Formation AppInventor

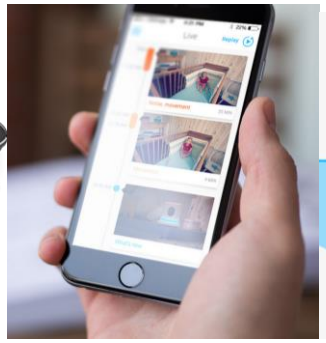
Journée du 22 mai 2015

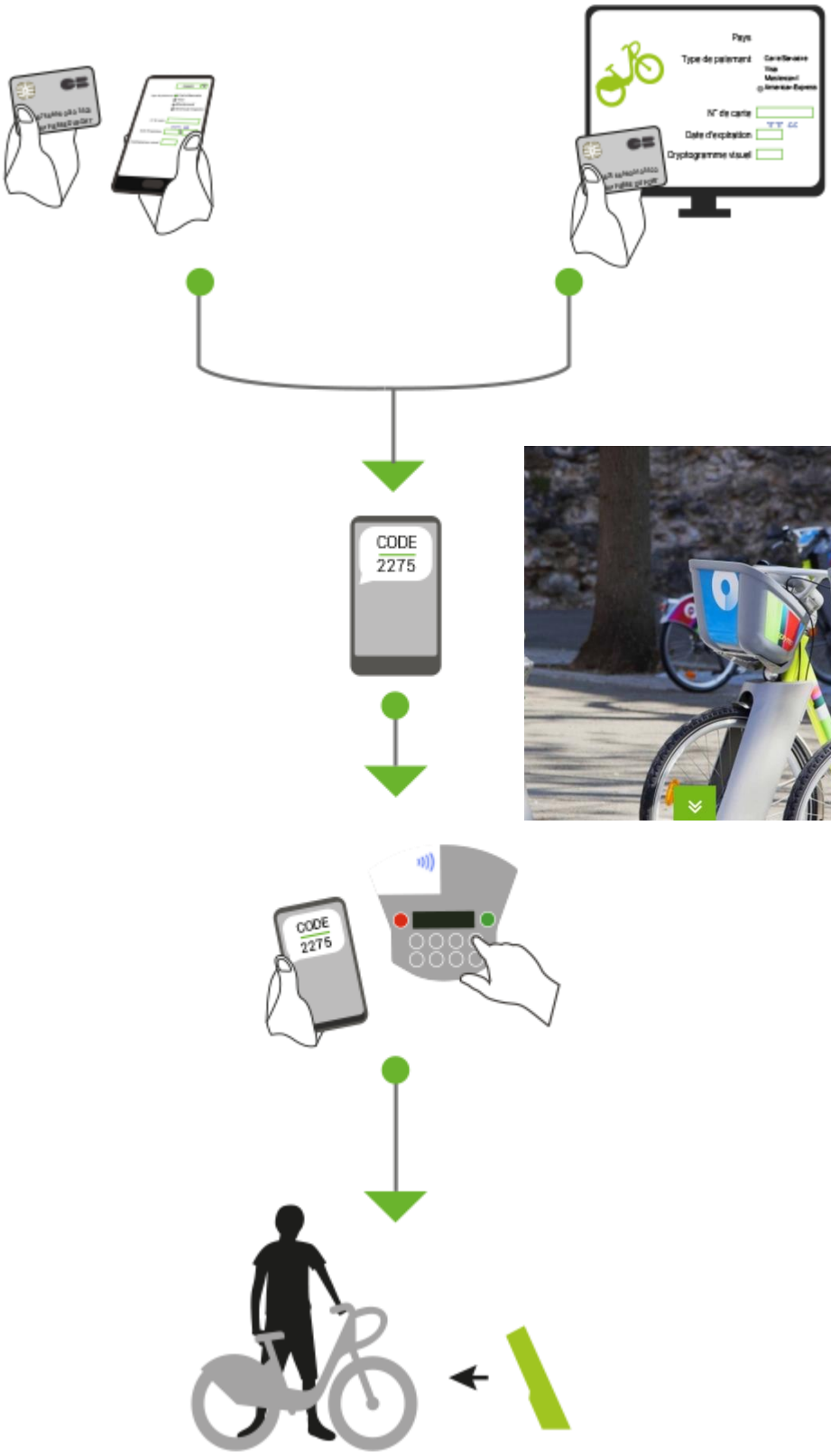
1 Présentation et contexte

L'environnement des élèves présente de plus en plus de systèmes qui intègrent des systèmes numériques.

Les smartphones sont des objets techniques très présents dans leur quotidien. Les usages associés à un smartphone s'étoffent régulièrement.

Communiquer à distance (phonie, texte sms, vidéo), agenda, prise de notes, paiement sans contact, etc…




Pays
Type de paiement
N° de carte
Date d'expiration
Cryptogramme visuel
CODE
2275
CODE
2275

2 Contexte des nouveaux programmes de collège

2.1 Projet de programme en mathématiques

Thème E – ALGORITHMIQUE ET PROGRAMMATION		
Attendus de fin de cycle		
Analyser un problème complexe, définir des sous-problèmes, des étapes de résolution Reconnaître des configurations récurrentes, mettre en évidence des interactions Traduire un algorithme dans un langage de programmation		
Repères pour la construction de l'attendu de fin de cycle	**Connaissances associées**	**Démarches, outils, exemples d'activités**
Documenter ses programmes D2 Partager ses programmes sur un réseau D2 Reconstituer la logique algorithmique d'un programme D2 Programmer des applications ludiques (labyrinthes, pong, bataille navale, nim, tic tac toe...) D2	Notion de variable informatique D1 Séquences d'instructions, boucles, instructions conditionnelles D1 gestion d'événements déclenchés par le clavier, la souris, etc. D1 Déclenchement de plusieurs séries d'instructions en parallèle D1 [4e-3e] Echange de messages entre objets, événements liés au déplacement d'un objet, clonage d'un objet D1	⇄ Réalisation et exploitation d'un sondage D3 ⇄ Traitements simples du texte (recherche, remplacement...) D3 Réaliser une figure à l'aide d'un logiciel de programmation pour consolider les notions de longueur et d'angle D4 ⇄ S'initier au chiffrement (Morse, chiffre de César...) D4 ⇄ S'initier aux principes des correcteurs orthographiques (calculs de tables de conjugaison, de pluriels...) D4 ⇄ Calculs de calendrier D5 ⇄ Calculs de répertoire d'adresses (recherche, recherche inversée...) D5

2.2 Projet de programme en technologie

Informatique, traitement numérique, démarche algorithmique et programmation	Chaine d'information Forme et transmission du signal Capteur, actionneur, interface. Algorithme, programme, instruction, séquence d'instructions, contrôle d'une séquence d'instructions Réseau informatique	Analyser un programme informatique en identifiant la nature des instructions, leur rôle dans la chaîne d'information et leurs effets sur le comportement du système technique commandé. Concevoir et représenter un algorithme en vue de programmer le comportement d'un système technique. Utiliser un environnement de programmation graphique pour réaliser un programme commandant un système technique simple Corriger/déboguer un programme en fonction des résultats obtenus. Analyser et créer une page plurimédia ou publication et application numérique répondant à un besoin de communication. Utiliser un réseau informatique pour transmettre des programmes et des documents.

Programme du Cycle 3

3 Evolution des systèmes Android

3.1 Les smartphones

Year	Product	Important features
1993	IBM Simon	The first smartphone – apps included a calendar, address book, world clock, calculator, note pad, e-mail, and games. Touch screen Text entered using "predictive" on-screen keyboard or QWERTY keyboard. Price: $899
1996	Palm Pilot	PDA; Touch screen (with stylus) Office apps as well as third party Handwriting recognition plus on-screen keyboard
1998	Nokia Communicator	Smartphone that combined keyboard and screen in a “micro laptop” clamshell format.
2002	RIM Blackberry	Smartphone with mini QWERTY keyboard Introduced RIM messaging service Very popular in business
2007	Apple iPhone	User-friendly multi-touch interface First app store launched with 2nd generation (3G)
2008	HTC Dream	First Android-based smartphone

3.2 Histoire d’Android

La société Android est créée en 2003, pus rachetée en 2005 par Google.

Chaque société de smartphone développe son propre système d’exploitation.

En 2007 arrive l’Iphone qui révolutionne le marché. L’Open Handset Alliance se charge de regrouper des sociétés pour concurrencer la marque à la pomme.

3.3 Evolution d’Android

Table 1

Worldwide Device Shipments by Segment (Thousands of Units)

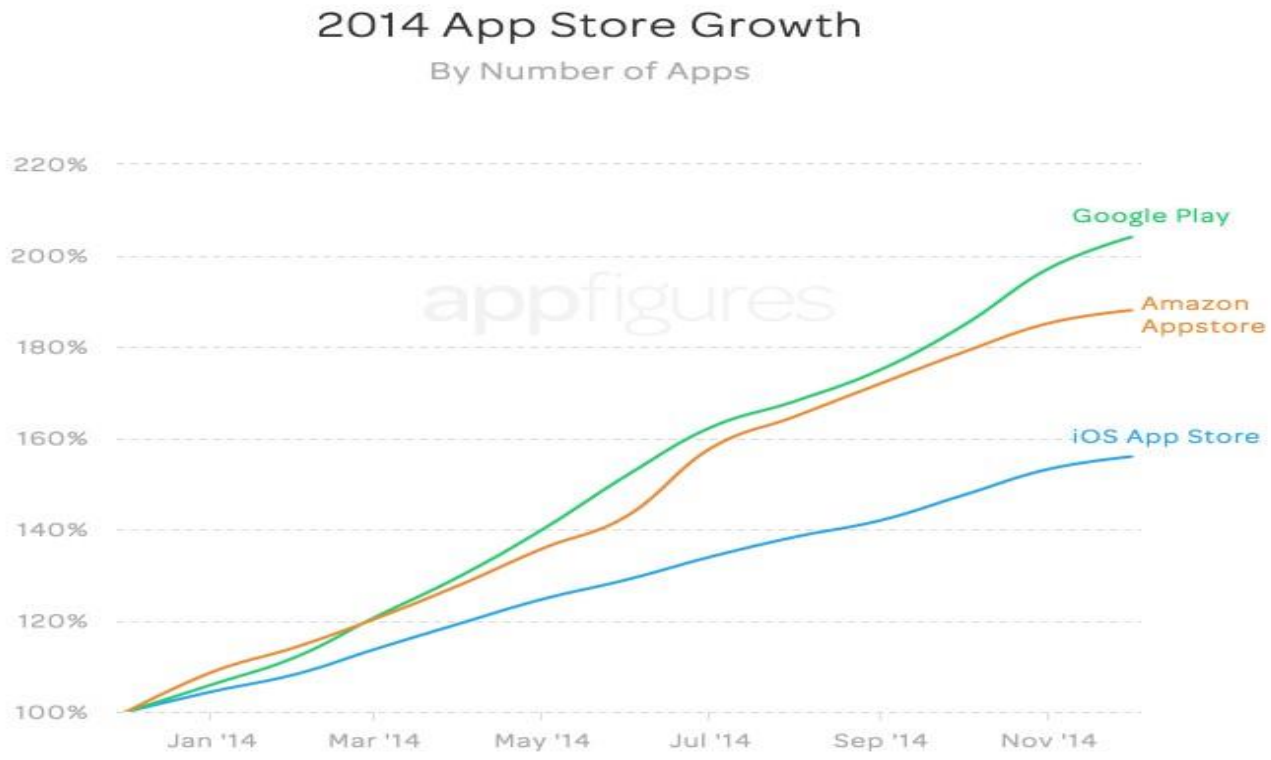
Device Type	2012	2013	2014	2015
PC (Desk-Based and Notebook)	341,273	299,342	277,939	268,491
Tablet (Ultramobile)	119,529	179,531	263,450	324,565
Mobile Phone	1,746,177	1,804,334	1,893,425	1,964,788
Other Ultramobiles (Hybrid and Clamshell)	9,344	17,195	39,636	63,835
Total	**2,216,322**	**2,300,402**	**2,474,451**	**2,621,678**

Table 2

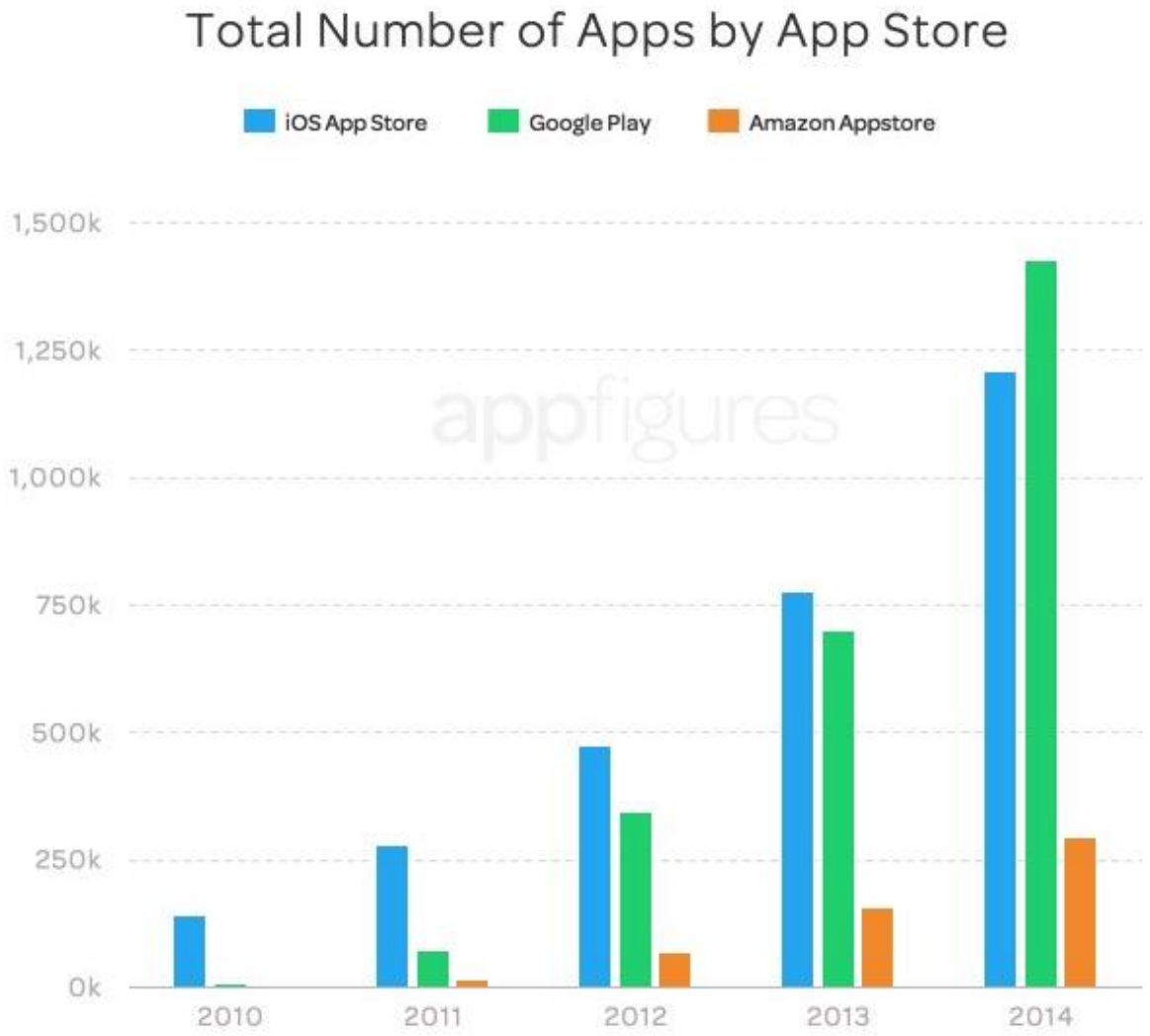
Worldwide Device Shipments by Operating System (Thousands of Units)

Operating System	2012	2013	2014	2015
Android	503,690	877,885	1,102,572	1,254,367
Windows	346,272	327,956	359,855	422,726
iOS/Mac OS	213,690	266,769	344,206	397,234
RIM	34,581	24,019	15,416	10,597
Chrome	185	1,841	4,793	8,000
Others	1,117,905	801,932	647,572	528,755
Total	**2,216,322**	**2,300,402**	**2,474,414**	**2,621,678**

Source : http://www.gartner.com/newsroom



http://www.phonandroid.com/le-play-store-surclasse-lapple-store-tous-les-niveaux.html



4 Système d'exploitation pour un smartphone ou tablette

Un terminal mobile a besoin d'un système d'exploitation pour fonctionner.

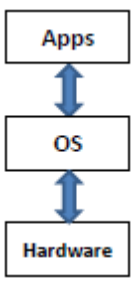
Androïd, est un système d'exploitation mobile pour smartphones, tablettes tactiles.

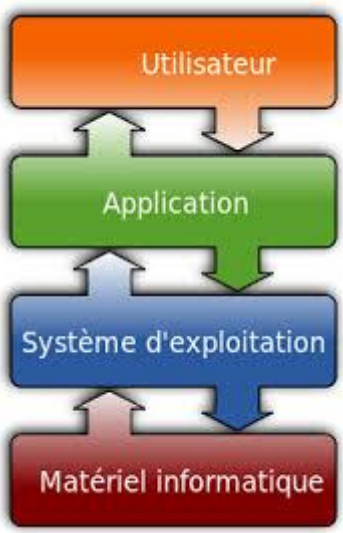
Il utilise un noyau Linux qui est un système d'exploitation libre.

Un système à besoin d'un système d'exploitation pour fonctionner. C'est une interface entre les applications et le matériel.

