colore della scritta della **Label1**.

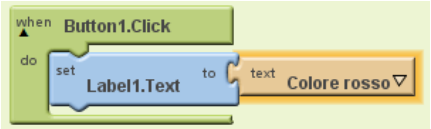
Per cambiare la scritta della **Label1** sappiamo già come fare.

Inseriamo le seguenti istruzioni nell'area di programmazione, attaccandole all'istruzione del **Button1**:

Scheda	Palette	Istruzione
MyBlocks	Label1	Set Label1.Text to
Built-In	Text	text text



Ora modifichiamo la frase dentro l'istruzione **text text,** inserendo la frase *Colore rosso*



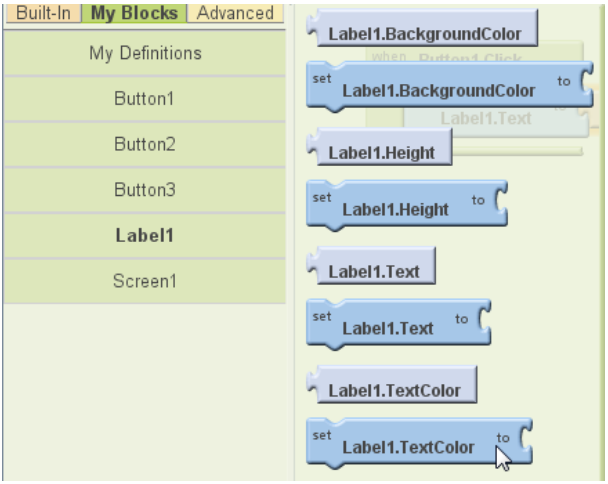
Fino ad ora, abbiamo detto al programma di cambiare la scritta della **Label1** con la scritta *Colore rosso*, solo quando verrà fatto click sul **Button1**

Ora però dobbiamo anche dire al programma di cambiare il colore del testo di quella scritta.

Dovendo cambiare il colore della scritta contenuta nella **Label1**, è evidente che dovremo agire proprio su questa **Palette**.

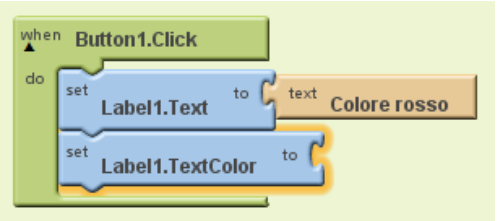
Selezioniamo la seguente istruzione:

Scheda	Palette	Istruzione
MyBlocks	Label1	Set Label1.TextColor to



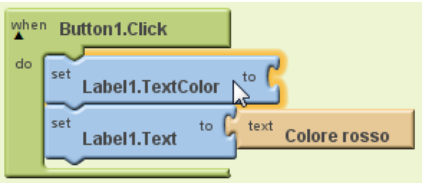
E' intuibile dal nome dell'istruzione, che **set Label1.TextColor to** permette di settare il TextColor (cioè colore del testo) della **Label1**.

Questa istruzione va inserita sempre nell'istruzione del **Button1.Click**.

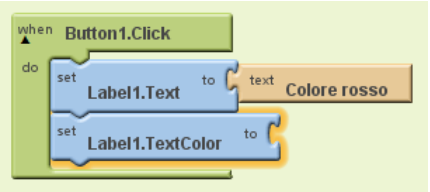


Notate come l'istruzione si inserisce perfettamente nelle istruzioni del programma già esistente.

E' indifferente se mettiamo l'istruzione del settaggio colore, prima:



o dopo:



Perchè la lettura di questa istruzione, nel primo caso sarebbe:

- quando viene fatto click sul Button1, imposta il colore della Label1 e poi cambia la scritta della Label1;

nel secondo caso sarebbe:

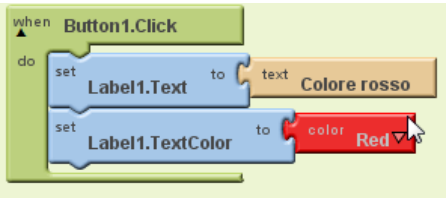
- quando viene fatto click sul Button1, cambia la scritta della Label1 e poi imposta il colore della Label1

Ora però dobbiamo dire al programma, con quale colore dovrà essere colorato il testo della **Label1**.

Considerato che il colore è un'istruzione generica, cioè un valore che può essere comune ed utilizzato anche da altre istruzioni, lo troveremo nella scheda **Built-In**.

Infatti, infondo a quella scheda, troveremo le istruzioni **Colors**, e premendoci sopra, avremo l'elenco dei diversi colori utilizzabili.

