CoderDojo Firenze

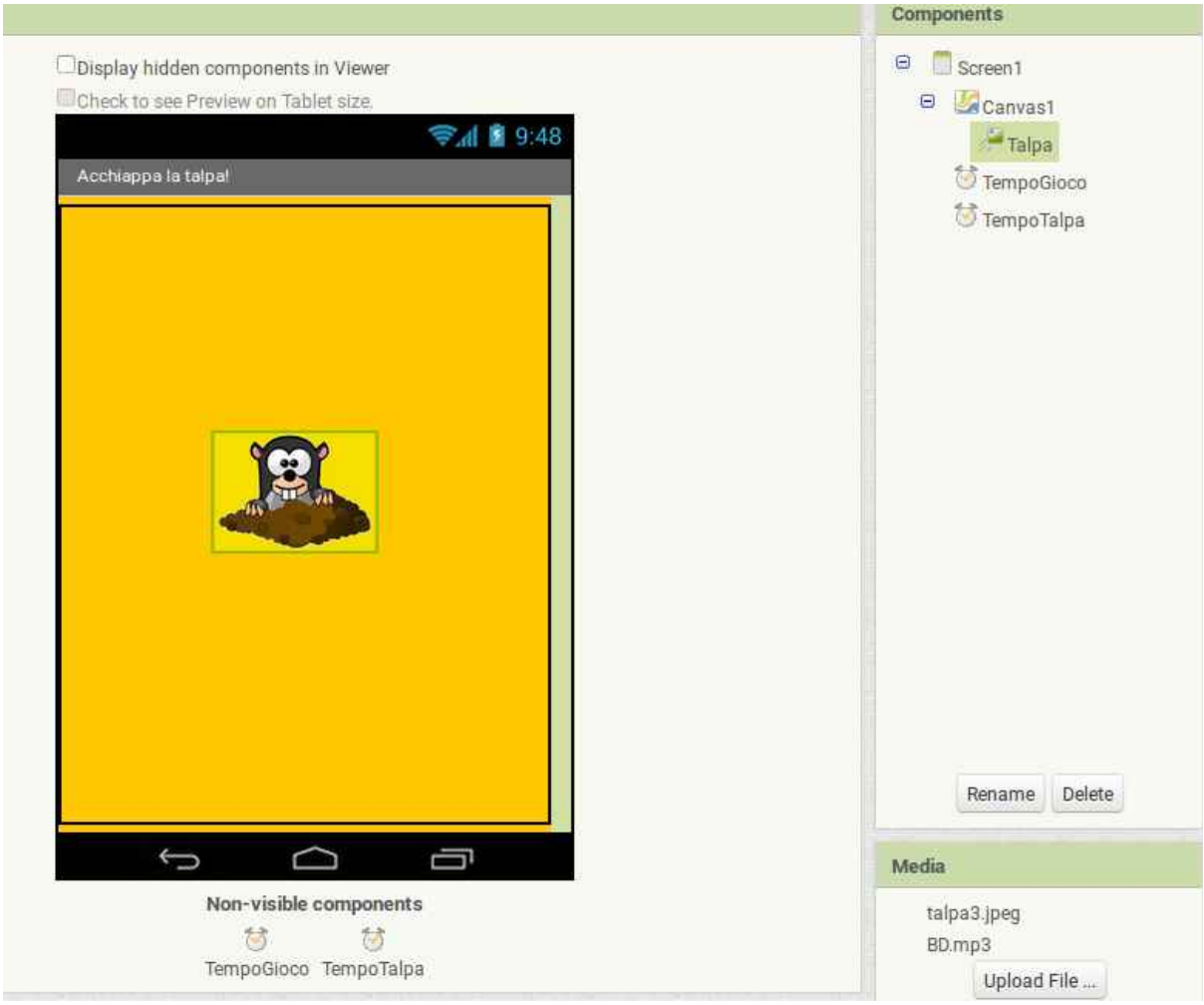
'*Acchiappa la talpa*' con AppInventor

Ci sono due tipi di visualizzazione dell'ambiente: Designer e Blocchi, selezionabili tramite pulsante in alto a destra dello schermo.

NB: Ogni tanto salvare le modifiche al progetto dal menu Progetto, scegliendo **Salva Progetto**

Selezionare la visualizzazione **Designer** (default).

- Scaricare tramite Google Play Store la app **MIT AI2 Companion** e dimenticarsela
- Andare sul sito di AppInventor http://appinventor.mit.edu/explore/ e cliccare su 'Create App': collegarsi col proprio account Google
- Selezionare la lingua italiana
- Si crea un nuovo progetto con nome “acchiappalatalpa”
- Il progetto si apre vuoto con uno Screen denominato **Screen1**.
- Nel tab **Proprietà** a destra dello schermo si imposta la proprietà **Titolo** a “Acchiappa la talpa!”
- Si imposta il **colore di sfondo** a Arancione

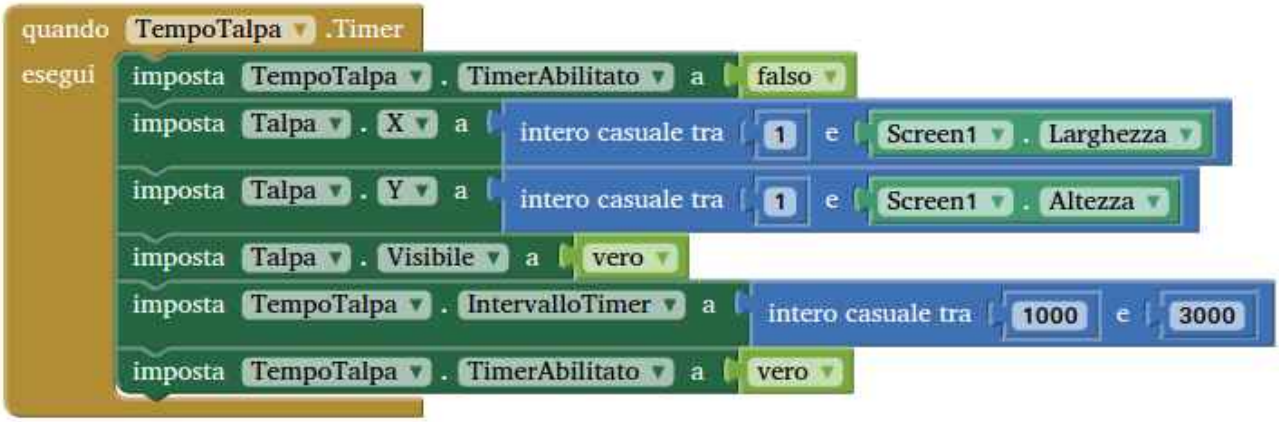


- nel tab **Multimediale** in basso a destra si caricano i file necessari al gioco: l'immagine della talpa ed il suono (che segnala che abbiamo beccato la talpa)
- Nel tab **Tavolozza** (Palette) a sinistra, nel gruppo 'Disegno e animazione' si trascina **Tela** (Canvas) in Screen1 (diventa Tela1); impostare le proprietà **Altezza** e **Larghezza** a 'Riempi contenitore' (Fill Parent) e il '**Colore di sfondo**' (Backgound color) a Arancione
- Dal gruppo 'Disegna e Animazione' trascinare un controllo **ImageSprite** dentro Tela1 e rinominarlo in **Talpa**; nel tab delle **Proprietà** del controllo impostare: **Larghezza** 78 pixel, **Altezza** 105 pixel; **Immagine**, selezionando il file della talpa caricato in precedenza.
- Dal tab **Sensori** inserire 2 oggetti di tipo **Orologio** (Clock), denominarli **TempoGioco** e **TempoTalpa**

Passare alla visualizzazione **Blocchi** (Blocks).

- A sinistra fare un click su Screen1 e scegliere il blocco '**Quando Screen1.Inizializza – Esegui**' ed inserire le **inizializzazioni** degli oggetti Talpa, TempoGioco e TempoTalpa come nell'immagine seguente. Queste istruzioni si possono trovare cliccando a sinistra sull'oggetto interessato e selezionando il blocco di interesse tra quelli disponibili.

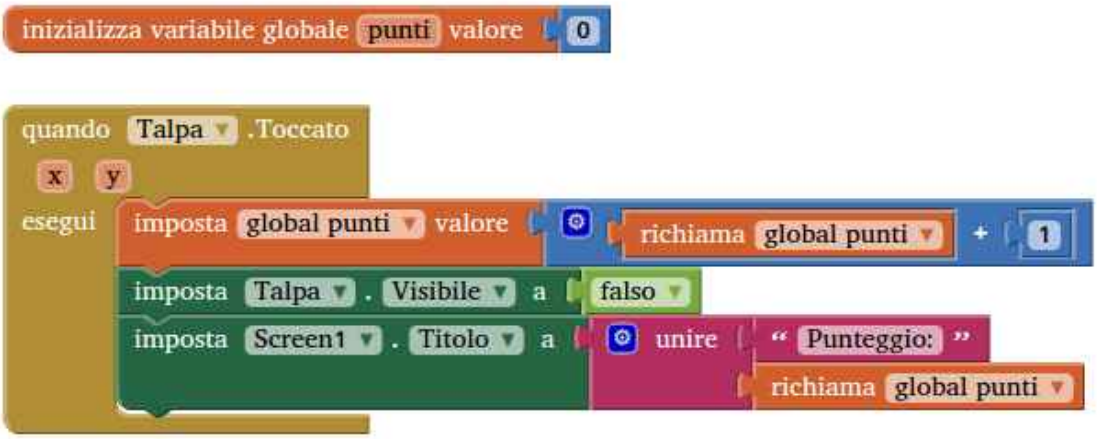
- A partire da un blocco **Quando TempoTalpa.Timer– Esegui**, che si può trovare dentro l'oggetto TempoTalpa a sinistra, si scrive il codice che serve per far apparire la talpa in modo casuale. Il codice contenuto nel blocco **Quando TempoTalpa.Timer– Esegui** viene eseguito ad intervalli di tempo pari al valore impostato per la proprietà **TempoTalpa.TimerInterval**.