Le système d'exploitation à plusieurs rôles

- permettre d'accéder au matériel de façon transparente
- gérer les ressources (accès physiques, mémoire, CPU)
- veiller à la sécurité des applications et des données
- fournir une qualité de service (gestion du temps machine)
- être robuste





5 IDE AppInventor

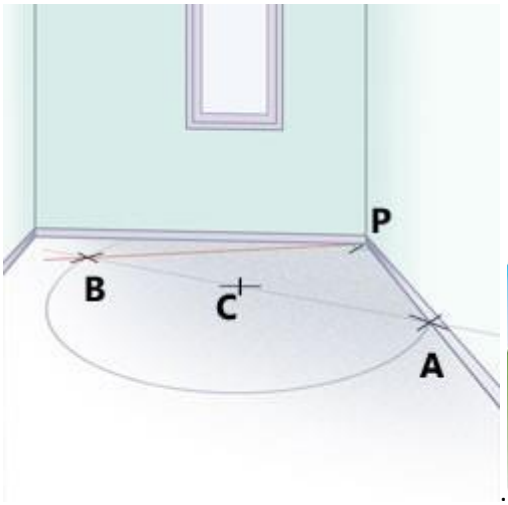
App Inventor permet de créer des applications Android à travers une interface purement visuelle (WYSIWYG pour « What You See Is What You Get ») et de configurer les actions de l'app par un système de blocs logiques.

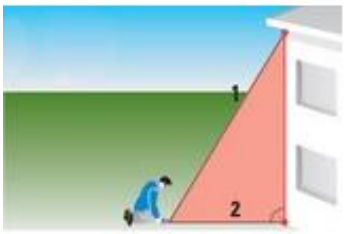
5.1 Méthode pédagogique

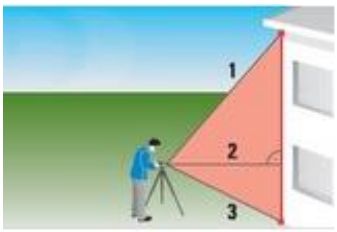
On commencera par des informations de haut niveau applicatif pour ensuite descendre dans les concepts par une pédagogie en spirale.

5.1.1 Analogie en mathématiques.

Voici un exemple autour du théorème de Pythagore. On utilise le théorème de Pythagore donné pour fixer la position d'un mur à angle droit ou faire un contrôle de cet angle.

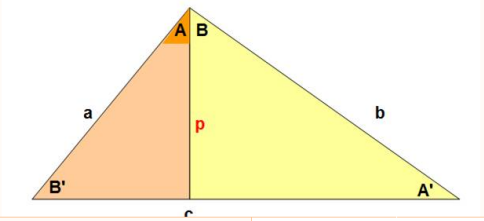






Puis on le démontre

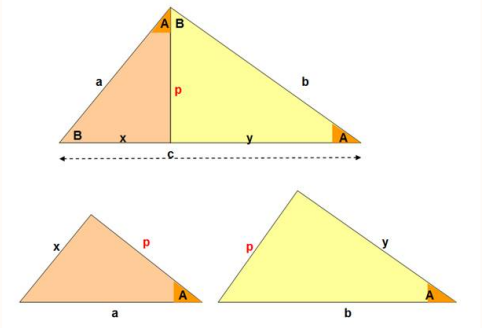
Trois triangles semblables



Soit un triangle rectangle initial. L'angle droit est la somme des angles A et B.	A + B = 90°
On dessine la hauteur.	hauteur p
Elle sépare deux triangles rectangles plus petits. La somme des deux angles est égale à 90°.	A' + B = 90° A + B' = 90°
En comparant avec la première égalité.	A' = A B' = B
Les trois triangles, le grand et les deux petits.	ont tous leurs angles égaux 2 à 2 => ils sont semblables

Conclusions pour notre théorème

- Dessinons les trois triangles de la même manière pour mieux s'y retrouver:
  - hypoténuse horizontale,
  - angle A vers la droite.



- Égalités dans les triangles semblables

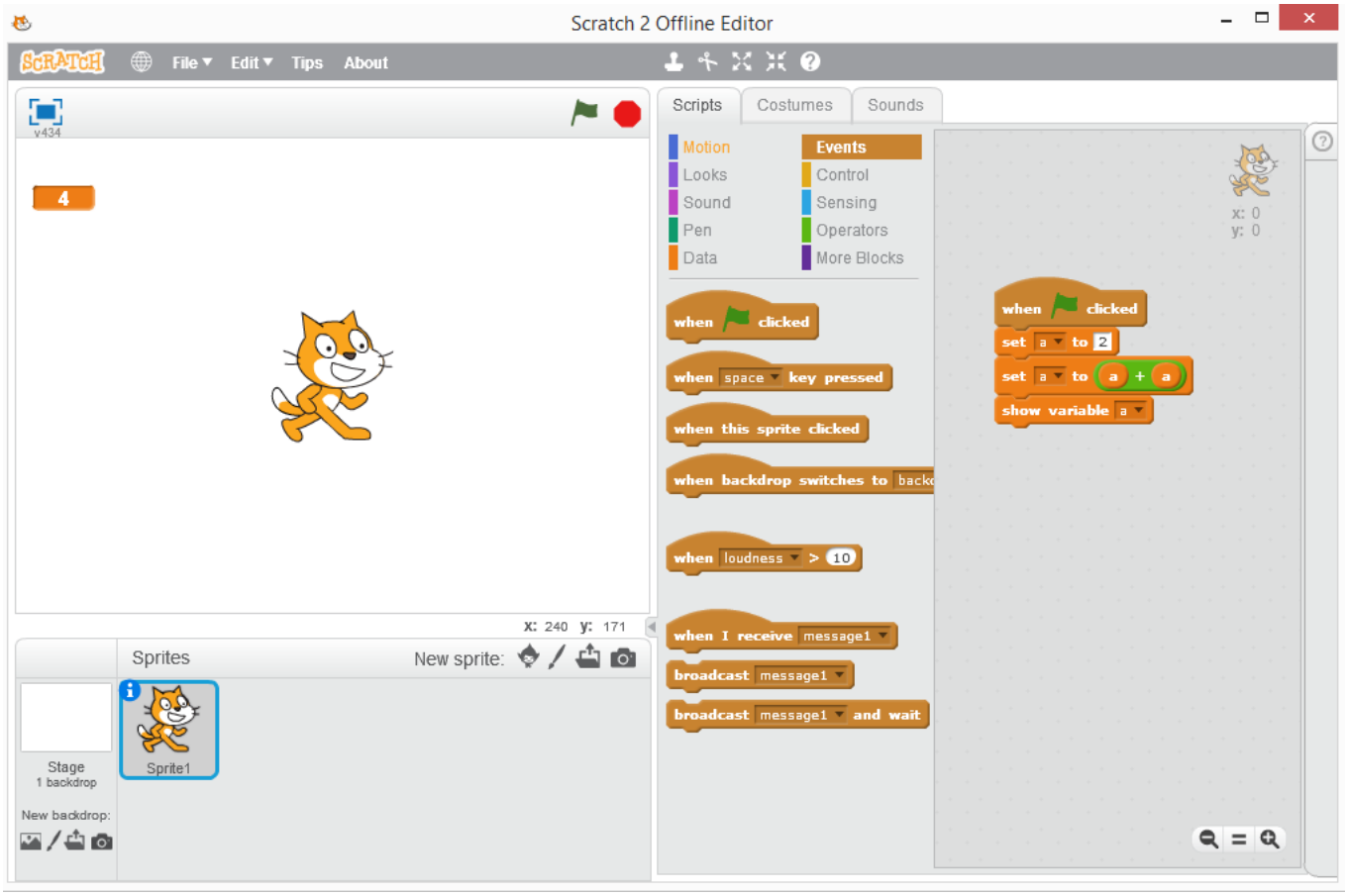
a/c = x/a	=>	$a^2 = cx$
b/c = y/b	=>	$b^2 = cy$
p/x = y/p	=>	$p^2 = xy$

- Première propriété: théorème de Pythagore

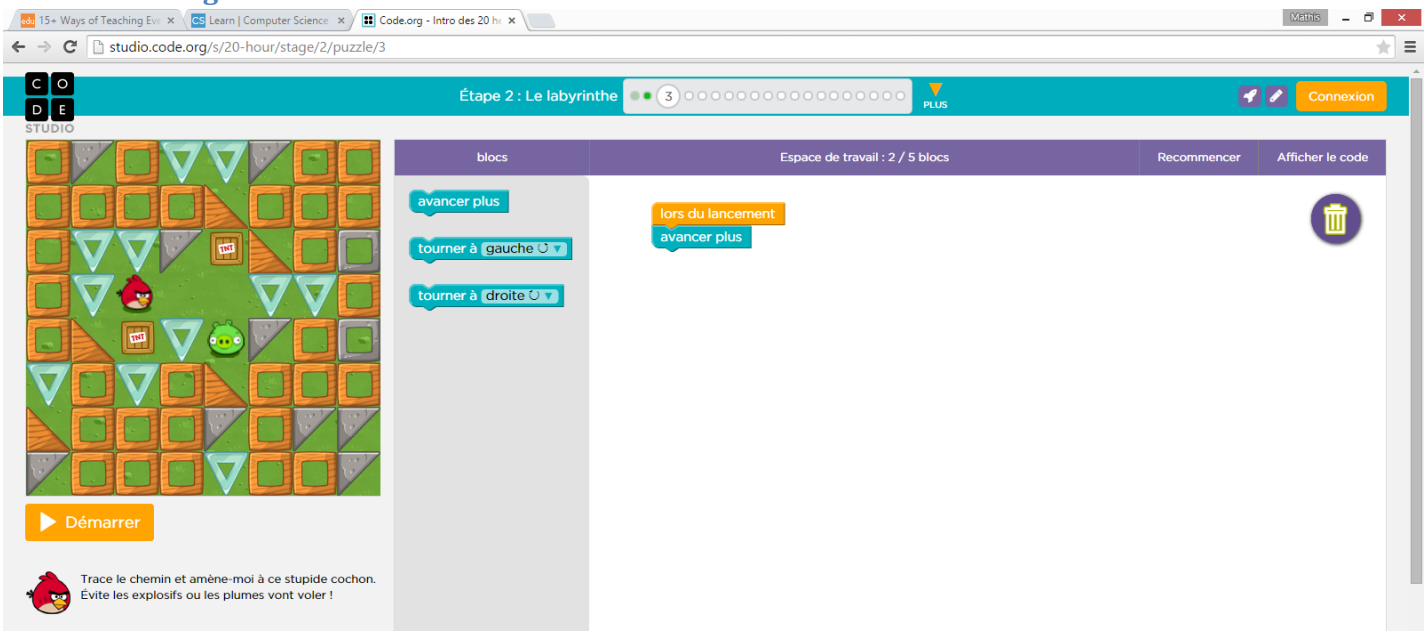
$a^2 = cx$ $b^2 = cy$	=>	$a^2 + b^2 = c(x + y)$ $= c (c)$ $a^2 + b^2 = c^2$

5.2 Exemples d'IDE qui utilisent le mode de programmation par blocs

5.2.1 Scratch2



5.2.2 Code.org



5.2.3 http://microalg.info/

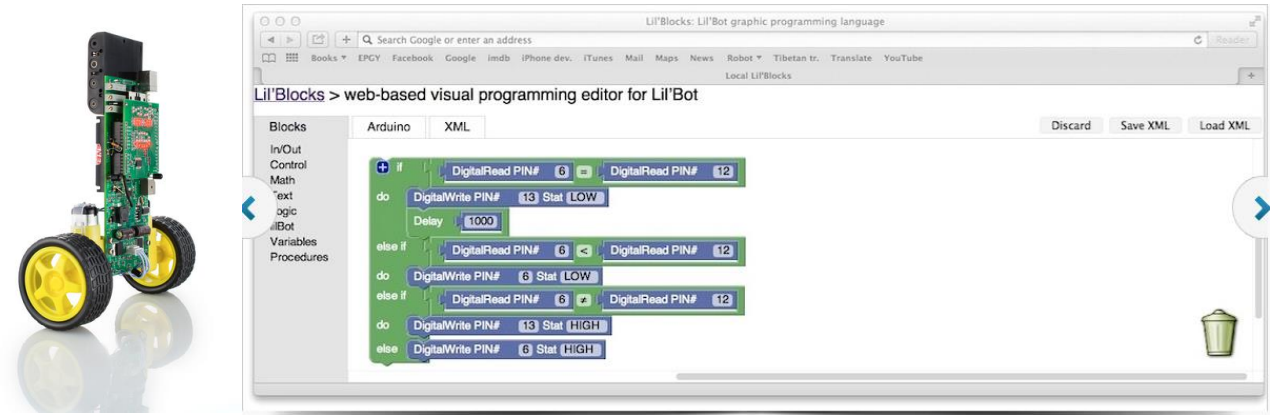
Site d'un collègue pour passer de la programmation par blocs à la programmation par pseudo code.

Blocs


```
(!!! "Bouger les blocs pour voir le code se mettre à jour.")
(Afficher "Veuillez entrer un nombre.")
(Afficher
    (Concatener
      "Un de plus: "
      (Texte (+ 1 (Nombre (Demander)))))
)
)
```

OK

5.2.4 Lil'Blocks

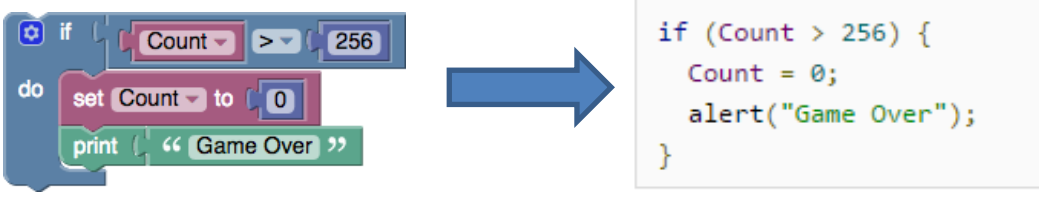


5.2.5 Blockly

L'IDE Blockly est maintenant réutilisé par plusieurs autres IDE.

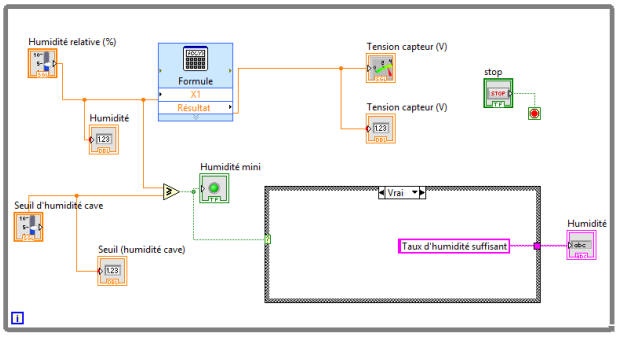
https://developers.google.com/blockly/about/examples

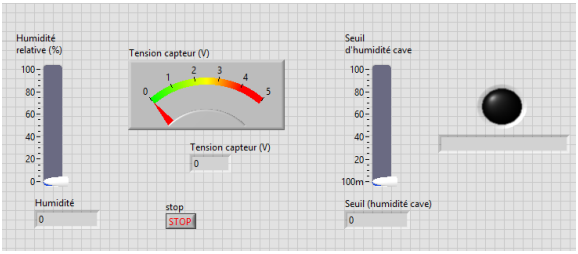
En fait Blockly traduit l'IDE graphique en codes Javascript



5.2.6 Programmation par flux de données

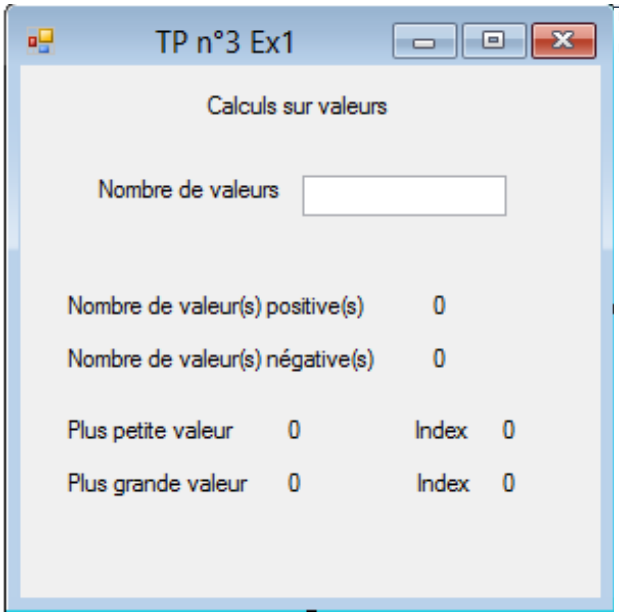
LabVIEW utilise le flux de données, pour cela on définit un flux d'exécution au travers un code de programmation, en créant des diagrammes qui montrent comment les données se déplacent entre les fonctions (appelées instruments virtuels ou VIs). Cette programmation est graphique. Elle utilise une partie IHM et une partie diagrammes.