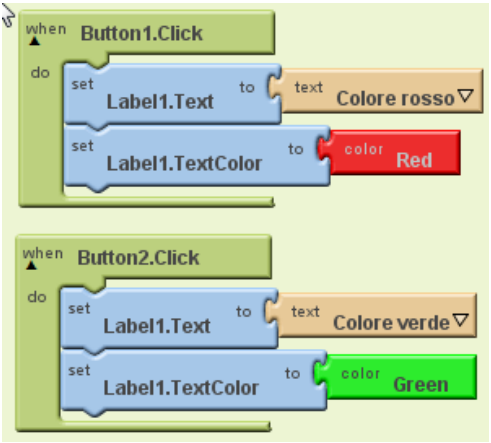
Trasciniamo l'istruzione **Red** (che significa rosso) all'interno dell'area di programmazione e colleghiamola all'istruzione che imposta il colore della **Label1**.



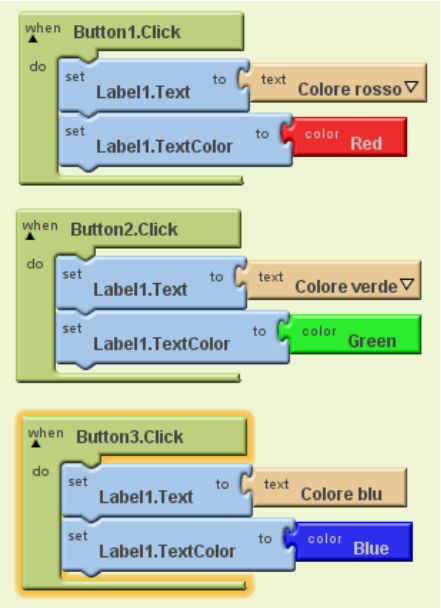
L'istruzione relativa alle operazioni che deve eseguire il **Button1** è terminata. Ora dobbiamo fare la stessa cosa per gli altri due **Button**.

Vi metto di seguito le istruzioni da inserire e cambiare, quindi l'immagine del programma terminato. Dovreste essere in grado di fare tutto da soli:

Scheda	Palette	Istruzione	
MyBlocks	Button2	When Button2.Click do	
MyBlocks	Label1	Set Label1.Text to	
Built-In	Text	text text	Colore verde
MyBlocks	Label1	Set Label1.TextColor to	
Built_in	Colors	Green	

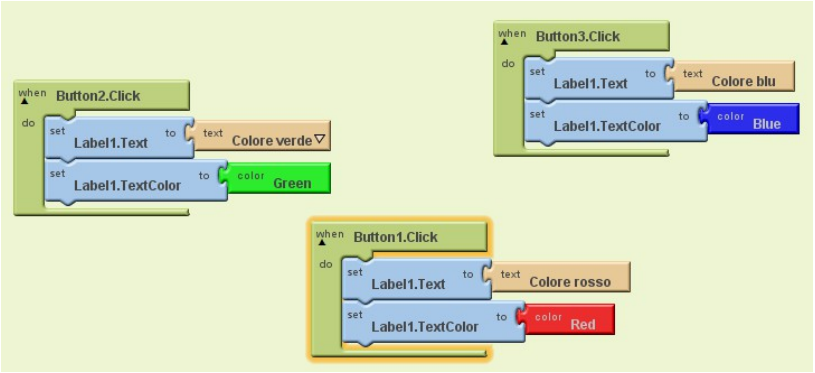


Terminiamo con la programmazione dell'ultimo pulsante che vi mostro nell'immagine, senza ripetere le istruzioni che già avete imparato ora:



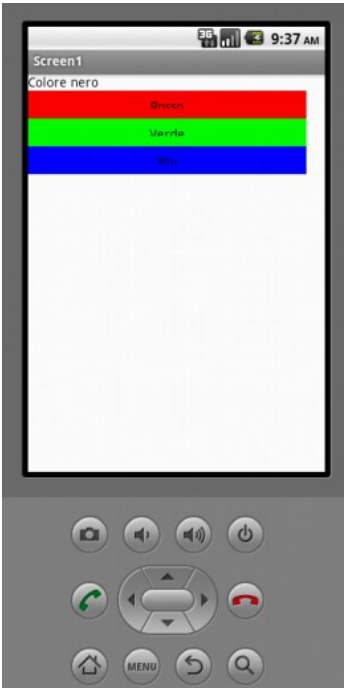
Altra cosa importante da sapere è che, anche se cambiamo l'ordine della programmazione, se spostiamo i blocchi di codice in parti diverse, anche in ordine diverso, verranno eseguite lo stesso, senza alcun problema. L'importante è che ogni blocco di codice sia corretto.

Ecco, ad esempio, una distribuzione dei blocchi di codice in posizioni sparpagliate. Il programma funzionerà ugualmente.



Trasferiamo il programma all'emulatore per testarlo.

Ecco come apparirà nell'emulatore:



Ora proviamo a premere sul pulsante blu. Ecco cosa succederà:

La scritta della **Label** (etichetta) cambierà con la frase *Colore blu* e la scritta stessa si colorerà di blu.

Con questo terzo programma, abbiamo imparato altre cose nuove e soprattutto abbiamo imparato a muoverci, con più sicurezza, tra le varie istruzioni ed impostazioni.