- Salvare e generare una prima versione del progetto caricandola sullo smartphone o tablet Android: per fare questo selezionare dal menu **Compila** la voce **App (mostra QR code per apk)**. Quando verrà generato e mostrato il codice QR, **acquisirlo** con lo smartphone o tablet tramite l'apposita funzione della App **MIT AI2 Companion** scaricata ed installata all'inizio.

- Si prosegue gestendo il tempo di gioco: questo blocco di codice viene eseguito ad intervalli di tempo pari al valore impostato per la proprietà **TempoGioco.TimerInterval**, nel nostro caso una sola volta perchè serve a far terminare il gioco (istruzione **imposta TempoGioco.TimerEnabled a falso**)

- si aggiunge la gestione del "**tap**" sulla talpa e del **punteggio**

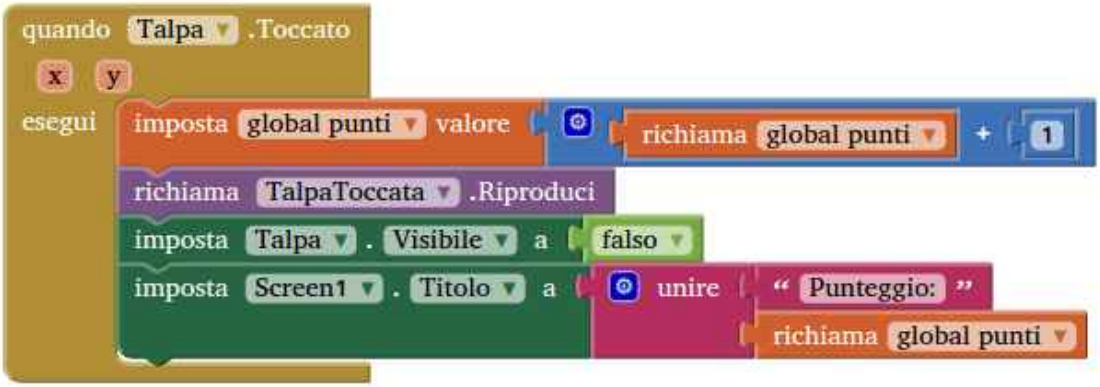


A questo punto il gioco è scaricabile e giocabile.

Aggiunta suono

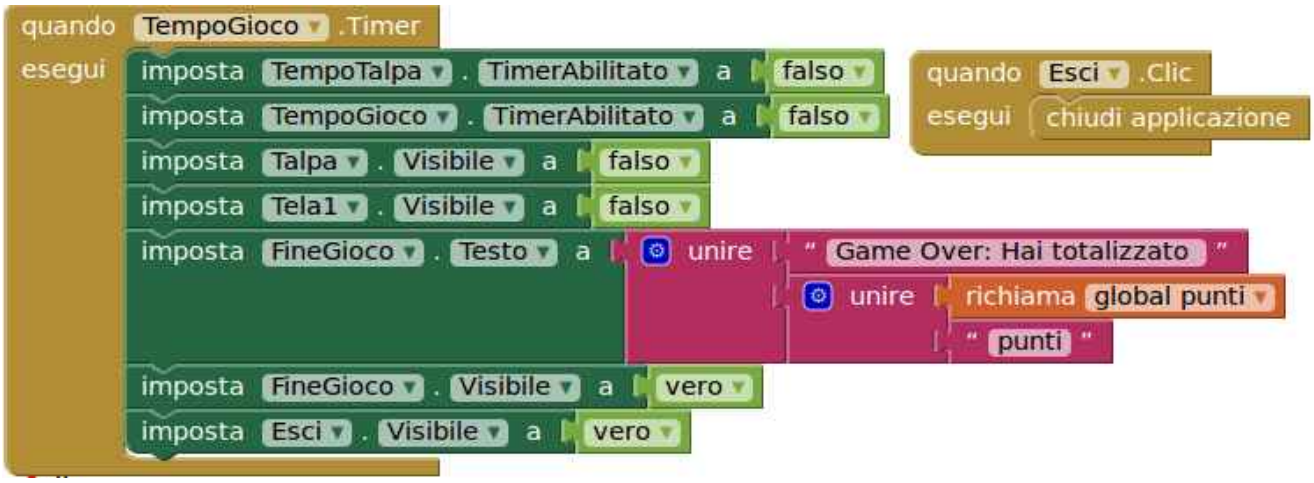
Per aggiungere un suono che segnali quando la talpa è stata acchiappata, si torna in visualizzazione **Designer** e si aggiunge un oggetto **Suono** dal gruppo '**Multimediale**' rinominandolo in '**TalpaToccata**'. Nelle proprietà di questo oggetto si imposta il campo **Sorgente** inserendovi il file del suono caricato inizialmente.

Si torna, quindi, alla visualizzazione **Blocchi** e si modifica il codice del tap della talpa come segue



Gestione Game Over

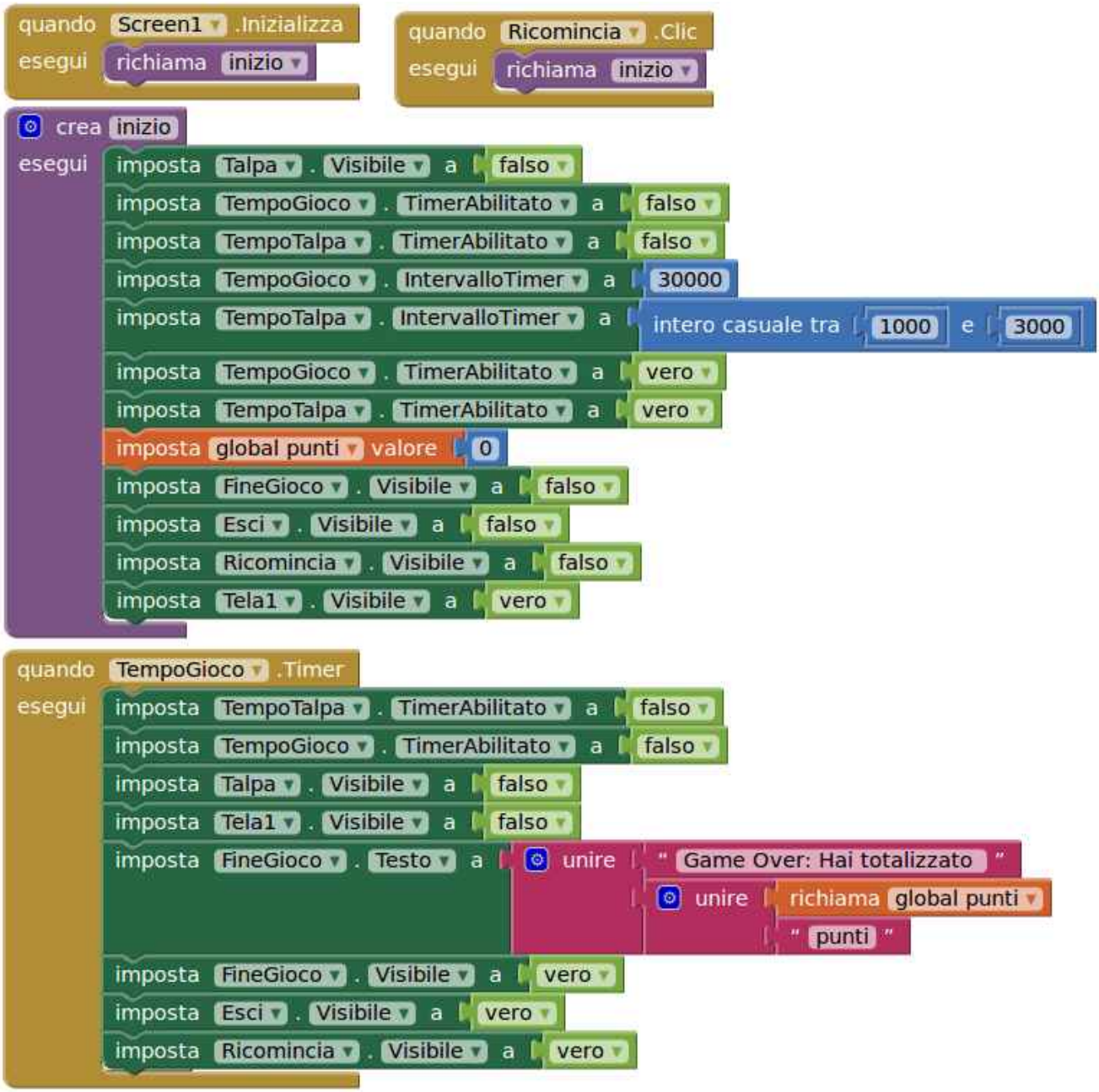
Al termine della partita si mostra una scritta: “Game Over: hai totalizzato XX punti”, e si mostra un bottone per uscire dall'applicazione.
Per la scritta dalla visualizzazione **Designer** si usa un oggetto di tipo **Etichetta** che portiamo sotto la **Tela**.
Rinominiamo l'oggetto in '**FineGioco**' e leviamo la spunta dalla proprietà **Visibile**, in modo che l'oggetto inizialmente non sia visibile.
Per il bottone fine gioco sempre nella modalità **Designer** si aggiunge un oggetto **Pulsante**, lo rinominiamo '**Esci**' e impostiamo la proprietà **Testo** a '**Esci**' e leviamo la spunta dalla proprietà **Visibile**, in modo che l'oggetto inizialmente non sia visibile.
Si passa alla modalità **Blocchi** e si lavora nel blocco di fine gioco, ovvero **TempoGioco.TimerInterval**. In questo blocco dobbiamo far sparire la **Tela**, comporre la scritta, farla apparire e far apparire il bottone **Esci**.
Va poi gestito il **Clic** sul bottone **Esci** con l'apposito blocco **quando Esci.Clic-Esegui** utilizzando l'istruzione **chiudi applicazione**.