5.2.7 Programmation dans un langage textuel.

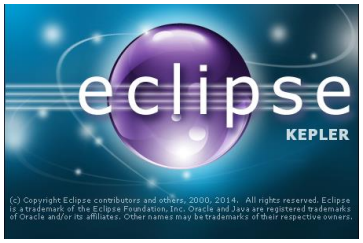
5.2.7.1 Visual studio



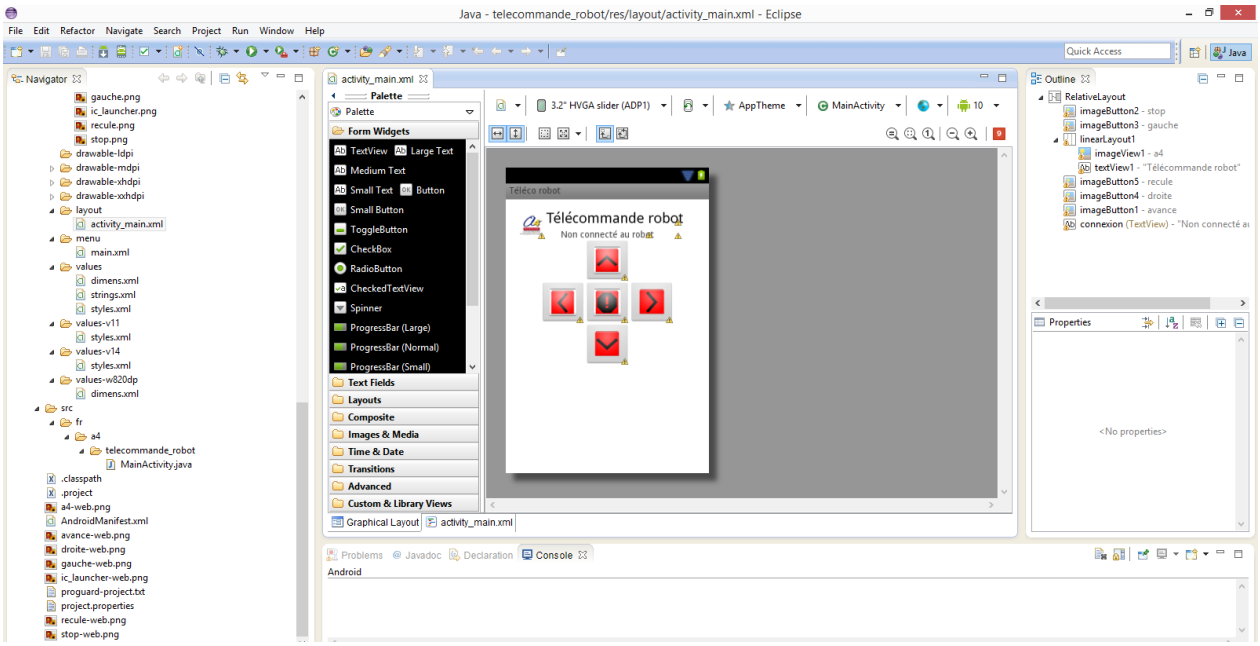

```
Public Class Form1

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        Dim i, I_nb_val, I_nb_val_pos, I_pos_min, I_pos_max As UInteger
        Dim I_min, I_max As Integer
        Dim val As Integer()
        I_nb_val = CInt(InputBox("Entrer le nombre de donner à saisir"))
        ReDim val(I_nb_val - 1)
        For i = 0 To I_nb_val - 1
            val(i) = CInt(InputBox("Donnée n°" + CStr(i + 1)))
            If val(i) >= 0 Then
                I_nb_val_pos = I_nb_val_pos + 1
            End If
            If i = 0 Then
                I_min = val(0)
                I_max = val(0)
                I_pos_max = 0
                I_pos_min = 0
            Else
                If val(i) < I_min Then
                    I_min = val(i)
                    I_pos_min = i
                End If
                If val(i) > I_max Then
                    I_max = val(i)
                    I_pos_max = i
                End If
            End If
        Next
        T_nb_val.Text = CStr(I_nb_val)
        T_nb_val_pos.Text = CStr(I_nb_val_pos)
        T_nb_val_neg.Text = CStr(I_nb_val - I_nb_val_pos)
        T_p_val.Text = CStr(I_min)
        T_index_p_val.Text = CStr(I_pos_min)
        T_g_val.Text = CStr(I_max)
        T_index_g_val.Text = CStr(I_pos_max)
    End Sub
End Class
```

5.2.7.2 Eclipse

Exemple de programmation en Java pour programmer un Smartphone ou tablette sous Android avec l'IDE Eclipse.



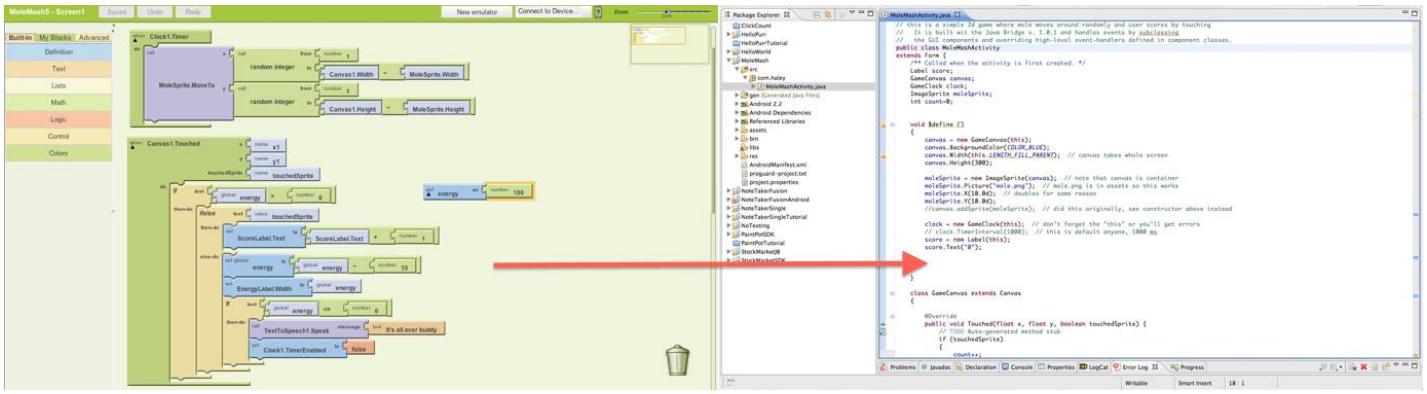
Définition d'un IHM avec ses objets.



activity_main.xml | MainActivity.java

```
//application pour piloter par exemple un portail coulissant automatique A4

package fr.a4.telecommande_1_bouton_attente;

import java.io.IOException;


public class MainActivity extends Activity {

    TextView connexion,text_attente;
    Button bouton_commande;
    EditText commande_envoyee,Text_temps_attente_max;
    Timer t;
    int temps,code_envoye,temps_attente_max=0;

    NetworkTask networktask;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        connexion=(TextView)findViewById(R.id.connexion);
        text_attente=(TextView)findViewById(R.id.temps_attente);
        bouton_commande=(Button)findViewById(R.id.bouton_commande);
        commande_envoyee=(EditText)findViewById(R.id.commande_envoyee);
        Text_temps_attente_max=(EditText)findViewById(R.id.temps_attente_max);
        bouton_commande.setOnClickListener(bouton_commande_listener);

        bouton_commande.setClickable(true);
        networktask = new NetworkTask();
        networktask.execute();
    }
```

Passage de la programmation par block à une programmation en java. La difficulté d'apprentissage est sans commune mesure entre les deux modes de programmation.



6 Exemple d'application smartphone

6.1 Téléchargement sur google play ou avec un fichier APK

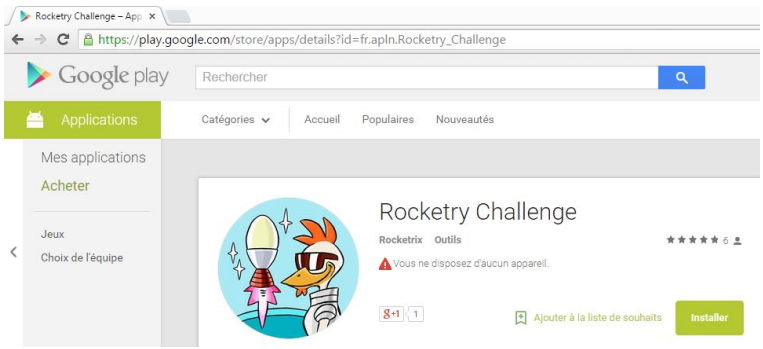
6.1.1 Google play

Vous pouvez installer directement une application à partir du site web de Google play. Il faut que l'application soit disponible sur Google play. Les applications disponibles peuvent être libres de droit ou payantes.

Télécharger l'application **Rocketry Challenge** disponible sur Google play.

Pour cela lancer l'application Google play qui est déjà installée sur votre tablette ou smartphone. Il suffit de taper dans le moteur de recherche le nom de l'application et de cliquer sur installer.

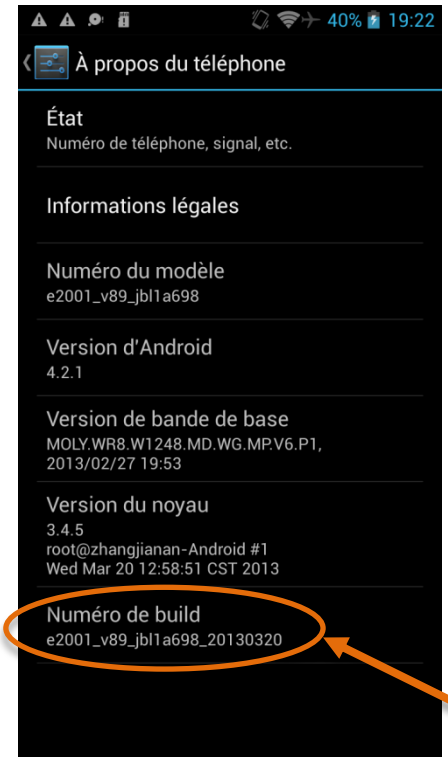
Une fois l'application installée, lancer l'application et testez là.



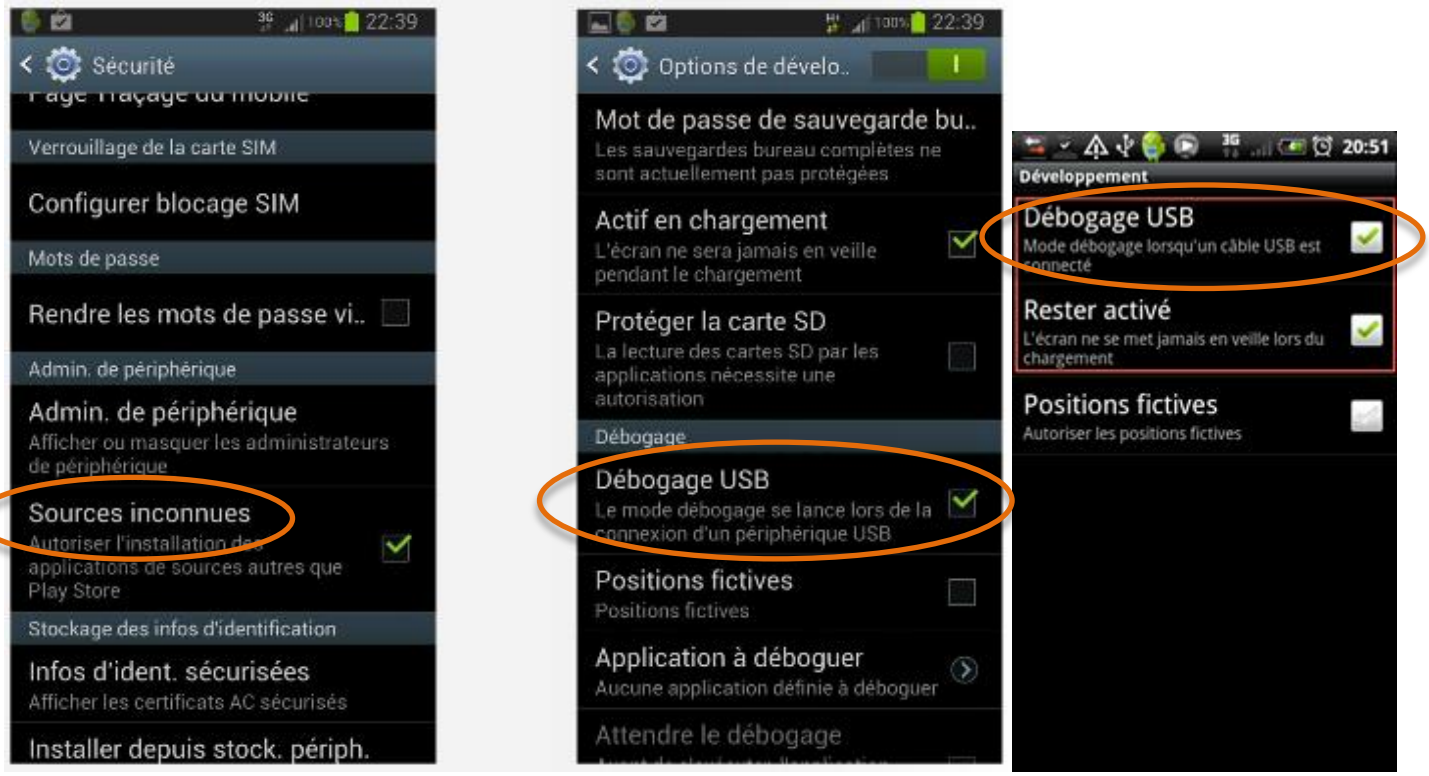
6.1.2 Paramétrage de son smartphone

Avant de commencer à utiliser son smartphone pour télécharger sa première application, il faut réaliser quelques réglages de ce dernier pour pouvoir utiliser l'outil de développement AppInventor.

Lancer l'application paramètre de votre smartphone.



Cliquer 7 fois ici pour activer le mode développeur.

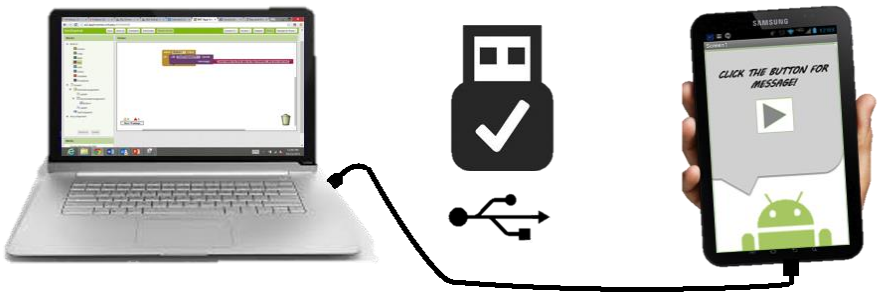


6.1.3 A partir d'un fichier APK

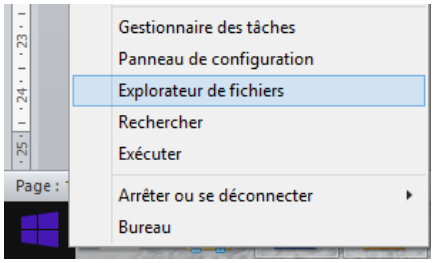
Une application pour smartphone ou tablette sous Android possède une extension de fichier APK (nom_fichier.apk).

Vous allez installer l'application ESSAI.APK qui se trouve sur l'ENT dans la partie formation/dossier.

Pour télécharger le fichier Apk qui se trouve sur votre ordinateur vers votre tablette ou smartphone connectez votre smartphone ou tablette via un câble USB à votre PC.



Utiliser l'explorateur de fichier de votre PC.

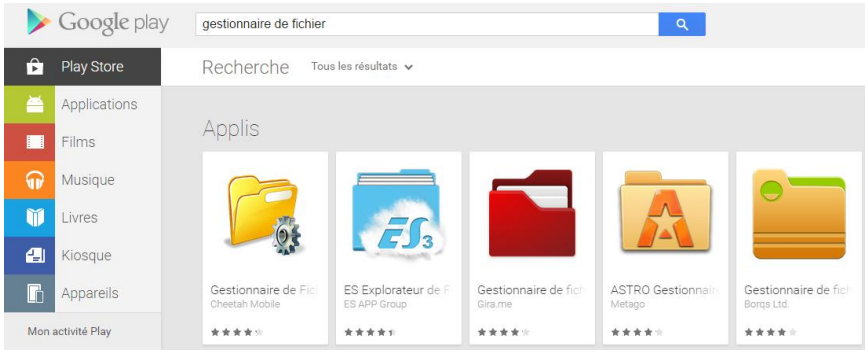


Vous devez alors voir votre tablette ou smartphone qui apparait dans la liste des périphériques de stockage.

Faite un copier-coller dans le répertoire de votre convenance dans votre tablette ou smartphone. Naviguer dans l'arborescence de votre tablette ou smartphone pour coller votre fichier APK. Il y a souvent un répertoire nommé « document ».

Il faut maintenant accéder au fichier APK collé dans votre smartphone ou tablette.

Soit vous avez déjà un gestionnaire de fichier installé sur votre tablette ou smartphone, soit installez un gestionnaire



Lancer ce gestionnaire et déplacez-vous dans l'arborescence pour retrouver votre fichier apk.

Cliquer sur le fichier apk, l'installation démarre, il suffit de suivre les instructions. Votre application s'installe et un icône est créer dans la liste des applications.

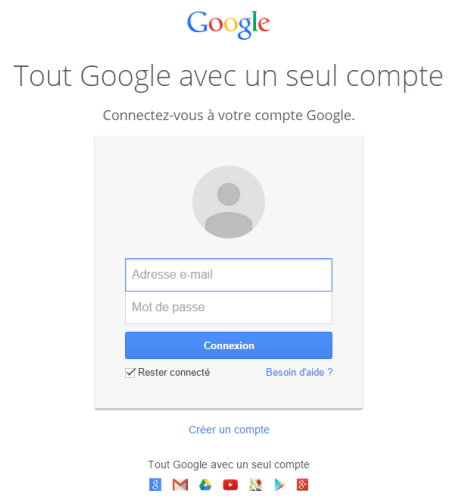
7 Interface homme/machine (designer)

7.1 Analyse d'un cahier des charges

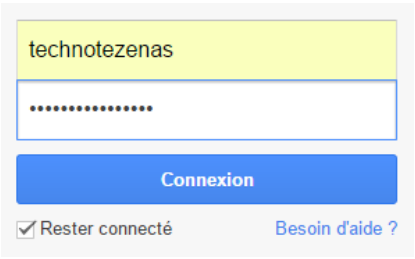
7.2 Se connecter

Taper dans la zone d'adresse URL http://ai2.appinventor.mit.edu/

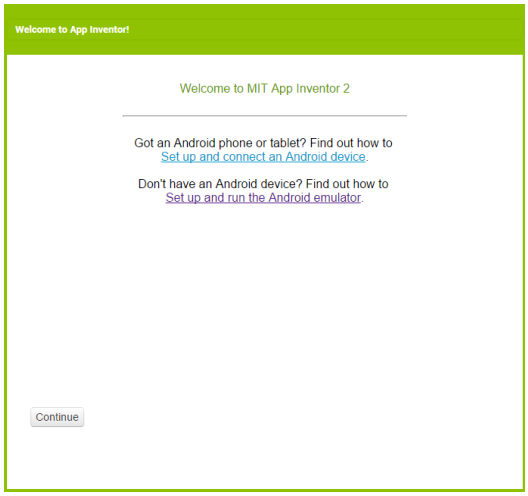
Le système va vous demander de vous connecter à un compte google.



Saisissez votre login et mot de passe personnel d'un compte Gmail.

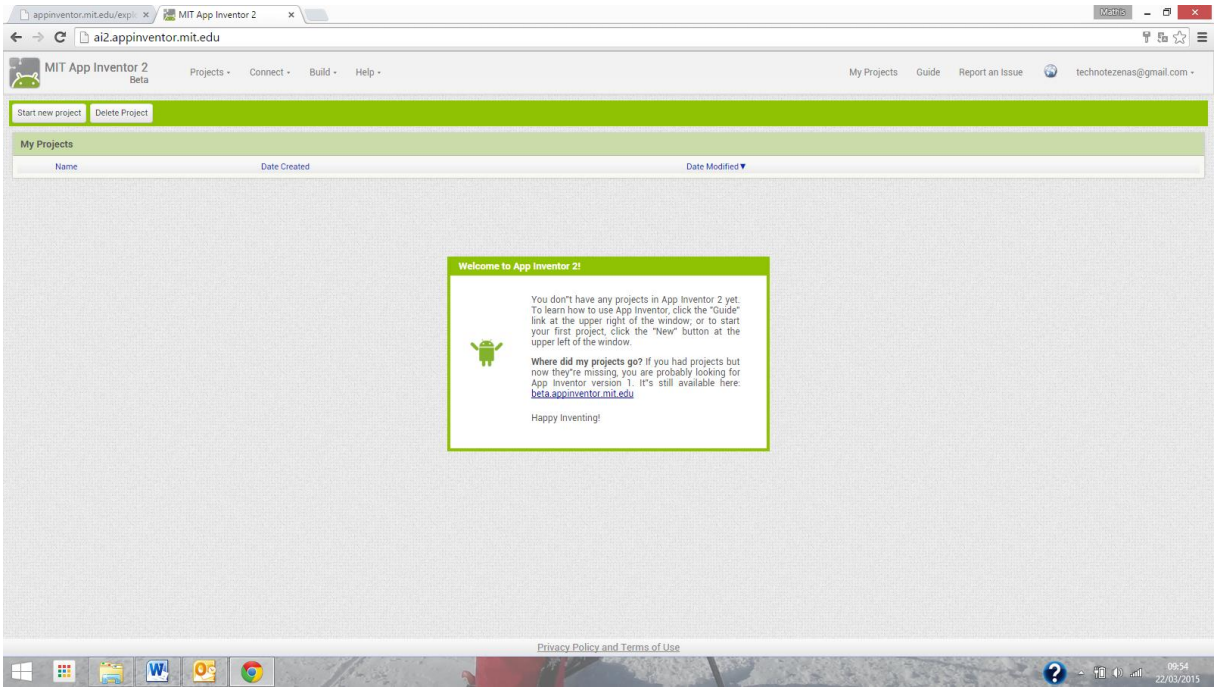


Cliquer sur continue

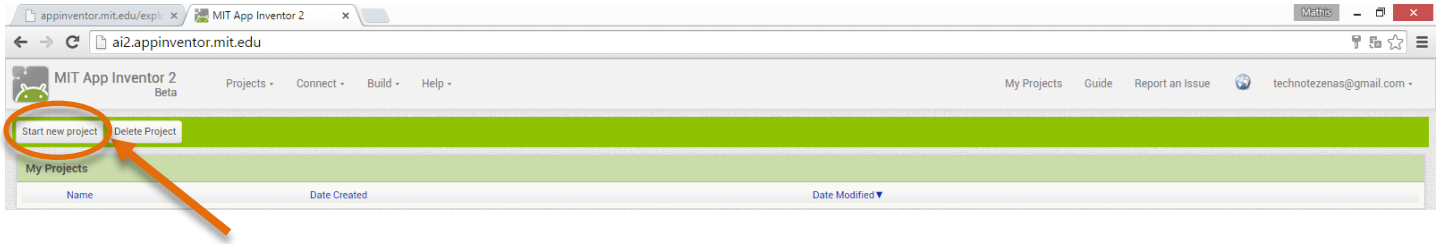


7.3 IDE d'AppInventor

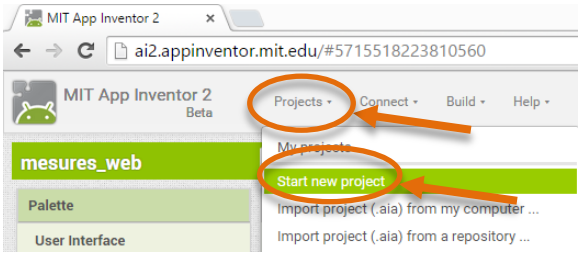
Vous êtes arrivés dans l'environnement d'Appinventor 2



Démarrez un nouveau projet.

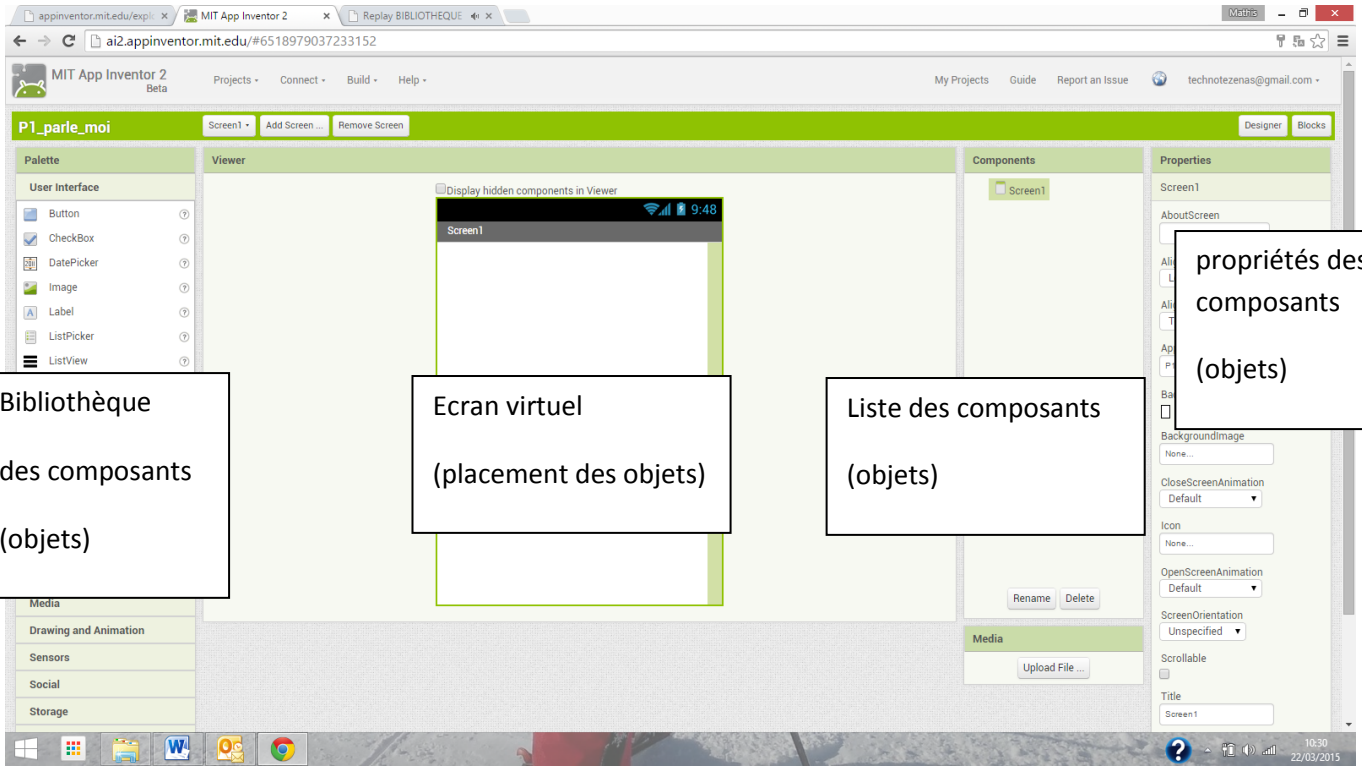


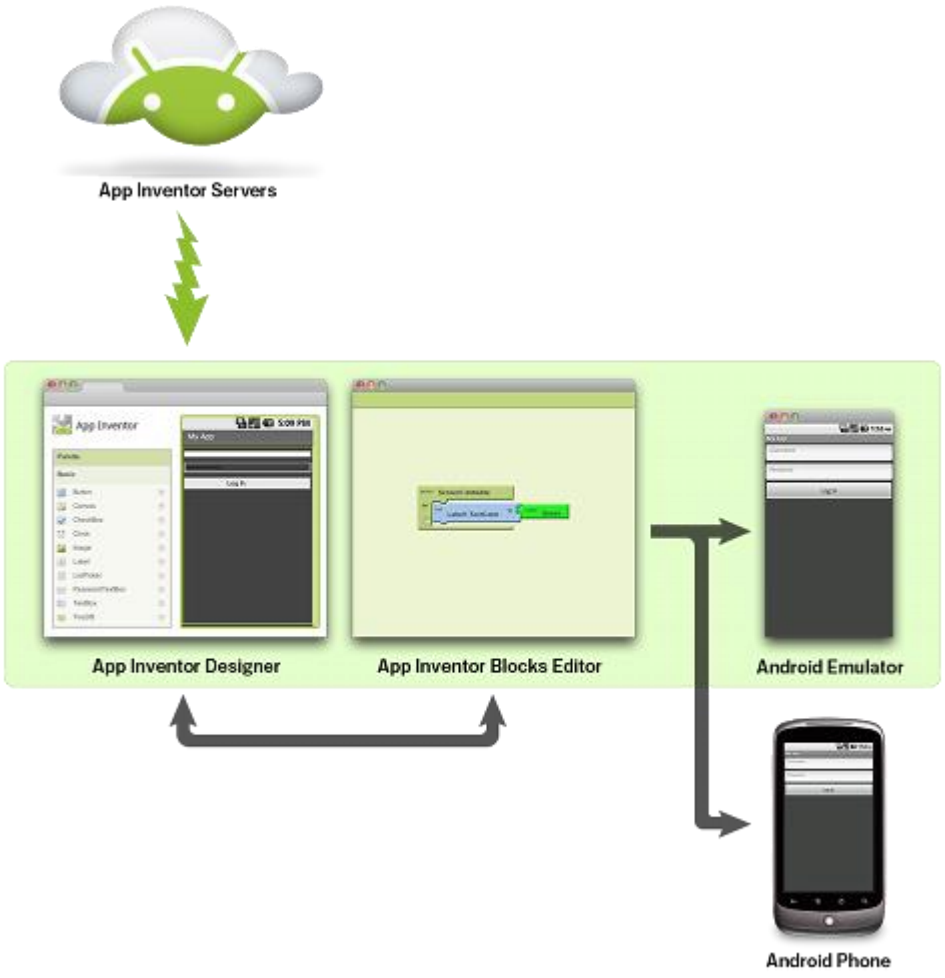
Cliquez sur Projects / Start new project.



Taper le nom de votre projet : EX_7_4_affichage

Vous obtenez l'écran de travail suivant :





7.4 Conception de l'application

7.4.1 Objectif de l'application

Vous allez réaliser une application qui ne fait que l'affichage d'informations textuelles dans un premier temps.

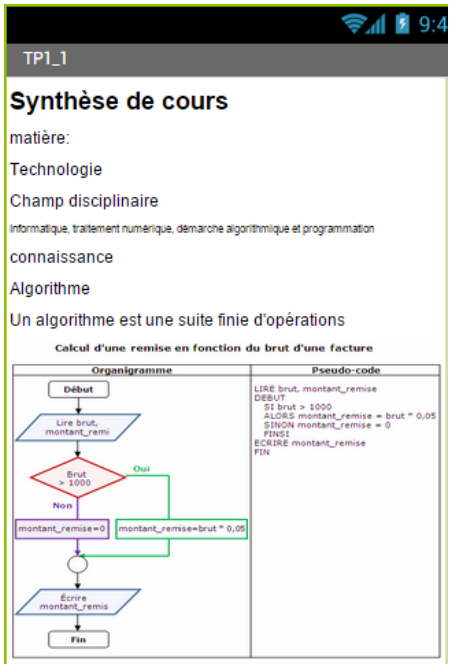
Cela pourrait-être par exemple un résumé des connaissances et compétences associé à un cours d'une matière donnée.

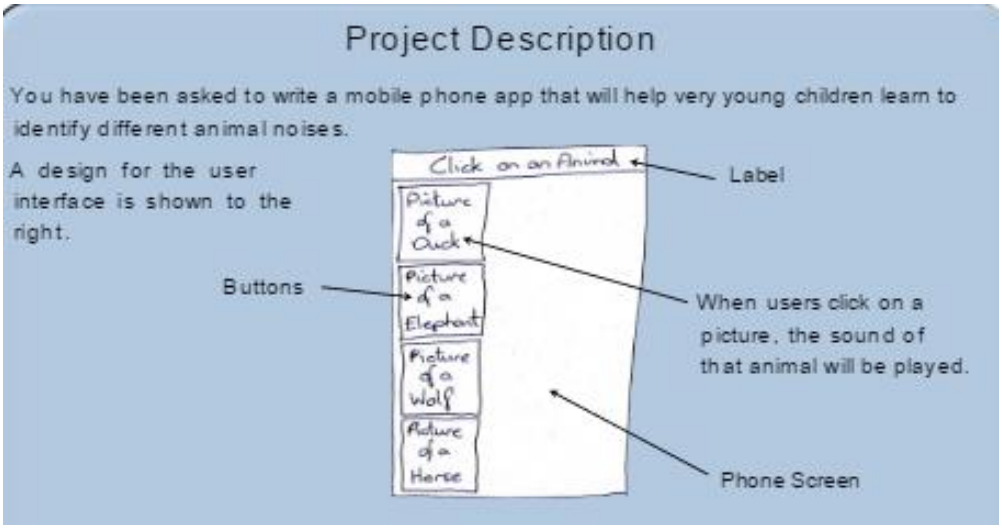
Ici, on se propose de mettre à disposition des élèves une synthèse sur la notion d'algorithme.

7.4.2 Dessin de l'IHM

La première étape est le dessin de l'interface homme/machine.

Il faut prendre un papier et un crayon et dessiner !





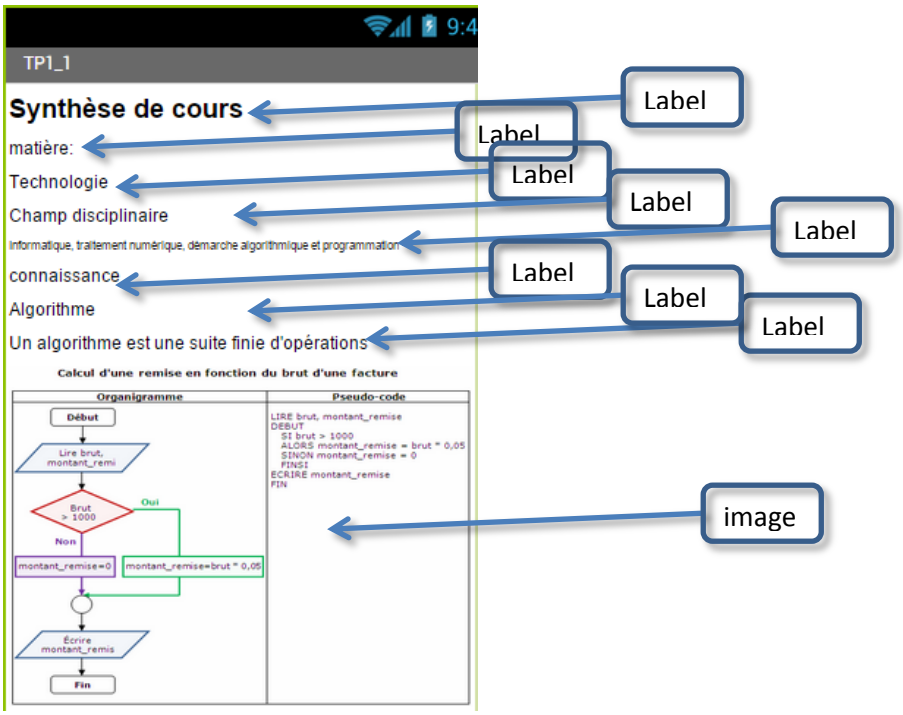
7.4.3 Objets à insérer

Une fois l'IHM dessinée, il faut indiquer de quel objet on a besoin dans l'IHM.

Ici on a besoin d'ajouter des zones de texte (objet label) ainsi qu'un objet image.