È possibile aggiungere un secondo bottone chimandolo '**Ricomincia**'per far ricominciare il gioco, questo prevede di modificare molti più blocchi, perché le istruzioni del blocco **quando Screen1.inizializza** vanno richiamate anche a seguito del clic sul nuovo bottone, inoltre va gestita la sparizione dell'etichetta **FineGioco** e dei bottoni **Ricomincia** e **Esci.**

In pratica conviene spostare le istruzioni del blocco **quando Screen1.inizializza** in un blocco **procedura**, aggiungendo le istruzioni per gestire la visibilità (o meglio l'invisibilità di **FineGioco, Tela1, Esci** e **Ricomincia**).
Va poi inserito nel blocco **TempoGioco.TimerInterval** l'istruzione per far apparire il bottone **Ricomincia** al termine del gioco.

Ottengo dunque i blocchi seguenti:

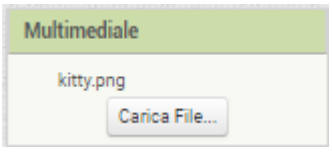


In questo tutorial andremo a creare un'app chiamata "HelloTom" un' immagine di un gatto che miagola quando viene toccata e "si arrabbia" quando si agita il cellulare o il tablet.

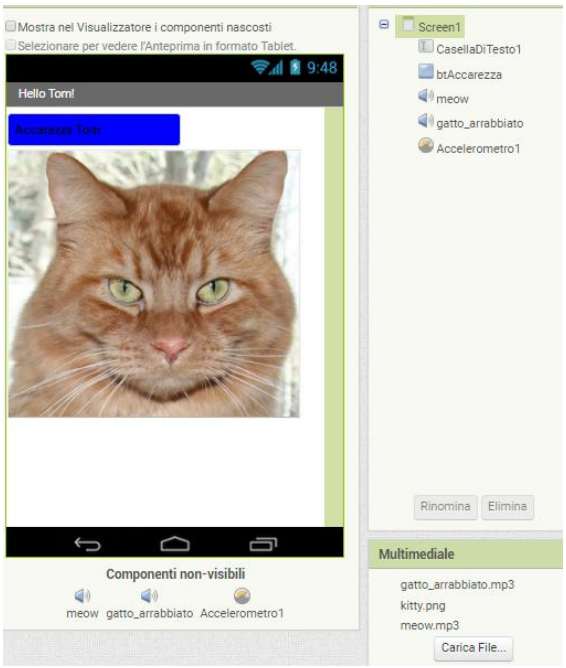
Iniziamo!

- Andiamo sul sito di AppInventor (http://appinventor.mit.edu/) ed accediamo con il nostro utente e la nostra password;
- Creiamo quindi un nuovo progetto dal menù "Progetti\Avvio Nuovo Progetto…" e chiamiamolo "HelloTom";
- Se lo si desidera impostiamo la lingua Italiano attraverso l'apposito menù in alto a destra;
- Nella sezione **Interfaccia Utente** trasciniamo l'oggetto **CasellaDiTesto**, quindi sulla destra, nella sezione proprietà, inseriamo "Accarezza Tom" nel campo **Testo**, mentre impostiamo **Blu** nel primo parametro: **ColoreSfondo**;
- Scaricare i tre files seguenti sul proprio pc:
  - Immagine di Tom: http://kata.coderdojo.it/archivio/10_AppInventor/HelloTom/kitty.png
  - Miagolio di Tom: http://kata.coderdojo.it/archivio/10_AppInventor/HelloTom/meow.mp3
  - Tom Arrabbiato: http://kata.coderdojo.it/archivio/10_AppInventor/HelloTom/gatto_arrabbiato.mp3

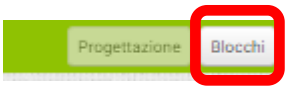
- Aggiungiamo ora un **Pulsante** trascinandolo sotto la **CasellaDiTesto**. Nelle proprietà del pulsante, posizionarsi alla voce **Immagine** (dove c'è la scritta Nessuna…); apparirà una finestra che ci permetterà di caricare l'immagine di Tom scaricata in precedenza. Premete **Carica File** e andate a trovare l'immagine, quindi OK per confermare il caricamento. Dopo pochi secondi apparirà l'immagine desiderata. Rimuoviamo ora il valore nel campo **Testo**.
- **SALVIAMO!**
- Aggiungiamo ora i due files audio al progetto; lo faremo in due passaggi:
  - Caricamento del file nel progetto: nella sezione **Multimedia** (in basso a destra) premere il pulsante **Carica File…**; procedere quindi con il caricamento dei due mp3 scaricati in precedenza;



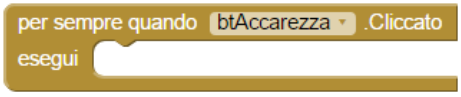
  - Associare i due files ai corrispettivi oggetti **Suono**: per fare questo sarà sufficiente trascinare nella nostra applicazione due oggetti suono che si trovano nella sezione **Componenti Disponibili** (sulla sinistra), **Multimediali**; assegniamo loro un .mp3 ciascuno e rinominiamoli in base al file associato.
- Infine aggiungiamo l'ultimo elemento, l' **Accelerometro**, che si trova all'interno dei **Sensori,** semplicemente trascinandolo sulla nostra App;
- Al termine dell'operazione dovremo avere una situazione molto simile a questa:



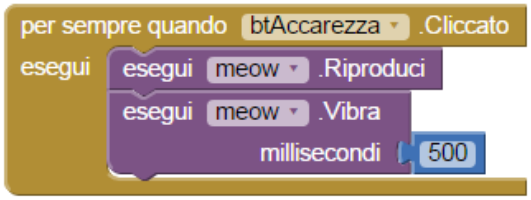
- **SALVIAMO!**
- La parte di "design" è terminata, andiamo quindi nella sezione **Blocchi** premendo l'apposito pulsante in alto sulla destra.



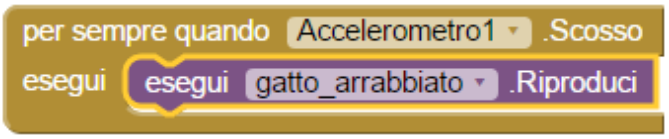
- La prima cosa che vogliamo fare è far miagolare Tom se lo accarezziamo premendo il pulsante **"Accarezza Tom"**;
- Selezioniamo quindi "btAccarezza", nella lista a sinistra, e trasciniamo il comando:



- Poi, sempre dalla lista dei componenti a sinistra andiamo a selezionare il suono **meow**, per poi trascinare all'interno di "**per sempre btAccarezza.Cliccando**" la proprietà **Riproduci**, e la proprietà **Vibra** alla quale andrà aggiunto inoltre il valore in millisecondi (500) grazie al primo mattoncino che troviamo nei blocchi **Matematica**. A fianco il risultato che avremo al termine.