Il faudra donc ajouter sur le dessin ne type d'objet à insérer.

Vous créerez un projet nommé : EX_7_4_infos_texte

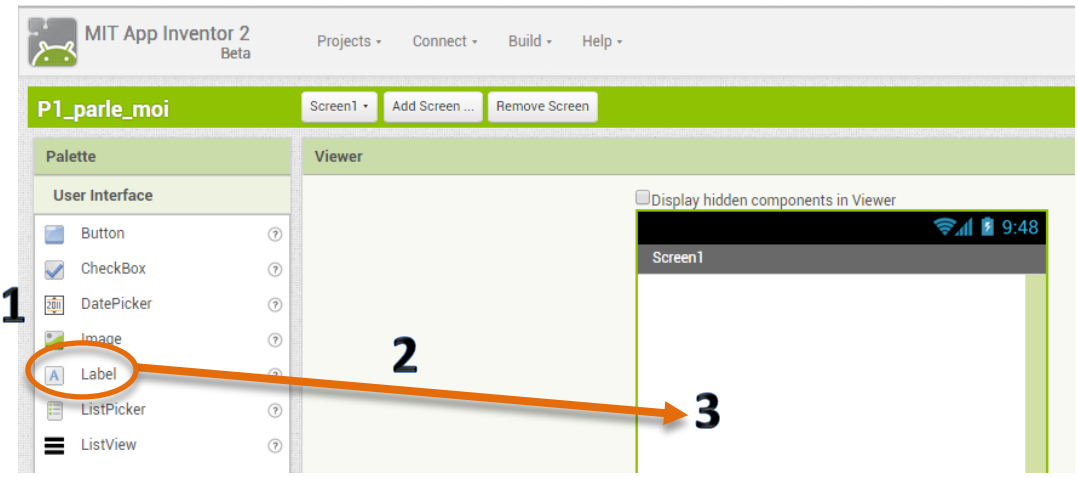


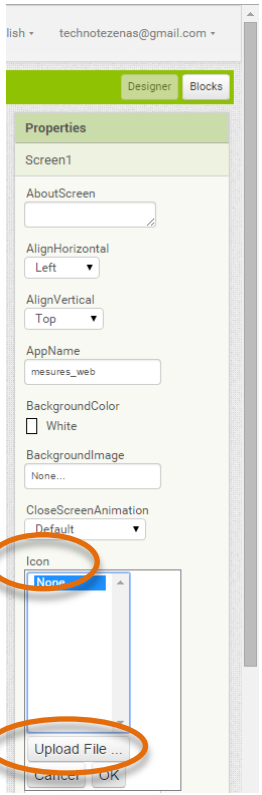
7.5 Les objets

Il faut maintenant sélectionner l'objet voulu dans la liste proposée sur Appinventor. Il existe de nombreux objets que l'on va découvrir en partie dans cette formation.

Ces objets sont disponibles dans la partie de gauche de l'interface Appinventor.

Pour placer un objet sur l'IHM, il suffit de faire un glisser déplacer de l'objet.





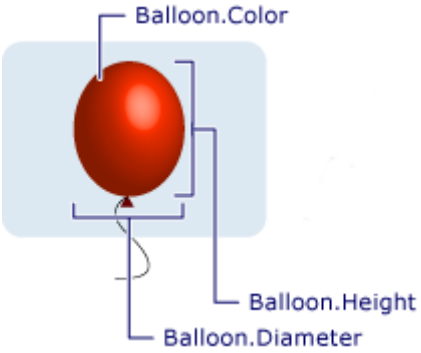
Vous pourrez aussi modifier le nom de votre écran ainsi que l''icône que vous souhaitez afficher sur votre smartphone pour identifier votre application. Pour cela vous sélectionnerez les propriétés de l'écran en cliquant sur l'écran puis…

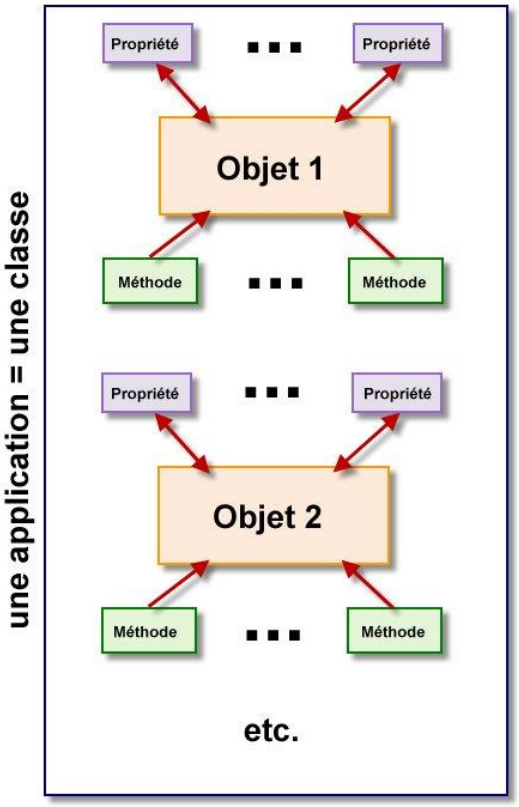
7.6 Positionnement des objets

Sans layout, c'est-à-dire les uns en dessous des autres sans positionnement particulier dans l'IHM.

7.7 Propriété des objets

Chaque objet a différentes propriétés. Une propriété caractérise l'objet.



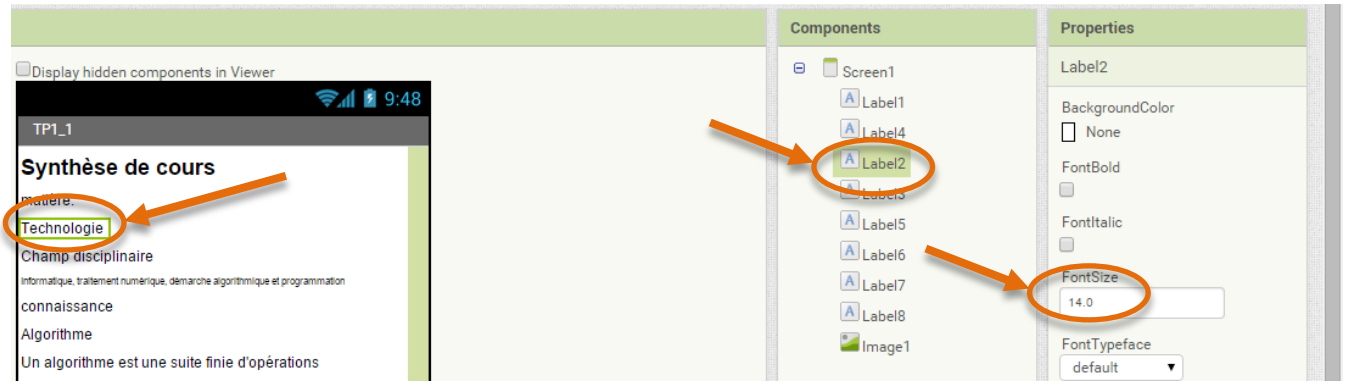


Propriété d'un objet en **vert**

Nom de l'objet

Propriété de l'objet

Ici on positionne la propriété taille de la police à 12



8 Simulation

Vous pouvez simuler l'application à l'aide de l'émulateur. Cela va vous permettre d'avoir un smartphone virtuel sur votre écran de PC et de vérifier son fonctionnement. Vous pourrez utiliser votre application comme sur un smartphone.

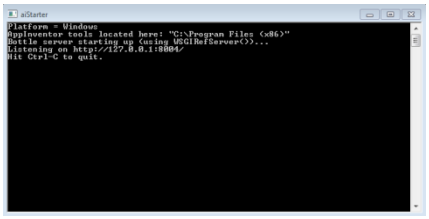
Build your project on your computer **Test it in real-time on your computer with the onscreen emulator**

Vérifier que vous avez bien installé aiStarter sur votre PC.

Lancer aiStarter après avoir lancer Appinventor puis AiStarter et pas l'inverse !

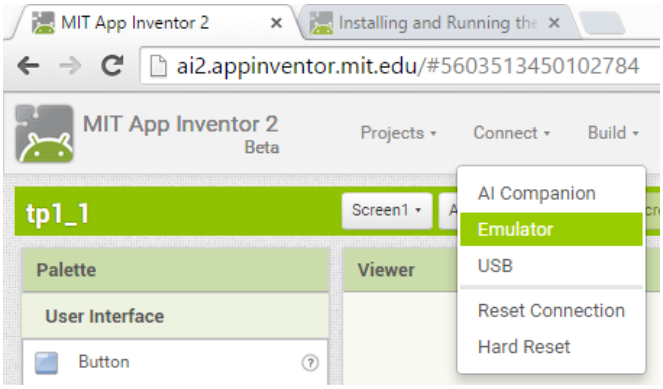
Ce programme permet au navigateur de communiquer avec l'émulateur ou câble USB. Ce programme aiStarter a été installé lorsque vous avez installé le package d'installation de App Inventor. Vous n'avez pas besoin d'aiStarter si vous utilisez seulement le compagnon sans fil.

Vous obtenez la fenêtre suivante qui s'ouvre



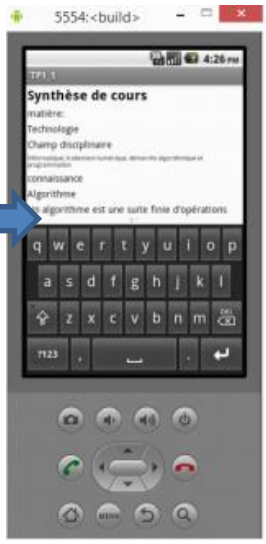
Retournez dans le navigateur sur AppInventor.

Cliquez sur le menu Connect/emulator



Il vous faudra être patient pour attendre que l'émulateur se lance, cela prend quelques minutes.

Voici la progression des différents écrans qui arrivent chronologiquement sur l'émulateur. Laissez faire et patience…



Pensez à laisser actif l'émulateur pour éviter de devoir le redémarrer à chaque fois !

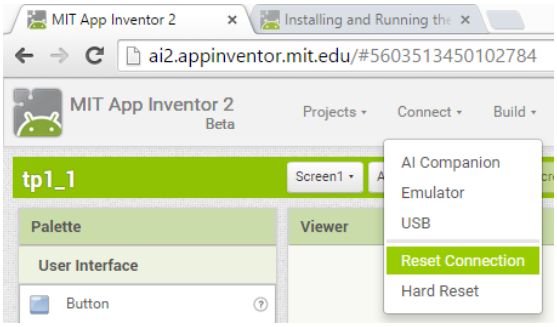
9 Compilation / fichier APK

Vous pouvez créer votre fichier APK qui pourra être ensuite téléchargé dans votre smartphone à l'aide d'un câble USB. Il suffira de copier-coller le fichier APK dans le smartphone.

Une fois téléchargé, il faut installer l'application.

10 Téléchargement / installation sur smartphone

10.1 Remise à zéro de l'émulateur



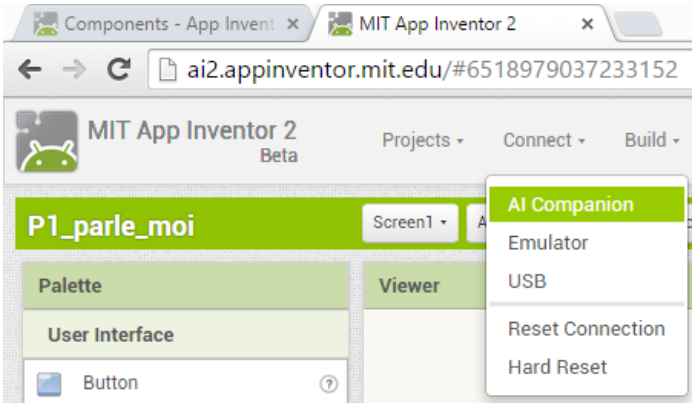
Attention, ne sélectionner jamais hard reset, sinon il vous faudra remettre à jour certaines applications !

10.2 Via QR code

Vérifier que vous avez bien téléchargé et installé l'application « MIT AI2 Companion » sur votre smartphone ou tablette.

Build your project on your computer

Test it in real-time on your device

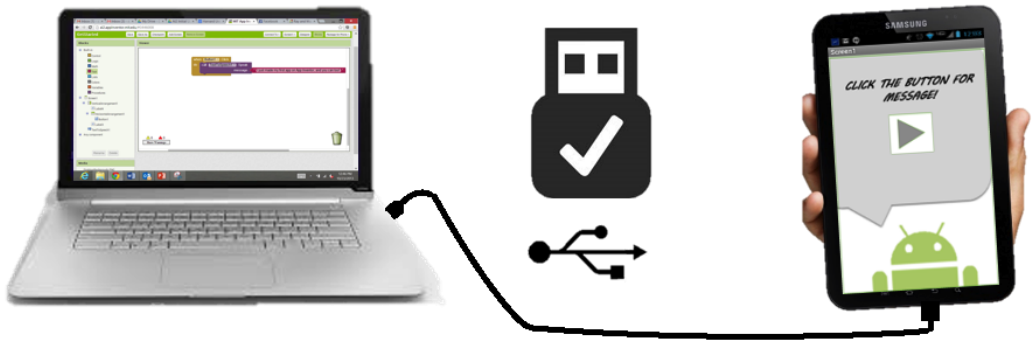


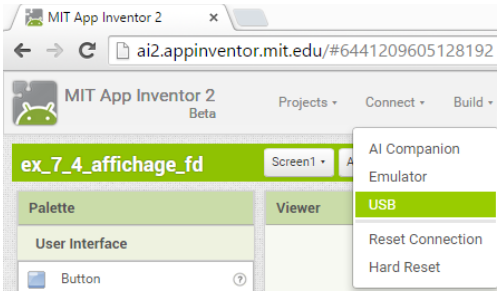
Vérifier que vous avez un accès wifi sur votre smartphone. Votre PC ainsi que votre smartphone devront être connecté à un point d'accès internet.

Lancer l'application AI2 sur smartphone.

Flasher à l'aide de votre appareil photo de votre smartphone le QRcode.

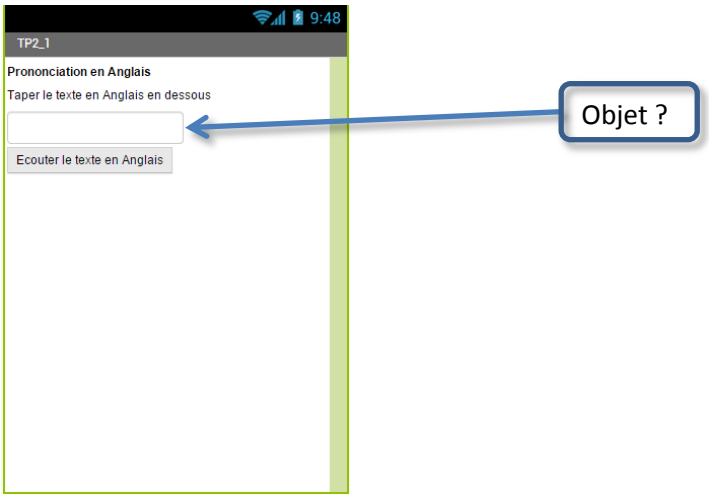
10.3 Test sur smartphone connecté via USB





11 Programme sur évènement

Exemple d'application qui parle. L'objectif de cette nouvelle application sera de prononcer dans la langue le texte tapé en Anglais.



1. Réalisez votre IHM
2. Recherchez les types d'objets

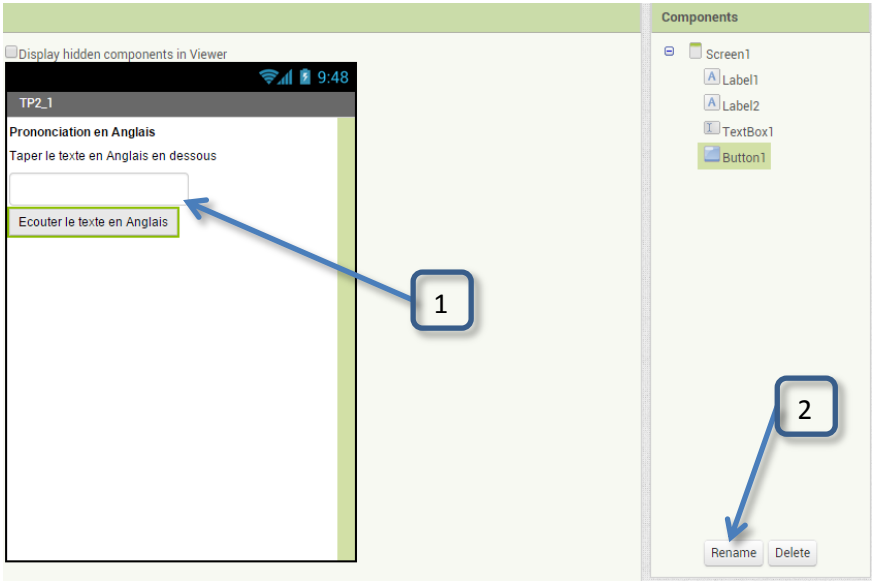
Ce qui est nouveau maintenant, est que l'application doit réagir suivant un évènement. Sur la première application, l'évènement principal était le lancement automatique de l'ouverture de l'écran principal. Cet évènement était automatique dès le lancement de l'application.

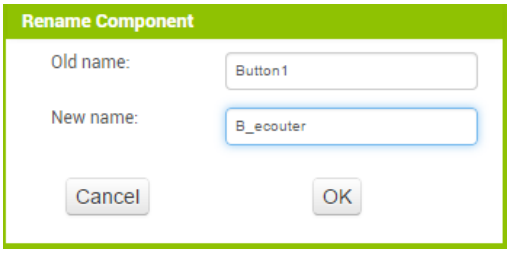
Ici, le nouvel évènement associé à l'environnement de la tablette ou du smartphone est l'appui sur le bouton « écouter le texte en Anglais ».

Il faut maintenant nommer l'objet que vous allez utiliser dans votre application.

Pour nommer l'objet de type bouton, il faut cliquer que l'objet pour le sélectionner et aller dans la partie composent de l'objet.

Puis cliquez sur Rename

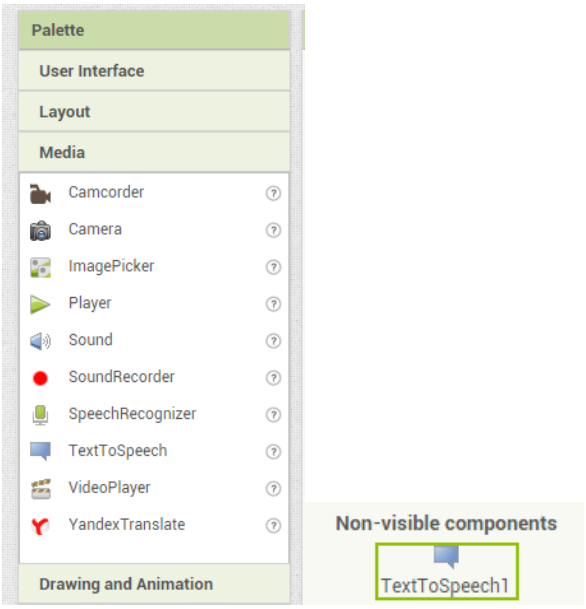




Vous allez synthétiser toutes ces informations indispensables pour le développement de votre application dans un tableau.

Tâche	**Type d'objet**	**Nom de l'objet**	**évènement**
Ecouter le texte en anglais	bouton	B_ecouter	Sur clic (appuie sur le bouton)

Il ne faudra pas oublier d'insérer un composent supplémentaire qui permettra la diction dans une langue à partir d'un texte donné.



11.1 Algorithme

Algorithme qui ne prend pas en compte la notion d'événement.

DEBUT

TANT QUE VRAI

SI bouton_ecouter appuyé

Lancer la prononciation de phrase_en_anglais

FIN SI

FIN TANT QUE

FIN

Algorithme associé à l'événement clique sur bouton

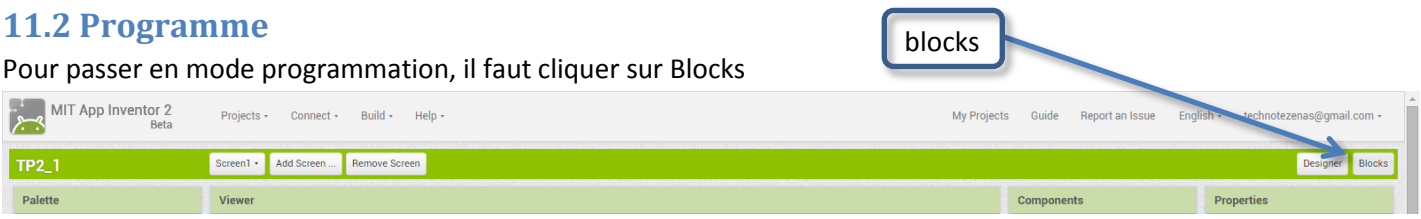
DEBUT

Lancer la prononciation de phrase_en_anglais

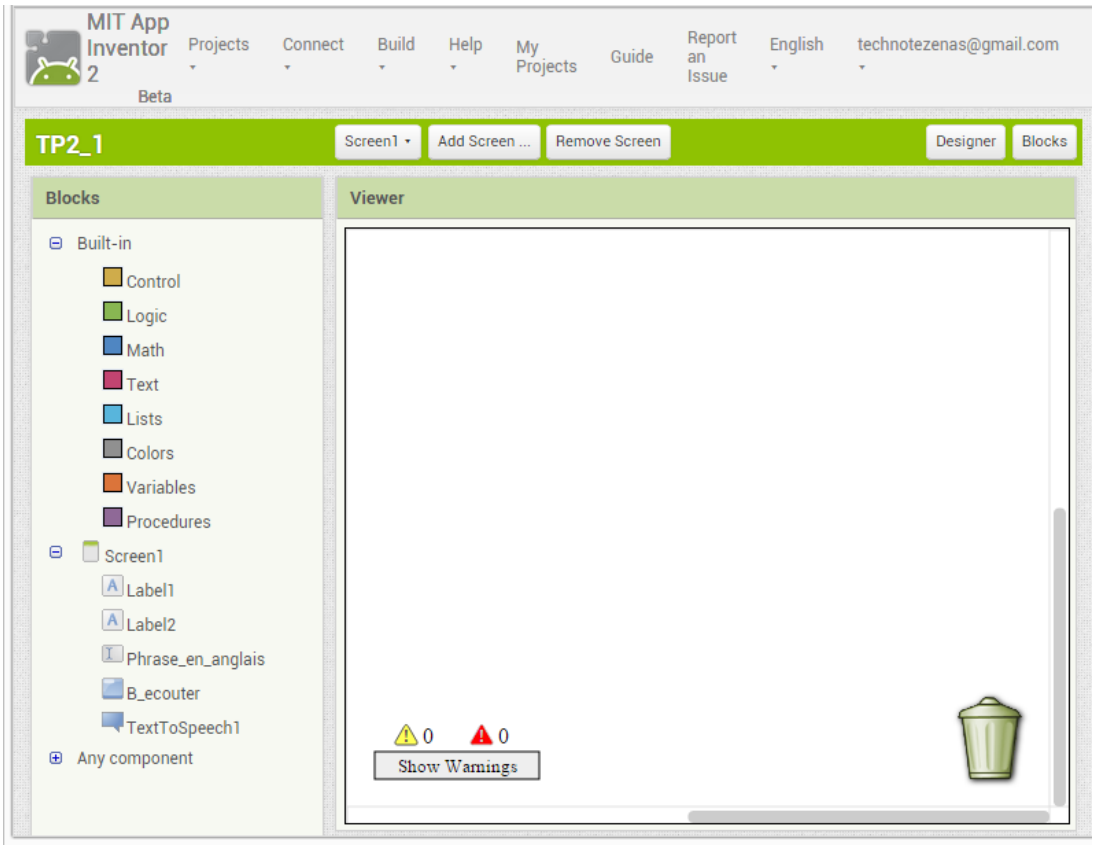
FIN

11.2 Programme

Pour passer en mode programmation, il faut cliquer sur Blocks

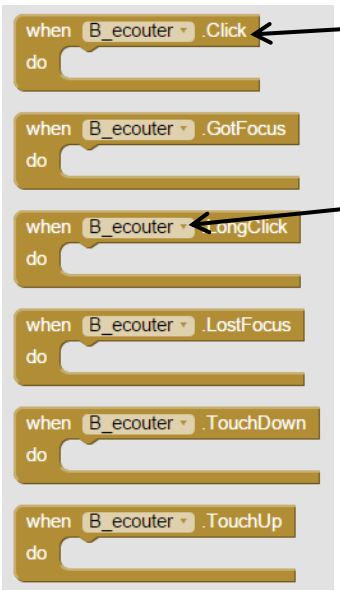


L'écran pour la partie programmation se présente de la façon suivante :



Il faudra sélectionner l'objet qui est associé à l'algorithme de l'événement choisi. Ici c'est l'objet bouton B_ecouter qu'il faut sélectionner pour lui associé l'algorithme précédent.

On a la liste des événements possibles associés à l'objet sélectionné.



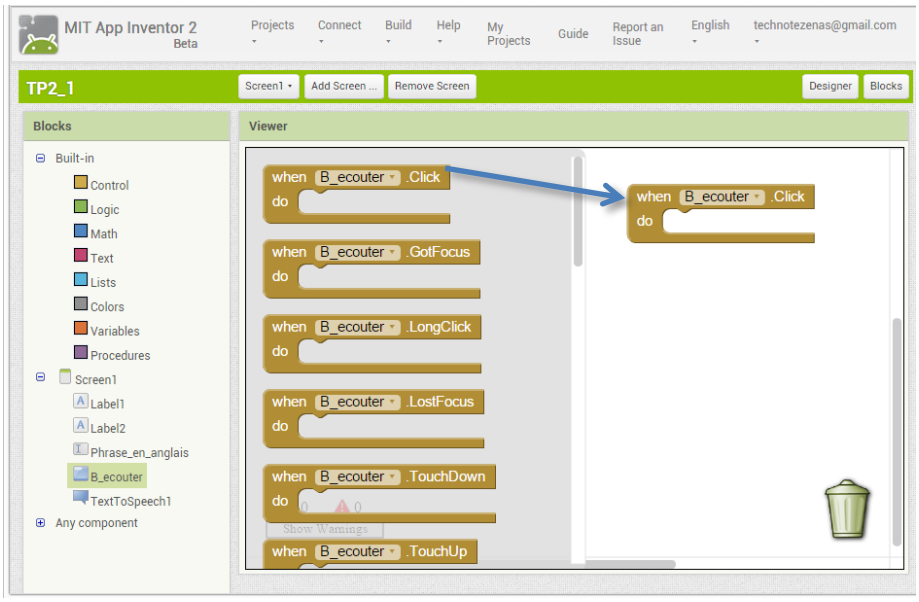
Type d'événement
Ici au clic

Nom de l'objet

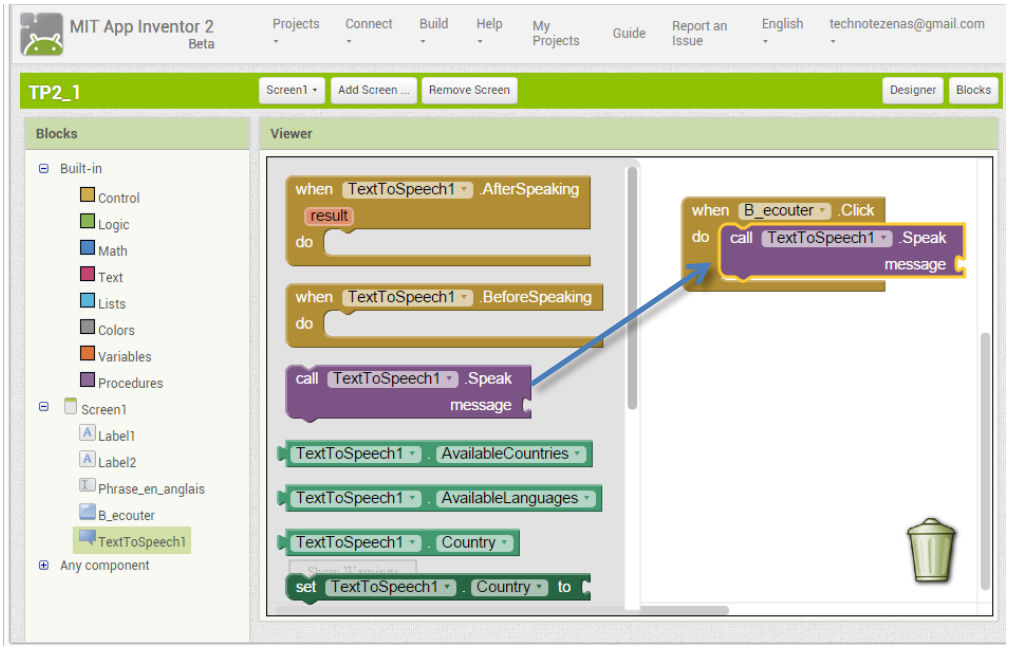
Un événement est en **marron**

Ici c'est l'évènement Click qui nous intéresse.

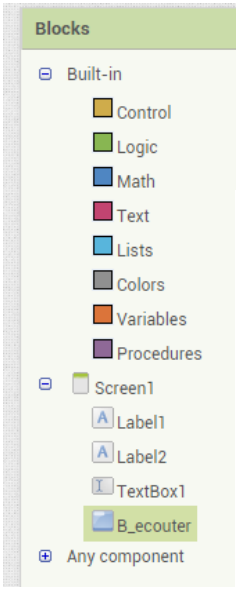
Il suffit alors de faire un glisser déplacer de l'évènement dans la zone de programmation à droite

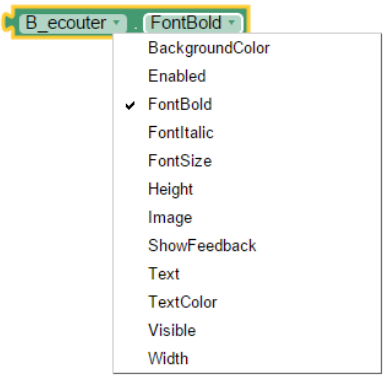


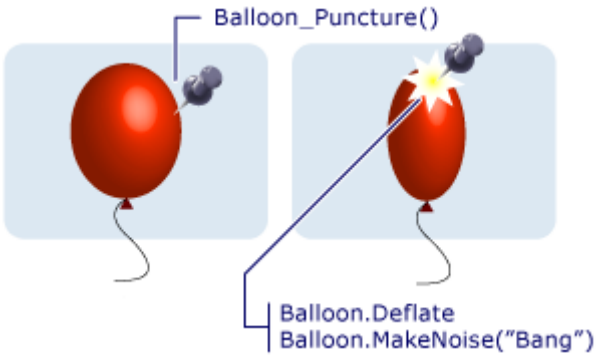
Il faut suivre l'algorithme qui demande de lancer la prononciation. Pour cela on va sélectionner le composant TextToSpeech1 et sélectionner une méthode associée.



Il suffit maintenant de lui spécifier quel est le message à prononcer, c'est-à-dire la propriété text de l'objet B_ecouter.

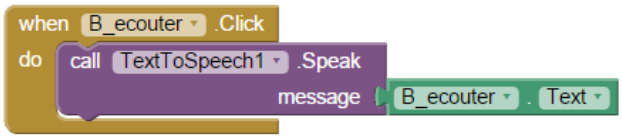


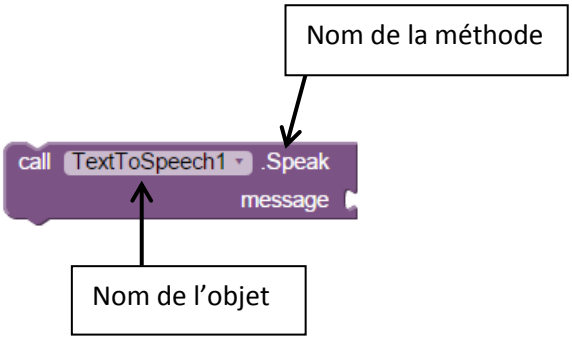


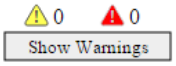


Une méthode est en **violet**

Vous obtenez alors le programme final suivant :





Vérifiez bien que vous n'avez pas d'erreur ni de warning 

Attention, ce n'est pas par ce qu'un programme n'a pas de warning ni erreur qu'il fonctionne !

11.3 Testez votre application

Simulez et télécharger votre application sur votre smartphone. Faite des essais.

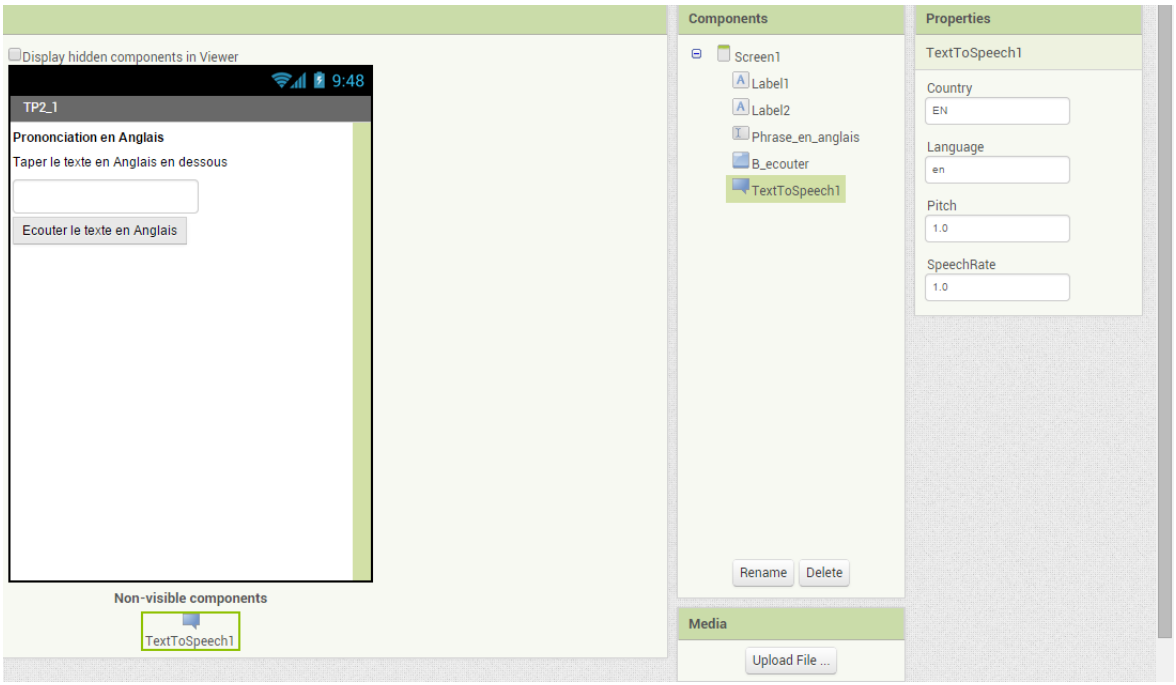
11.4 Modifications

Vous constatez que la prononciation n'est pas correcte.