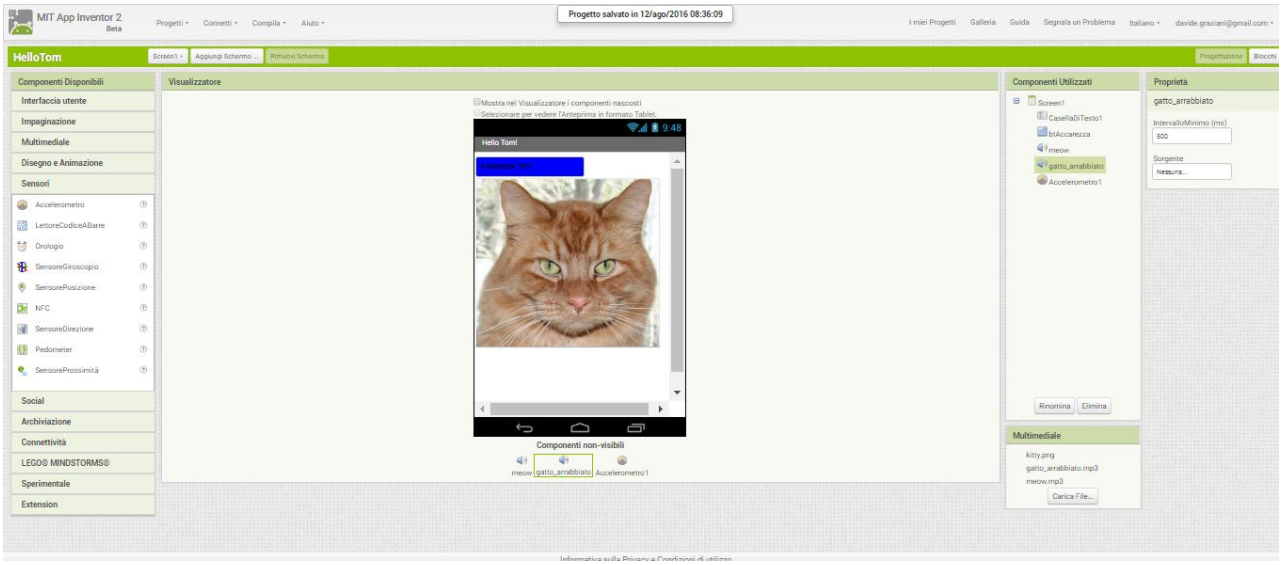


- **SALVIAMO!**
- Facciamo ora arrabbiare Tom! Andiamo sul componente **Accelerometro1** e trasciniamo all'interno della nostra area di lavoro "**per sempre quando Accelerometro1.Scosso**"; poi, spostandoci su **gatto_arrabbiato**, andremo a posizionare la proprietà "**esegui gatto_arrabbiato.Riproduci**". Al termine, il risultato che avremo sarà:

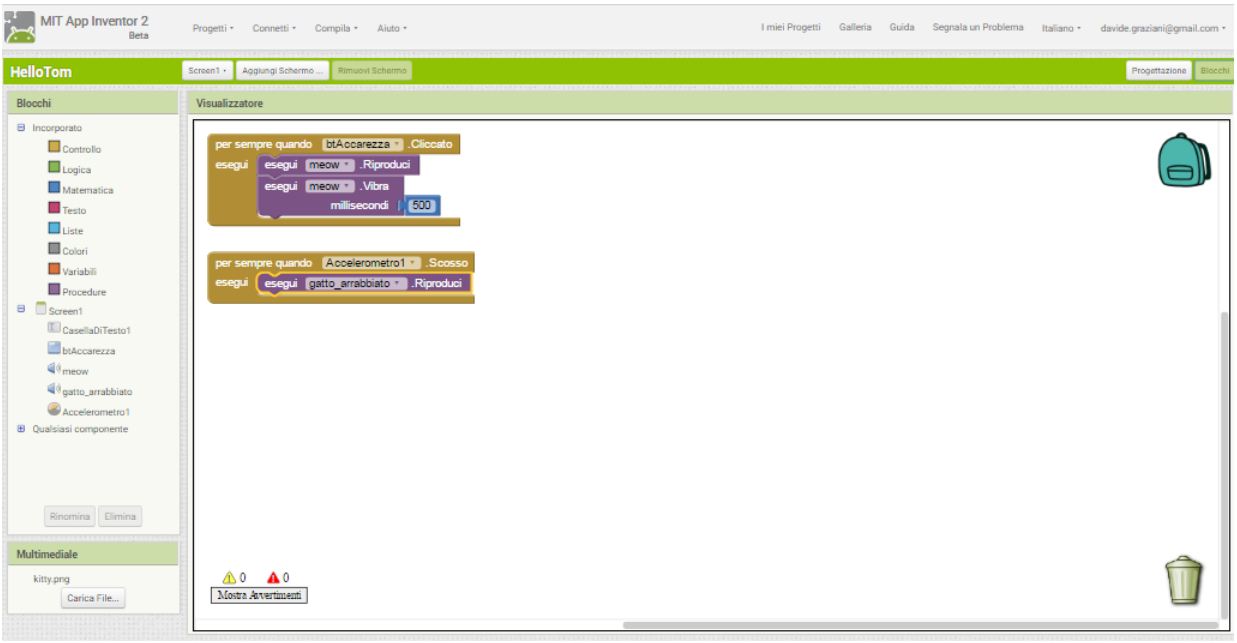


- Finito! Non resta che compilarlo e poi scaricarlo sul nostro cellulare o tablet.

Per riepilogare, nella sezione Designer dovremmo avere una situazione molto simile alla seguente:



Mentre nella sezione Blocchi

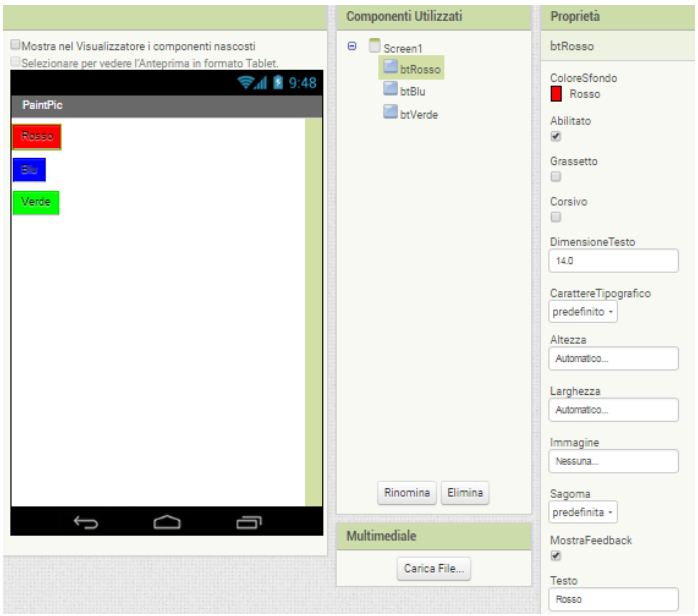


…BE COOL!!!!!

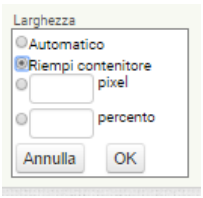
In questo tutorial andremo a creare un'app che ci permetterà di "disegnare" su una immagine oppure su una fotografia, anche appena scattata dal nostro tablet o smartphone.

Iniziamo!

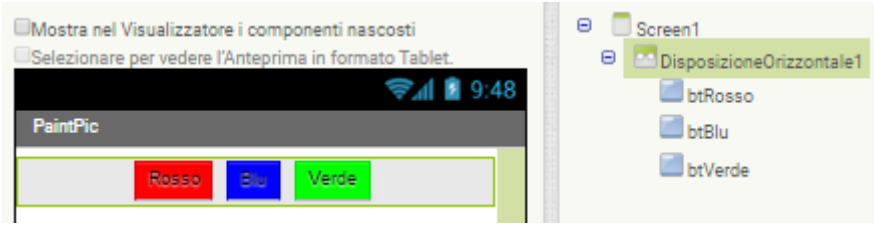
- Andiamo sul sito di AppInventor (http://appinventor.mit.edu/) ed accediamo con il nostro utente e la nostra password;
- Creiamo quindi un nuovo progetto dal menù "Progetti\Avvio Nuovo Progetto…" e chiamiamolo "PiantPic";
- Se lo si desidera impostiamo la lingua Italiano attraverso l'apposito menù in alto a destra;
- Carichiamo due nuovi files multimediale nel progetto: nella sezione **Multimedia** (in basso a destra) premiamo il pulsante **Carica File…** poi con **Scegli file** selezioniamo il file kitty.png (che potete trovare anche all'indirizzo http://kata.coderdojo.it/archivio/10_AppInventor/PaintPic/kitty.png) infine **OK** per confermare il caricamento;
- Ripetiamo quanto sopra anche per iconaPaintPic.png (http://kata.coderdojo.it/archivio/10_AppInventor/PaintPic/iconaPaintPic.png)
- Nelle proprietà di **Screen1** inseriamo inoltre "PaintPic" nel campo **Titolo** e in **Icona** selezioniamo iconaPiantPic dal menù a tendina;
- **SALVIAMO!**
- Trasciniamo un oggetto **Pulsante** dalla sezione **Interfaccia Utente** a sinistra;
- Rinominiamolo "btRosso";
- Modifichiamo, a destra, le proprietà seguenti:
  - **ColoreSfondo**: selezioniamo il colore **rosso** dal menù a tendina;
  - **Testo:** Rosso;
- Trasciniamo ora altri due pulsanti ripetendo le modifiche fatte per il pulsante rosso, ma uno sarà **blu** l'altro invece **verde**;
- Al termine dovremo avere una situazione simile alla seguente:



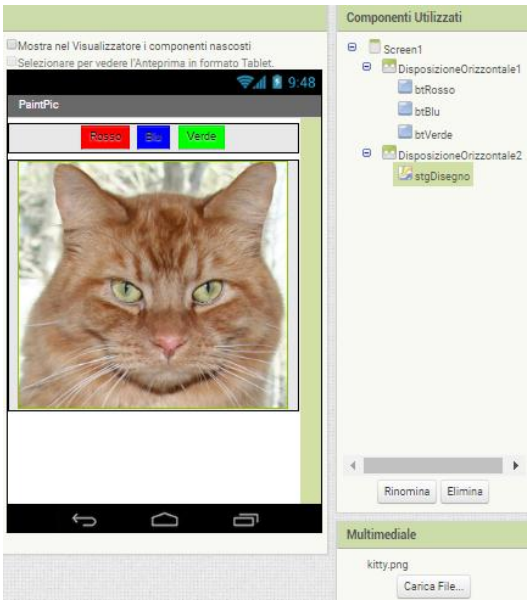
- Mettiamoli ora in ordine, uno accanto all'altro; per fare questo inseriamo l'oggetto **DisposizioneOrizzontale** che si trova all'interno della sezione **Impaginazione,** a sinistra.



- modifichiamo la proprietà **Larghezza,** impostando **Riempi contenitore** e confermando con **OK** , e la proprietà **AllineamentoOrizzontale**, selezionando **Centro**, confermando anche in questo caso con **OK.**
- Trasciniamo i pulsante creati in precedenza (btRosso, btBlu, btVerde) al suo interno, così da avere:

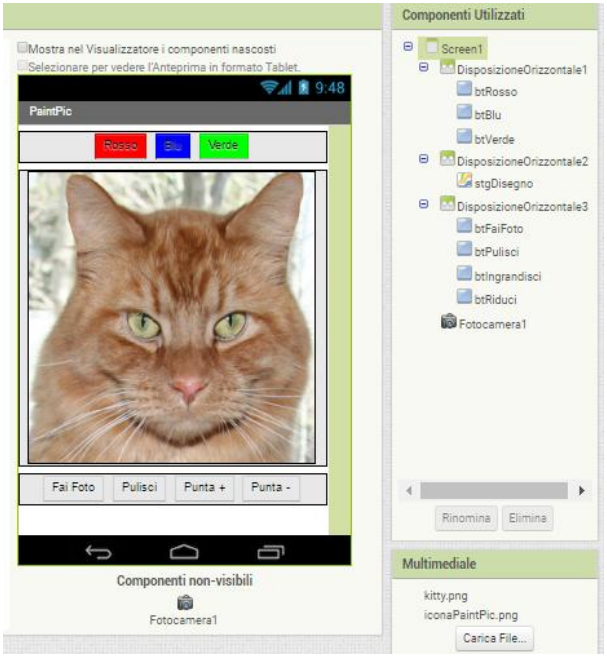


- **SALVIAMO!**
- Aggiungiamo ora una nuova Disposizione orizzontale (che dovrebbe prendere il nome di **DisposizioneOrizzontale2**) e modifichiamo le proprietà come abbiamo fatto in precedenza:
  - **AllineamentoOrizzontale**: Centro;
  - **Larghezza**: Riempi Contenitore;
- Sulla sinistra, nella sezione **Disegno e Animazione** trasciniamo l'oggetto **Stage** all'interno di **DisposizioneOrizzontale2** e rinominiamolo **stgDisegno**;
- Modifichiamo lo seguenti proprietà:
  - **ImmagineSfondo**: segliamo il file multimediale kitty.png;
  - **Altezza**: 300 pixel;
  - **SpessoreLinea**: 2;
  - **ColoreDisegno**: rosso;
  - **AllineamentoTesto**: Centro;
- Al termine dovremmo avere una situazione come quella sotto:

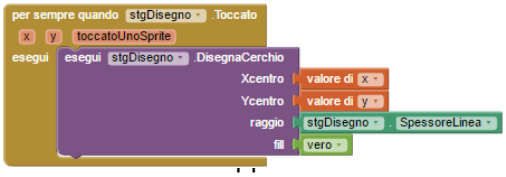


- **SALVIAMO!**
- Dopo aver impostato colori ed immagine, andiamo ad aggiungere i pulsanti "funzione" per la nostra app.

- Inseriamo un nuovo oggetto **DisposizioneOrizzontale** (DisposizioneOrizzontale3) ed impostiamo le proprietà:
    - **AllineamentoOrizzontale**: Centro;
    - **Larghezza**: Riempi Contenitore;
- Trasciniamo per 4 volte l'oggetto **Pulsante** all'interno di DisposizioneOrizzontale3, rinominandoli rispettivamente
    - btFaiFoto (ed imposta la proprietà Testo in: "Fai Foto");
    - btPulisci (ed imposta la proprietà Testo in: "Pulisci");
    - btIngrandisci (ed imposta la proprietà Testo in: "Punta +");
    - btRiduci (ed imposta la proprietà Testo in: "Punta -");
- Come ultima operazione andremo ad aggiungere l'oggetto **Fotocamera** che troverete nella sezione **Multimediale** (sulla sinistra) semplicemente trascinandola all'interno della nostra area di lavoro.
- Bene! Le attività sulla parte **Progettazione** sono terminate, dovreste avere così una situazione come quella sotto:



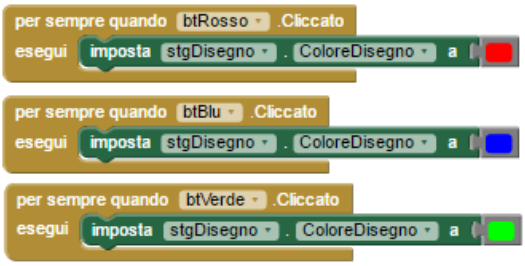
- Andiamo ora nella sezione **Blocchi**, ma non prima di aver fatto un bel **SALVATAGGIO!**
- Troviamo l'oggetto stgDisegno e trasciniamo il comando "**per sempre quando stgDisegno.Toccato…**" nell'area di lavoro. **X** e **Y** rappresentano le coordinate dove appoggiamo il dito su stgDisegno, mentre **toccatoUnoSprite** identifica se viene toccato un particolare **Sprite** (non verrà utilizzato in questo tutorial;
- All'interno del blocco appena inserito posizioniamo "**esegui stgDisegno.DisegnaCerchio**" (che si trova sempre all'interno dell'oggetto stdDisegno).
- Popoliamo ora i valori X e Y. Per fare ciò basta lasciare il puntatore del mouse per qualche secondo prima su X e poi su Y e trasciniamo "**valore di X**" su **Xcentro** e "**valore di Y**" su **Ycentro**.
- Inseriamo il valore di **stgDisegno.SpessoreLinea** sulla proprietà raggio che abbiamo impostato a 2 precedentemente;
- Dovremo così avere un blocco come il seguente:
- **SALVIAMO!**



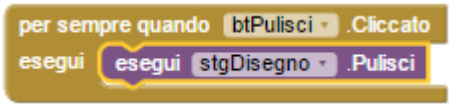
- Il comando DisegnaCerchio ci permette di disegnare un punto sullo schermo, ma come fare per disegnare una riga o un arco quando trasciniamo il dito sullo schermo?
- Per fare questa operazione utilizzeremo il "comportamento" "**per sempre quando stdDisegno.Trascinamento**" assieme al comando "**esegui stdDisegno.DisegnaLinea**"

- Per disegnare una linea abbiamo bisogno di due punti fondamentalmente: il punto da dove si parte, nel nostro cosa **x1** e **y1**, e il punto dove arriviamo **x2** e **y2**. I valori che ci interessano saranno Xprec e Yprec (perché non utilizzeremo Xiniziale e Yinizale? Provate ad utilizzarli!) per x1 e y1, Xattuale e Yattuale per x2 e y2. Laciando il puntatore del mouse per qualche istante su ciascuno, apparirà un pop-up dal quale selezioneremo "valore di …";
- Al termine si avrà un blocco come il seguente:

- **SALVIAMO!**
- Il prossimo passo sarà quello di poter cambiare colore per disegnare…vi ricordate i pulsanti **rosso**, **blu** e **verde**? Bene!
- Iniziamo con il **rosso**, andiamo **btRosso** e trasciniamo nella nostra area di lavoro "**per sempre quando btRosso.Cliccato**";
- All'interno trascinate "**imposta stdDisegno.ColoreDisegno a**" che trovate in **stgDisegno**, infine aggiungete il colore **rosso** proprio all'interno della sezione **Colori;**
- Ripetete le operazioni precedenti anche per **blu** e **verde** in modo da ottenere il risultato qui accanto;



- Trasciniamo ora, dall'oggetto **btPulisci**, "**per sempre quando btPulisci.Cliccato**" sull'area di lavoro e inserite al suo interno "**esegui stdDisegno.pulisci**";



- **SALVIAMO!**
- Concentriamoci ora sul pulsante "Fai Foto" e dall'oggetto **btFaiFoto** sposta nell'area di lavoro "**per sempre quando btFaiFoto.Cliccato**", poi aggiungi l'azione "**esegui Fotocamera1.ScattaFoto**" che trovi tra le funzioni di **Fotocamera1**;

- Dopo aver scattato la foto, la vogliamo utilizzare per disegnarci sopra impostandola come sfondo del nostro **stgDisegno**: quindi aggiungere "**per sempre quando Fotocamera1.ScattataFoto**" ed all'interno trascinare "**imposta stgDisegno.ImmagineSfondo a**"; per