Il faudra donc modifier la langue de la prononciation. Pour cela il faut se positionner sur les propriétés associées au composant TextToSpeech1 et rechercher l'information à renseigner.

Utiliser l'aide d'Appinventor pour connaître l'information à renseigner

http://beta.appinventor.mit.edu/learn/reference/components/other.html#TextToSpeech



11.4.1 Diagramme d'état

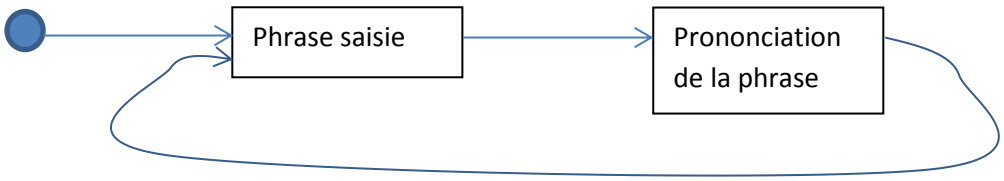
Analyse des évènements associés à des tâches.

Il faut commencer par lister les différents états stables de votre application.

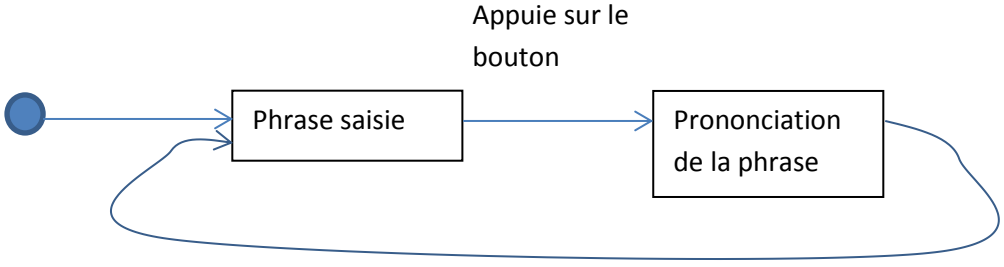
Ici la liste des états de l'application sont :

Phrase saisie / prononciation de la phrase dans la langue.

Ainsi le diagramme d'état est le suivant :

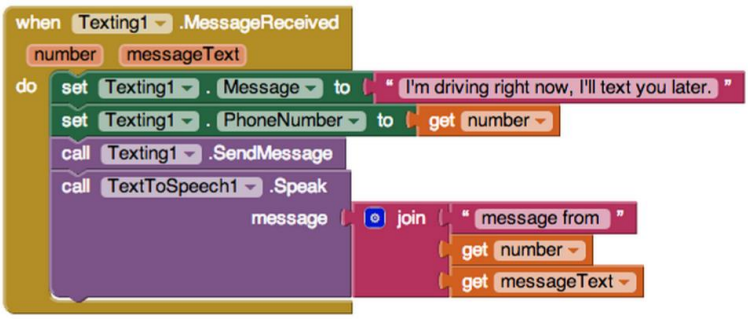


Dans un diagramme d'état il faut fixer aussi les transitions pour sélectionner quelle est l'information qui permet le passage d'une tâche à une autre.



11.4.2 Exercice sur un programme

Que permet de faire ce programme ?



Retrouver l'IHM, les objets, leurs propriétés, l'algorithme

12 Interface homme/machine avec positionnement

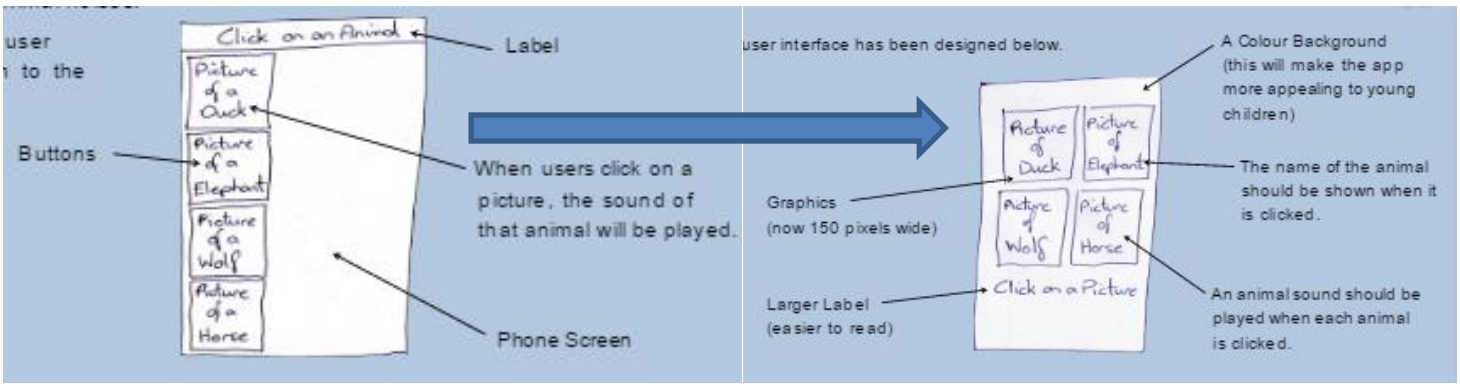
12.1 Positionnement des objets

Avec layout

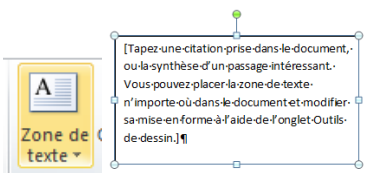
Vous avez remarqué qu'il n'était pas possible de positionner des objets les uns à côté des autres horizontalement.

Suivant votre IHM vous avez besoin de positionner précisément des objets comme on le veut.

Exemple pour passer de l'IHM de gauche à celle de droite



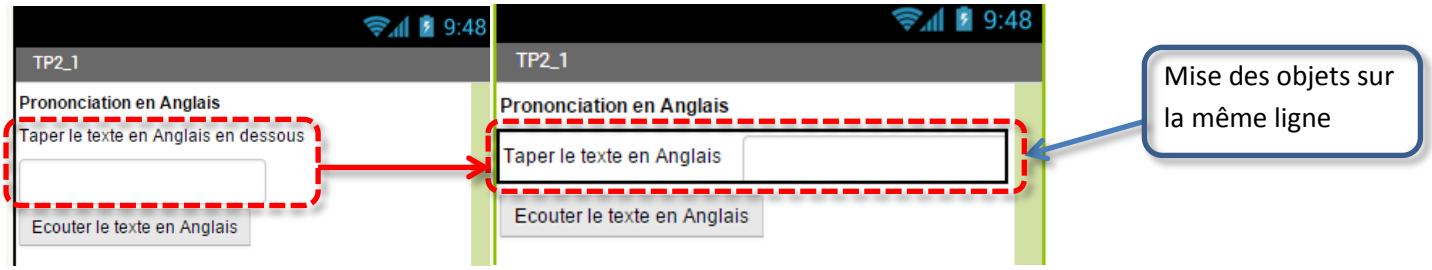
C'est la même chose que sur un traitement de texte pour positionner une zone de texte, mais la méthode est plus basique.



Avec AppInventor la méthode est la suivante :

Il faut commencer par découper en différentes zones l'écran de l'IHM

On souhaite modifier la dernière application pour présenter différemment les différents objets. Pour cela pensez à un faire un projet clone de l'application. Vous nommerez l'application EX_12_1_parle_anglais.



Pour cela vous utiliserez l'outil layout

L'outil vous propose plusieurs positionnement possible, soit aligner des objets :

- les uns à côté des autres,
- les uns en dessous des autres,
- dans un tableau.

les uns à côté des autres	les uns en dessous des autres	dans un tableau
HorizontalArrangement	VerticalArrangement	TableArrangement

Faite un glisser déplacer de la forme du Layout voulu dans l'IHM, vous voyez alors apparaitre la zone du layout, mais vide à l'intérieur.

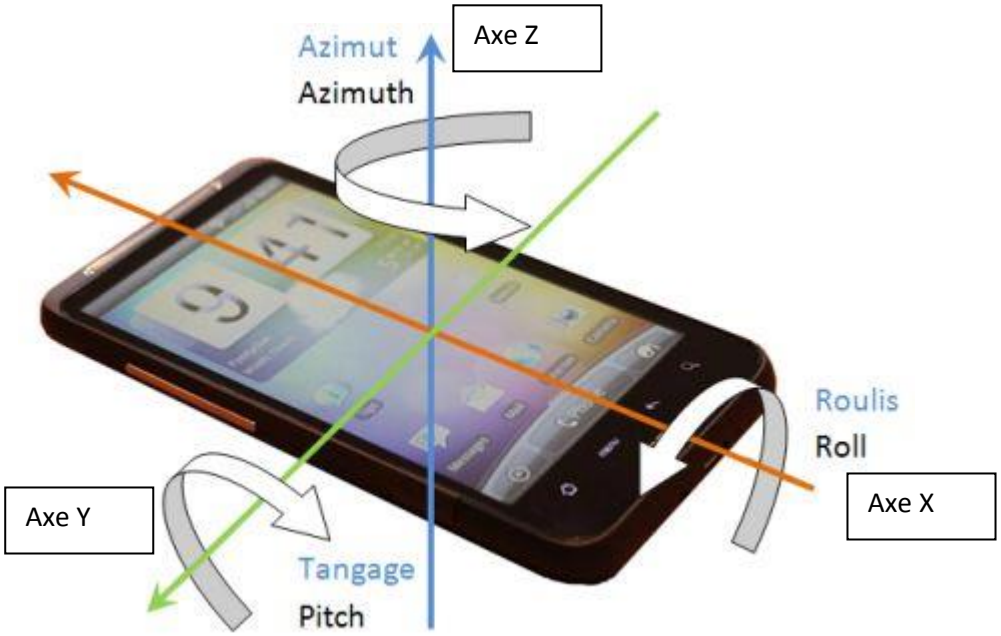
Il vous suffit de faire un glisser déplacer des objets dans ce layout.

Attention de bien paramétrer les propriétés de cet objet layout de positionnement !

13 Programmation avec variable

Il est parfois indispensable d'utiliser des variables pour pouvoir sauvegarder des résultats intermédiaires afin de réaliser des calculs d'un algorithme.

Réalisation d'une application niveau (axe à définir)



Le smartphone ou tablette possède des capteurs pour pouvoir « ressentir » des accélérations. Il possède un capteur qui permet de mesurer ces accélérations suivant les 3 axes.

L'application impose de ne pas bouger pour pouvoir mesurer un angle correct qui ne variera le moins possible. Oui, mais si le système est stable, alors il n'y a pas d'accélération. Il y a cependant une accélération que nous subissons en permanence qui fait que nous avons les pieds sur terre, c'est la force de gravitation P=m.g, g justement l'accélération de la pesanteur. Cette accélération dépend de la masse des objets voisins et de leur distance. Comme la terre est l'élément qui a la masse la plus importante et que la distance smartphone/terre a de très faible variation, on prendre g=constante. Le vecteur accélération est donc vertical et dirigé vers le bas. Donc logiquement, notre accéléromètre doit pouvoir détecter cette accélération même avec un smartphone immobile.

13.1 Réalisation d'une première version (visualisation des valeurs des capteurs)

Vous allez commencer par visualiser les valeurs des 3 accéléromètres sur les 3 axes.

Dessinez votre IHM papier, sélectionnez vos objets, rechercher les propriétés éventuelles associées.

Créer votre diagramme d'état

Vous allez pour cela utiliser un évènement associé à l'objet sensor.

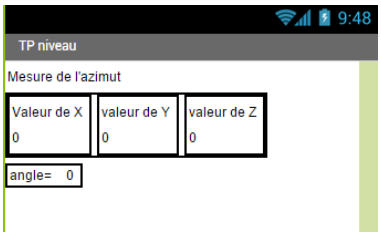
Compléter votre tableau tâche/objet/évènement

Concevez votre algorithme (graphique/ textuel)

Créer votre projet sur Appinventor nommée EX_13_1_visu_capteurs

13.2 Calcul de l'angle

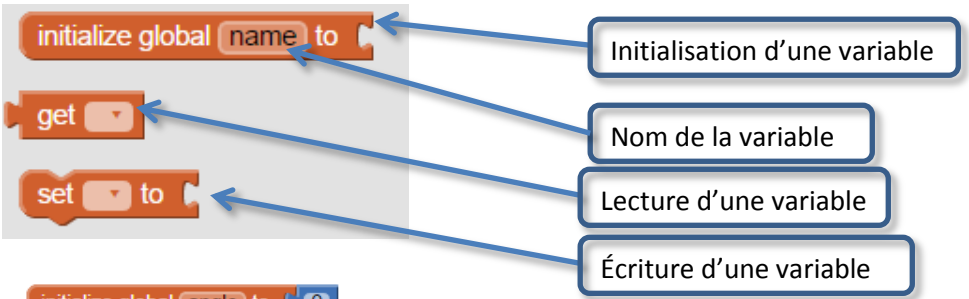
1. Dessinez votre IHM

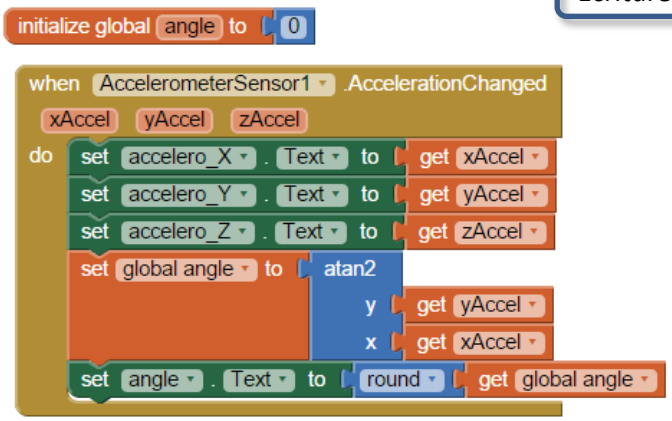


2. Lister les types d'objets
3. Analyser les événements associés à votre application
4. Algorithme
5. Programme sur Appinventor

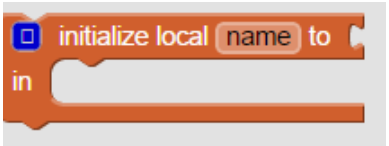
Vous aurez besoin de sauvegarder le résultat de votre calcul dans une variable.

Pour cela vous devrez initialiser cette variable à une valeur donnée et de nommer cette variable.





Ici la variable globale n'a pas de sens. En effet une variable globale est définie si cette variable a besoin d'avoir une portée sur plusieurs évènements, ce qui n'est pas le cas. Il faudra alors modifier l'application pour définir une variable locale à l'évènement clique sur bouton !



13.3 Type de variables

Il existe plusieurs types de variables avec AppInventor.

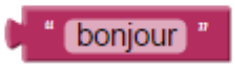
13.3.1 Variables de type nombre

Vous pouvez utiliser des nombres réels, des nombres entiers

13.3.2 Variables de type texte

Vous pourrez utiliser une chaine de caractères. Attention si vous souhaitez ajouter un saut de ligne. Si vous concaténer des chaînes de caractères, il n'y a pas de saut de ligne qui s'insère automatique. Il faudra alors ajouter les caractères suivants en fin de chaîne.

« \n »



13.3.3 Variables de type booléen

13.4 Contrôle de l'axe vertical Z

Afin d'avoir une valeur correcte, il faut positionner le mieux possible le smartphone ou tableau verticalement.

Il faut informer l'utilisateur du bon ou mauvais positionnement de son smartphone. Ne pas afficher la valeur de l'angle si nécessaire.

1. IHM
2. Type des objets
3. Nom des objets
4. Algorithme
5. programme

13.5 Calibration

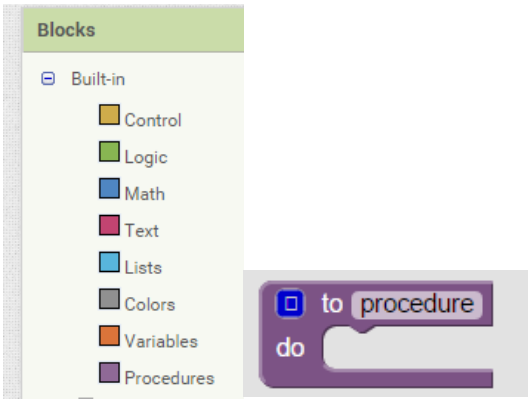
Vous avez constatez que les valeurs n'étaient pas parfaitement calibrées.

Vous pourrez améliorer votre application en faisant une calibration automatique en proposant à l'utilisateur de poser le smartphone sur une surface plane pour calibration et de calibrer les 3 axes. Il faudra alors un montage pour pouvoir positionner de façon fixe le smartphone suivant les 3 axes.

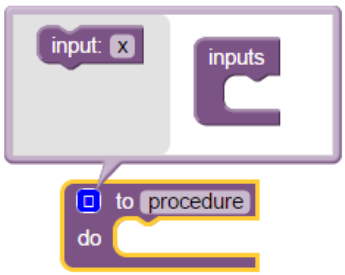
14 Notion de procédure

Pour simplifier l'écriture d'un programme, il nécessaire d'éviter de recopier n fois le même code. Cela permet de limiter la taille d'un programme, cela permet de modifier plus facilement un programme en n'intervenant qu'à un seul endroit, sa lisibilité est bien meilleure.

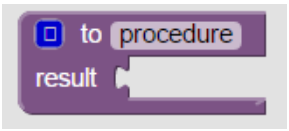
Créer un bloc qui encapsulera votre code.



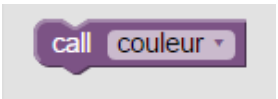
Il est aussi possible de passer des variables d'entrée à cette procédure.



Il est possible de créer une fonction qui renverra alors une donnée.