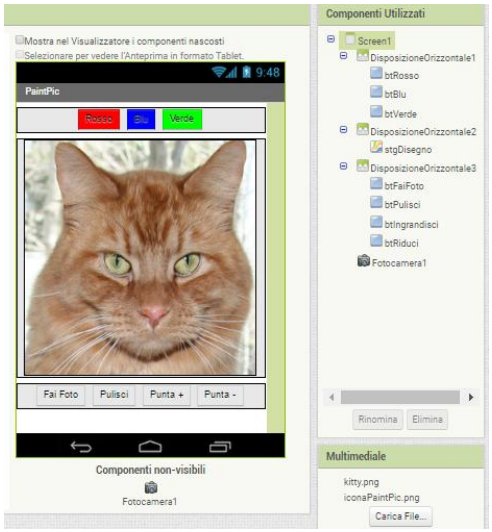
concludere lasciare il mouse qualche istante su “**Immagine**” e dal pop-up trascinare “**valore di immagine**”

- Non manca che configurare i due pulsanti per ingrandire o rimpicciolire la punta per disegnare.
- Da btRiduci aggiungere sull’area di lavoro “**per sempre quando btRiduci.Cliccato esegui**”;
- Da **Variabili** aggiungi “**porta global raggio a valore**” e da **Matematica** selezionare il primo blocco e impostare 2;

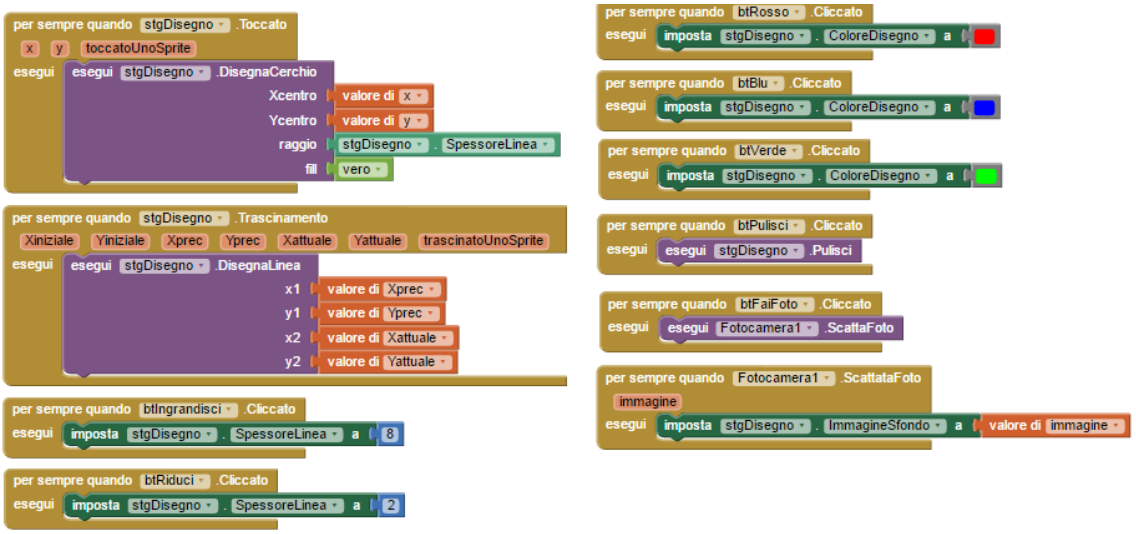
- Da btAumenta aggiungere sull’area di lavoro “**per sempre quando btRiduci.Cliccato esegui**”;
- Da **Variabili** aggiungi “**porta global raggio a valore**” e da **Matematica** selezionare il primo blocco e impostare 8;

- **SALVIAMO!**
- Finito! Non resta che compilarlo e poi scaricarlo sul nostro cellulare o tablet.

Per riepilogare, nella sezione Designer dovremmo avere una situazione molto simile alla seguente:



Mentre nella sezione Blocchi



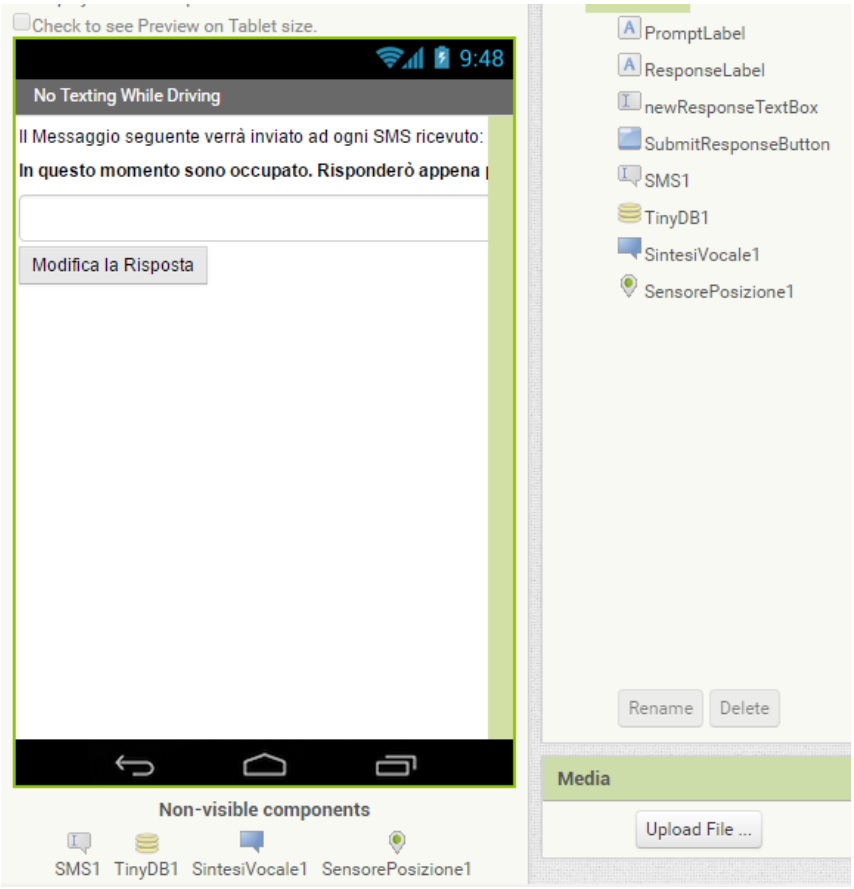
In questo tutorial andremo a creare un'app che risponderà per noi agli sms che ci arriveranno e sarà in grado di leggerne il contenuto (richiesto un cellulare).

Iniziamo!

- Andiamo sul sito di AppInventor (http://appinventor.mit.edu/) accedendoci con il nostro utente e la nostra password;
- Creiamo quindi un nuovo progetto dal menù "Progetti\Avvio Nuovo Progetto…" e chiamiamolo "SegreteriaSMS";
- Se lo si desidera, impostiamo la lingua Italiano attraverso l'apposito menù in alto a destra;
- Inseriamo quindi nell'interfaccia i seguenti componenti:

Componente	Gruppo	Nome	Scopo
Etichetta	Interfaccia Utente	PromptLabel	Indica all'utente come funziona l'app
Etichetta	Interfaccia Utente	ResponseLabel	E' la risposta che verrà inviata a chi vi spedisce un SMS
CasellaDiTesto	Interfaccia Utente	newResponseTextBox	L'utente potrà personalizzare la propria risposta
Pulsante	Interfaccia Utente	SubmitResponseButton	L'utente lo seleziona per confermare la risposta
SMS	Social	SMS1	Elabora l'SMS
TinyDB	Archiviazione	TinyDB1	Salva la risposta in un database
SintesiVocale	Multimedia	SintesiVocale1	Legge il testo dell'SMS
SensorePosizione	Sensori	SensorePosizione1	Indica dove è il telefono

- Quindi modificare le proprietà come segue:
  - PromptLabel: nel campo **Testo** inserire il testo "Il Messaggio seguente verrà inviato ad ogni SMS ricevuto:";
  - ResponseLabel: nel campo **Testo** inserire "In questo momento sono occupato. Risponderò appena possibile.", poi mettere il **Grassetto**;
  - newResponseTextBox: modificare il campo **Larghezza** selezionando "Riempi contenitore";
  - SubmitResponseButton: modificare il campo **Testo** inserendo "Modifica la Risposta".
- Al termine dovremmo avere una situazione simile alla seguente:



- La prima parte, quella di design, è terminata! Andiamo a modificare il comportamento degli oggetti che abbiamo inserito;
- Il nostro primo obiettivo è quello di rispondere all'SMS ricevuto; raggiungiamo quindi SMS1 e selezioniamo "per sempre quando SMS1.Messaggio ricevuto";
- Quindi inseriamo al suo interno:
    - "imposta SMS1.NumeroTelefonico a " e, lasciando il mouse per qualche secondo su "numero" apparirà un pop-up dove potrete selezionare "valore di numero" da collegare al comando precedente;
    - Dopo il numero telefonico inseriamo il messaggio inserendo "imposta SMS1.Messaggio a" e, in "ResponseLabel", andare ad inserire la funzione "ResponseLable.Testo";
    - Infine, da SMS1, trasciniamo il comando "esegui SMS1.InviaMessaggio";

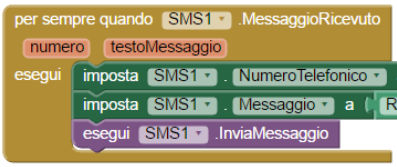
- **Salviamo!**
- A questo punto, una volta ricevuto un SMS, la nostra app sarà in grado di rispondere al mittente con un nostro messaggio;
- Adesso vogliamo anche aggiungere al messaggio la nostra posizione;
- Prima di tutto "inizializziamo", cioè mettiamo un valore iniziale, una variabile che chiameremo "lastKnowLocation";

- Andiamo quindi su “Variabili” e trasciniamo nella nostra area di lavoro il comando “inizializza variabile globale lastKnowLocation con valore” al quale collegheremo un valore di testo “Sconociuta”;

- Procediamo in modo che possa contenere la nostra posizione andando su **SensorePosizione1** e scegliendo “per sempre quando SensorePosizione1.PosizioneCambiata”;
- Da Variabili, inseriamo al suo interno il comando “porta global LasKnowLocation a valore”, attaccandoci “SensorePosizione1.IndirizzoAttale” che troviamo tra le opzioni di SensorePosizione1;
- Al termine, il nostro nuovo blocco si presenterà così:

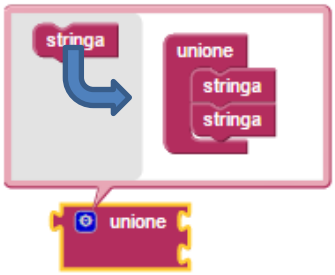
- Ottimo! Non ci resta che aggiungere il valore di lastKnownLocation all’SMS di risposta…ma prima…
- **Salviamo!**
- Riprendendo il blocco dove diciamo alla nostra app di inviare solo il testo e aggiungiamo anche la posizione;

PRIMA



DOPO

- Cosa abbiamo aggiunto?
- Prima di tutto il comando “unione” che troviamo all’interno di Testo;
- Attaccandolo a “imposta SMS1.Messaggio a “ vi accorgerete vi permetterà di unire solo due testi, per permettere di aggiungere tre o più sarà sufficiente cliccare su  e trascinare “stringa” all’interno di unione quante volte si desidera, nel nostro caso occorrerà aggiungerne solo una in più;



- Nella prima posizione rimettiamo “ResponseLabel.Testo”;
- Al centro, da “Testo”, aggiungiamo un testo vuoto nel quale scriviamo “La mia ultima posizione è: “
- Come ultimo, da “Variabili”, inseriamo “valore di global lastKnownLocation”;
- **Salviamo!**
- Ora aggiungiamo la “Voce” alla nostra app, ovvero, ogni qual volta ci arriva un messaggio, oltre a rispondere con un altro messaggio vogliamo che ci venga letto il numero di chi lo ha spedito e il contenuto dell’sms stesso.

- Andremo quindi a modificare il blocco “per sempre quando SMS1.MessaggioRicevuto” come segue:

PRIMA

DOPO

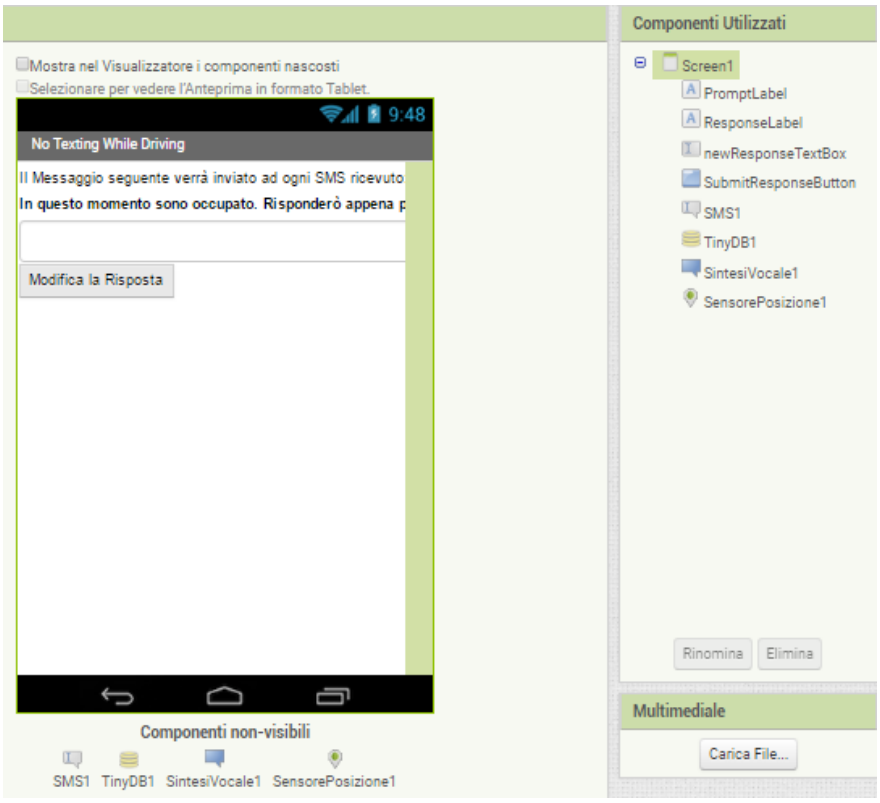
- Cosa è cambiato? In sostanza dopo “esegui SMS1.InviaMessaggio” abbiamo aggiunto un nuovo comando “esegui SintesiVocale1.PronunciaTesto” che trovate tra le opzioni disponibili in **SintesiVocale1;**
- Quindi abbiamo inserito il comando “unione” all’interno di **Testi**, impostandolo affinché potesse contenere 4 oggetti così come abbiamo fatto precedentemente con il messaggio SMS di risposta;
- Infine abbiamo messo alla prima posizione un oggetto “stringa di testo” (in **Testi**) con la frase “SMS ricevuto da”; alla seconda il “valore di numero”; alla terza un’altra “stringa di testo” con “il messaggio è”; infine il “valore di testoMessaggio”;
- Se non ricordate come recuperare i valori di “numero” e “testoMessaggio” andate a vedere alla pagina 2;
- Ricapitolando, in questo momento la nostra app, quando riceviamo un messaggio, è in grado di rispondere all’SMS con un altro che conterrà la nostra risposta e la nostra ultima posizione, inoltre saremo in grado di sentire il messaggio letto proprio dalla nostra app;
- **Salviamo!**
- Se volessimo personalizzare il messaggio di risposta? All’inizio avevamo impostato “In questo momento sono occupato. Risponderò appena possibile.”
- Per far sì che questo sia possibile bisogna salvare il nuovo messaggio da qualche parte così che, ogni volta che aprirò la mia app, il messaggio sia quello desiderato ed impostato l’ultima volta;
- Procediamo quindi inserendo sulla nostra area di lavoro, da **SubmitResponseButton**, “per sempre quando SubmitResponseButton.Cliccato”;
- Al suo interno spostiamo il contenuto di newResponseTextBox in ResponseLabel con il comando “imposta ResponseLabel.testo a” (in **ResponseLabel**) al quale attaccheremo “newResponseTextBox.testo” (in **newResponseTextBox**);
- Andiamo ora a cancellare il valore all’interno di **newResponseTextBox** aggiungendo “imposta newResponseTextBox.Testo a “ al quale verrà aggiunto, da Testo, il valore “vuoto” (primo elemento);
- Per fare in modo che il nuovo testo sia disponibile anche le volte successive continuiamo utilizzando “esegui TinyDB1.MemorizzaValore” (**che si trova in TinyDB1**);
- Ad “etichetta” inseriamo un Testo “responseMessage”, mentre in valoreDaMemorizzare “ResponseLabel.Testo”

- Il nuovo blocco di codice dovrebbe essere come sotto:

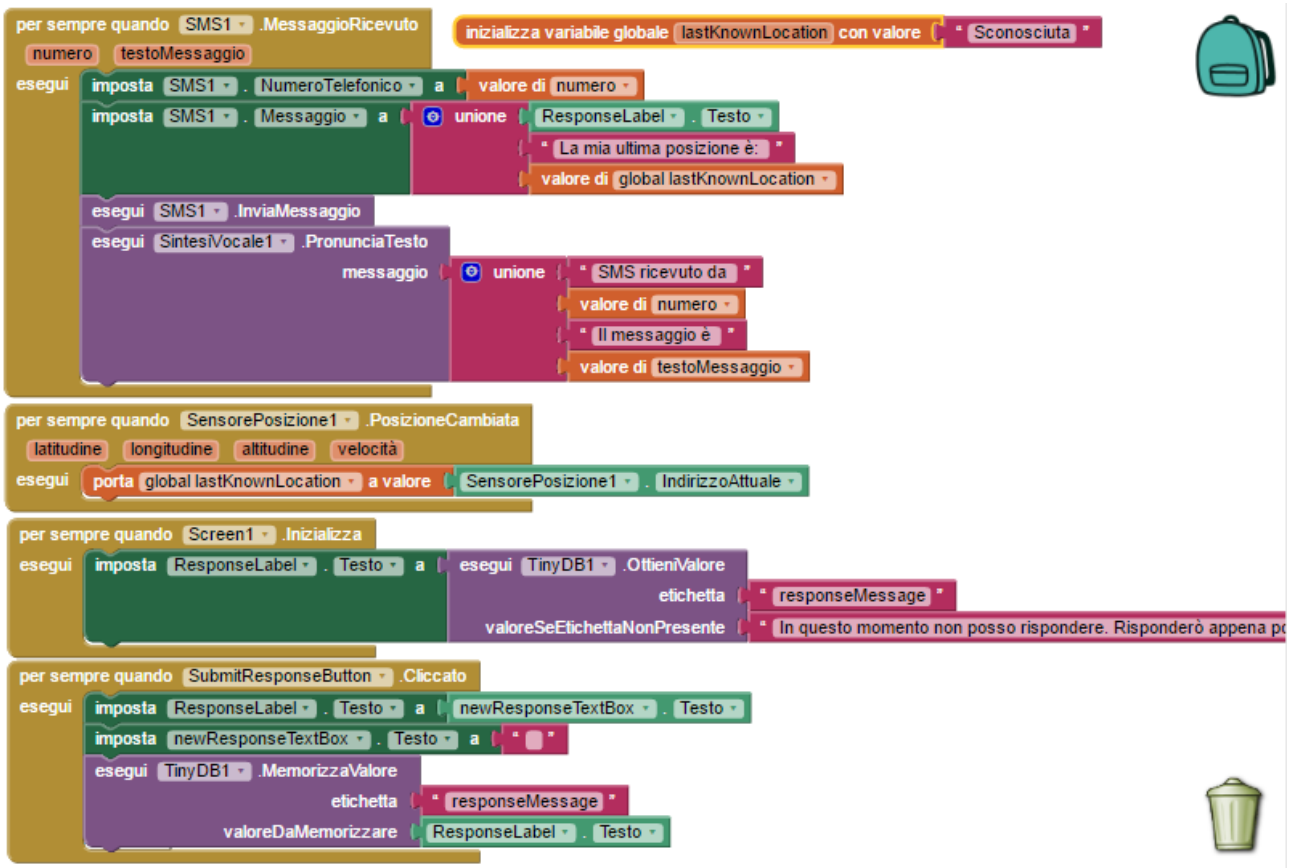
- **Salviamo!**
- Ultimo, ma non meno importante, occorre che all'avvio della nostra applicazione il messaggio che abbiamo personalizzato sia ricaricato, procediamo;
- Da **Screen1** trascinate "per sempre quando Screen1.Inizializza";
- All'interno da **ResponseLabel** "imposta ResponseLabel.Testo a" e a seguire "esegui TinyDB1.OttieniValore" (all'interno di **TinyDB1**); in "etichetta" inserire "responseMessagge" (ovvero l'etichetta del valore che abbiamo salvato nel blocco precedente), mentre in "valoreSeEtichettaNonPresente" imposteremo il valore di default "In questo momento non posso rispondere. Risponderò appena possibile.";
- Il blocco completo sarà:

- **Salviamo un'ultima volta!**
- Finito, non resta che compilarlo e poi scaricarlo sul nostro cellulare

Per riepilogare, nella sezione Designer dovremmo avere una situazione molto simile alla seguente:

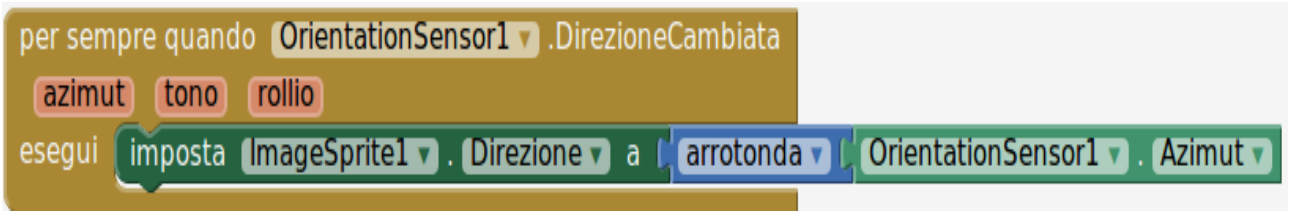


Mentre nella sezione Blocchi

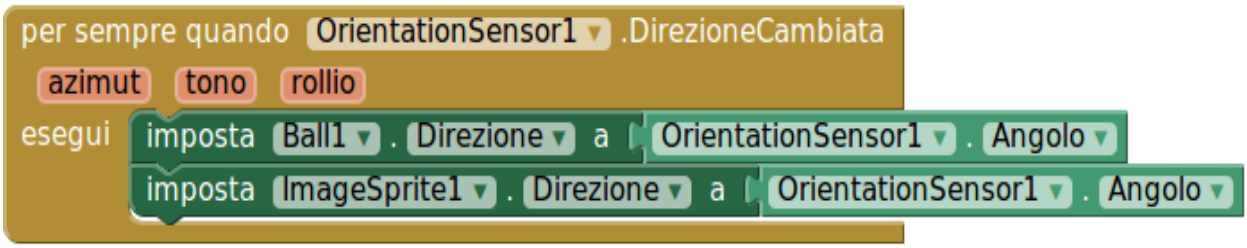


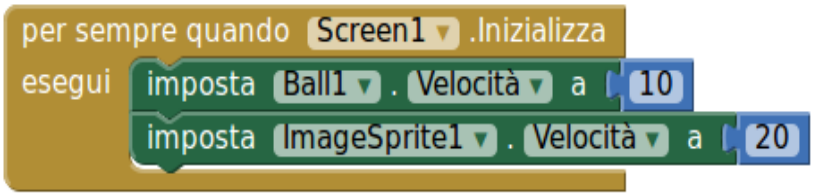
…BE COOL!!!!!

Bussola

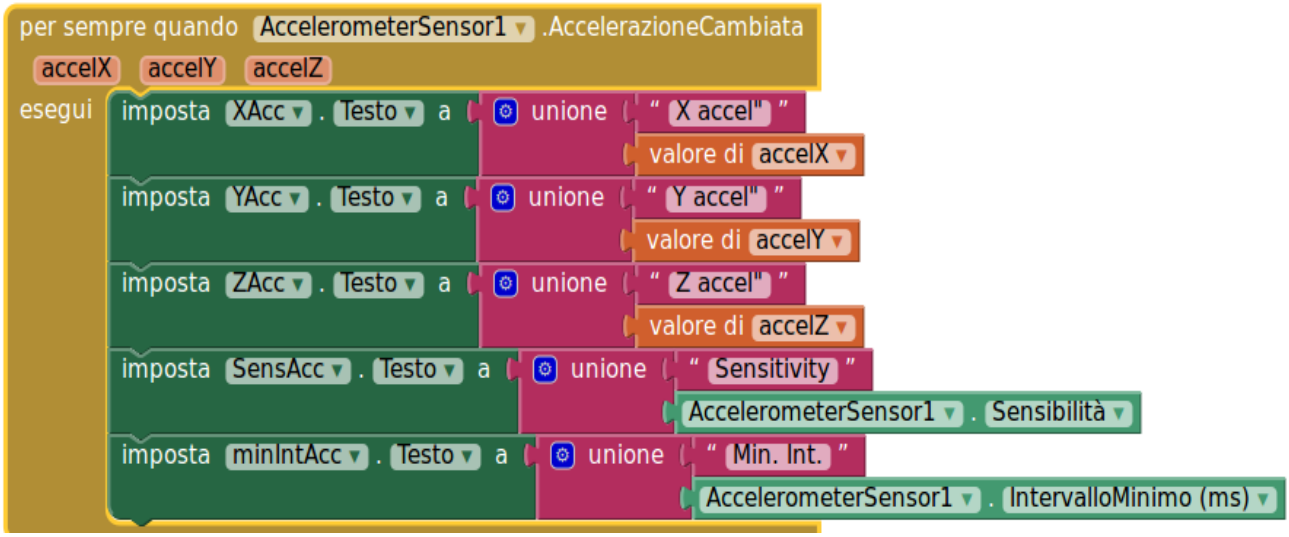


Movimento palla/sprite con sensore di orientamento

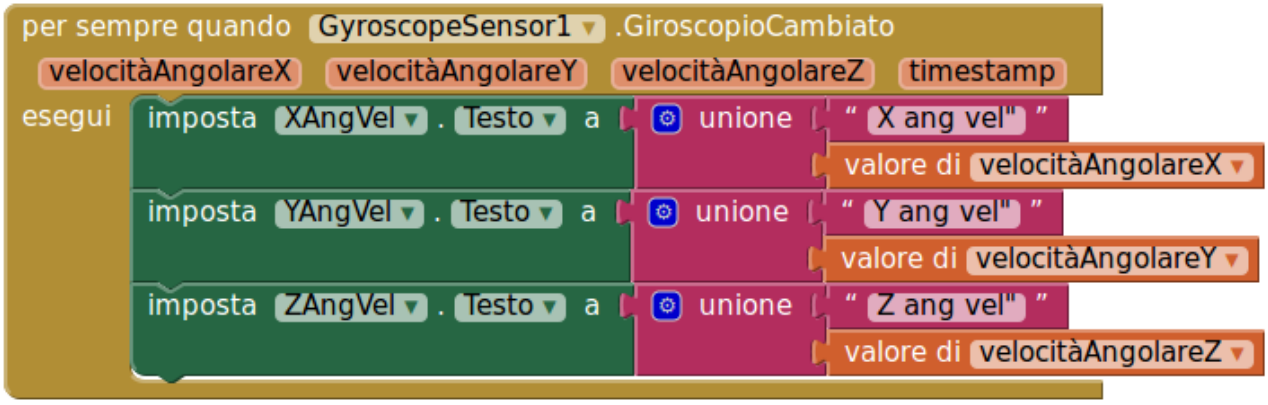




Accelerometro



Giroscopio



Sensore di prossimità