Il suffira ensuite pour utiliser ce code encapsulé, d'appeler le bloc créé. Une fois la procédure ou fonction créée, il faudra retourner dans le menu blocks/procédure.



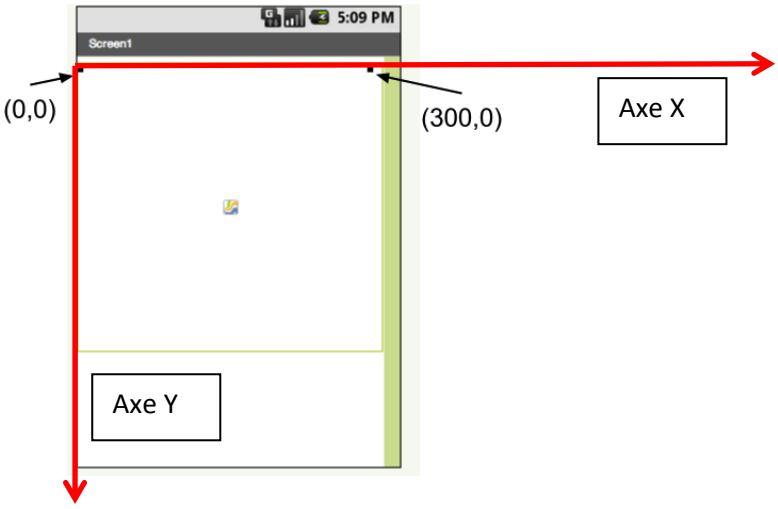
15 Dessiner des formes géométriques

Objectif : permettre la mesure d'un angle à partir du tracé de 2 segments de référence.

15.1 Mesure d'un angle à partir d'un segment prédéfini

Pour simplifier l'application, vous partirez d'un segment de référence qui sera horizontal. L'utilisateur n'aura plus qu'à tracer le deuxième segment et d'en calculer l'angle.

Pour pouvoir tracer des formes géométriques il faut utiliser le composent canvas. Ce composent utilise le repère suivant. (Regardez la documentation associée à ce composant).



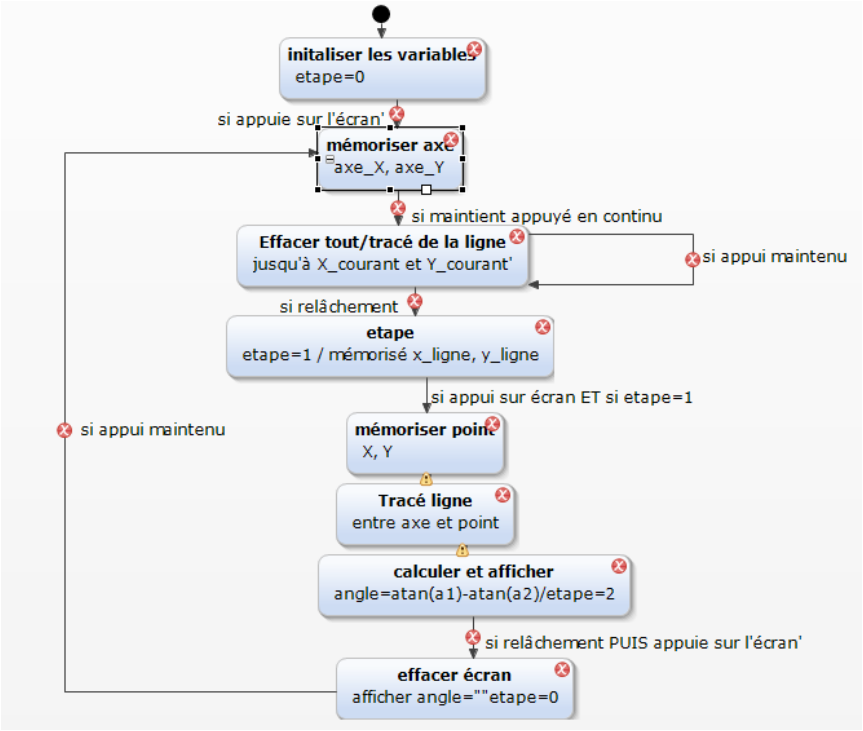
1. dessinez votre IHM
2. faite la liste des objets nécessaires
3. nommer les objets
4. réaliser un diagramme d'état
5. algorithme
6. programme Appiventor
7. simulation
8. génération du fichier APK

15.2 Mesure d'un angle à partir de 2 segments quelconques

Réaliser la même application, mais avec l'ajout du choix du premier segment qui ne sera pas forcément horizontal.

Il faudra pensez à réinitialiser le système pour refaire une nouvelle mesure si on touche à nouveau sur l'écran. Il serait aussi possible de réinitialiser la mesure à partir de l'appui sur un bouton par exemple. On pourra modifier l'affichage d'une information pour indiquer ce que doit faire l'utilisateur en fonction des différentes étapes.

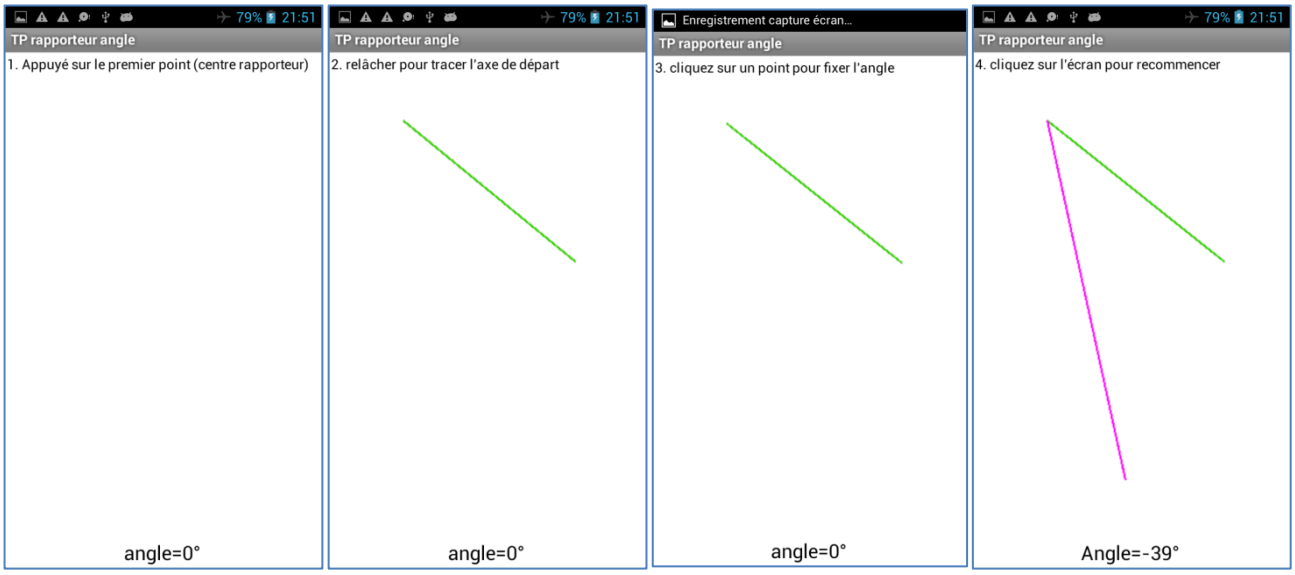
1. dessinez votre IHM
2. faite la liste des objets nécessaires
3. nommer les objets
4. réaliser un diagramme d'état

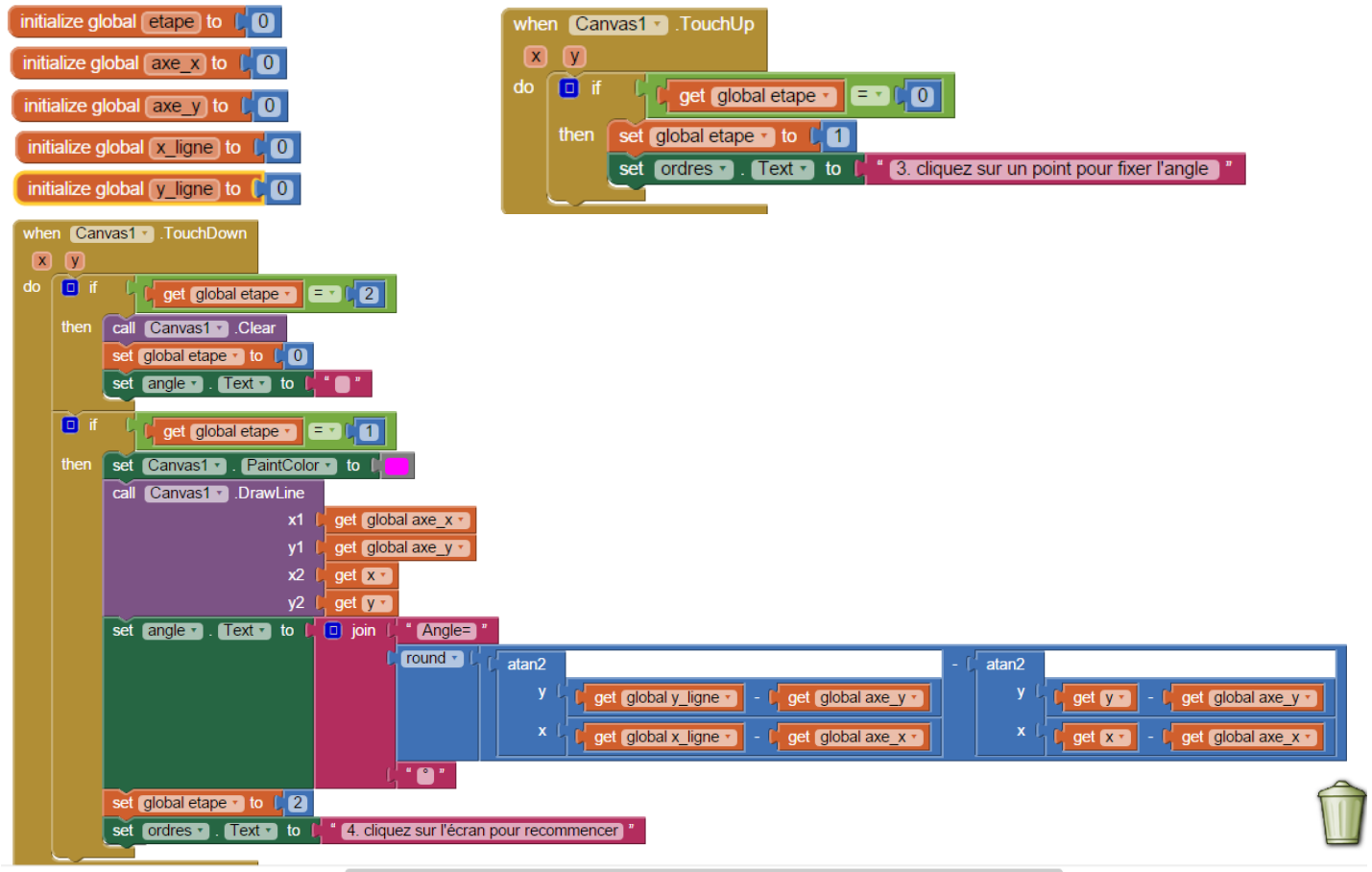


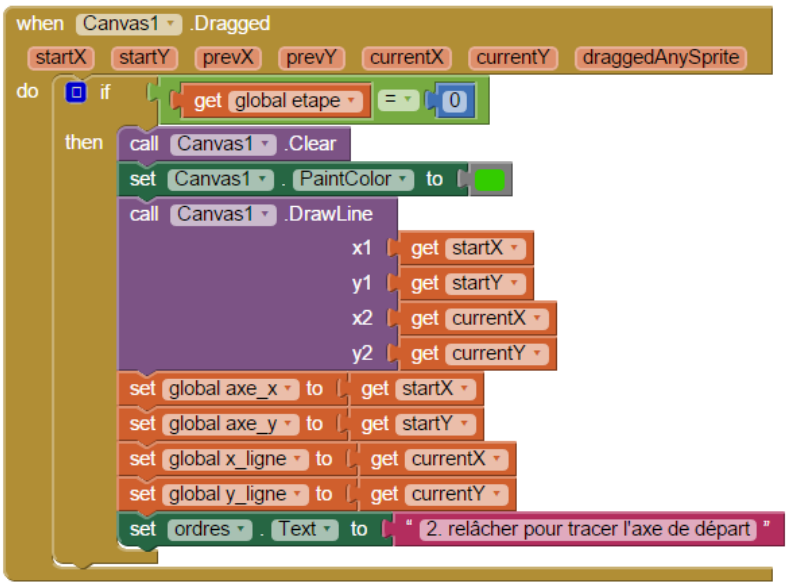
5. algorithme
6. programme Appiventor
7. simulation

Eléments de réponse

IHM







15.3 Horizon artificiel

Vous pourrez aussi réaliser une application qui permet d'afficher sous forme de segment la ligne d'horizon. Ce segment devra passer toujours par le point central du canevas. Il faudra alors calculer l'équation de la droite pour en déduire les coordonnées des 2 points du segment à tracer. Pour cela il faut calculer la pente de la droite grâce aux capteurs. Connaissant, la pente puis le point central, on en déduit les coordonnées des points du segment et on peut alors facilement tracer le segment.

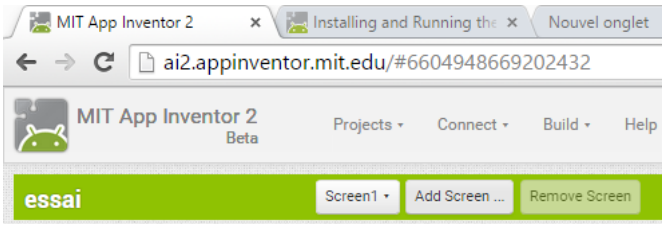
16 Gestion de plusieurs écrans

Il est possible de créer plusieurs écrans pour créer une application qui nécessiterait un menu pour pouvoir gérer plusieurs écrans dans l'application.

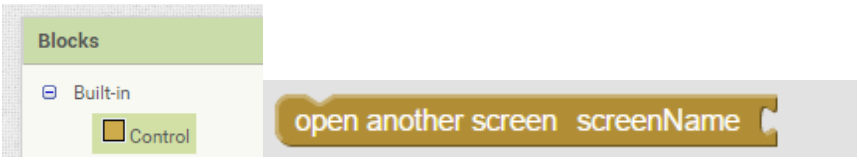
Vous allez réaliser une simple application qui permet de passer de l'écran d'accueil (Sreen1) à un écran de paramétrages (Screen2)

Commencer par créer les différentes IHM (écrans)

Cliquez sur Add Screen



Utilise la commande de contrôle suivante :



Ajouter un texte qui sera le nom de votre écran (attention pas le titre !)

17 Gestion d'une base de données

Sauvegarde d'une donnée en mémoire flash (initialisation)

17.1 Gestion de listes

Créer une application qui va jouer un son à partir d'une liste de sons. Il faudra donc créer et enregistrer une liste de sons.

Vous pourrez rechercher une bibliothèque de sons ici (https://soundation.com/studio) ou utilisez les fichiers sons wav déjà disponibles sur l'ENT.

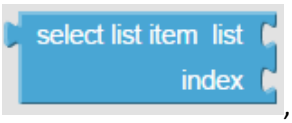
Respecter la méthode pour développer votre application.

Vous pourrez utiliser les sons proposés dans l'ENT dans la partie formation pour vous éviter de perdre du temps.

Il vous faudra créer une variable globale de type liste pour mémoriser les fichiers sons à lancer. Pour cela vous utiliserez la définition d'une liste.

La liste proposée est en fait une matrice à une dimension son(1,n).

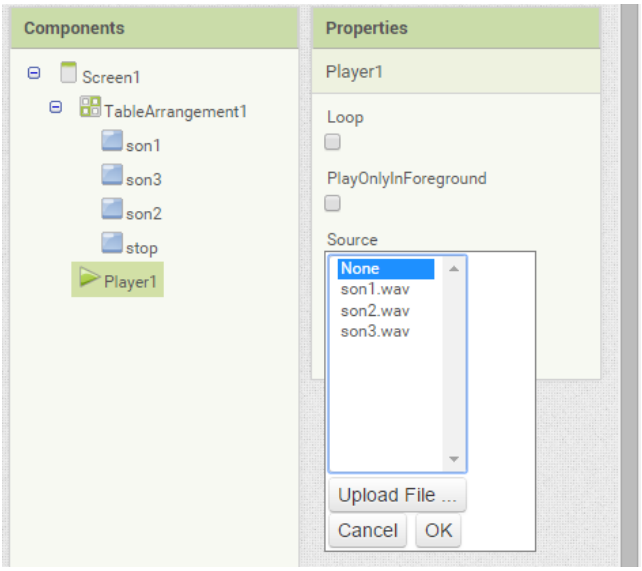
Pour sélectionner l'élément i dans la matrice son(1,i), vous utiliserez la propriété associée :

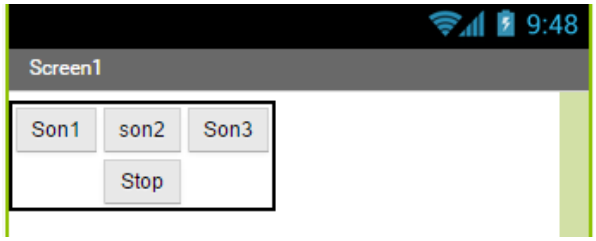


, l'index n permet de sélectionner le nieme item de la liste.

Il faut insérer un objet son, comme le player. Il faudra alors ajouter dans la partie de l'IHM dans les propriétés du player les différents fichiers wav en les téléchargeant dans l'application, ainsi, ces fichiers wav pourront être sélectionnés dans la partie programmation dans une liste.

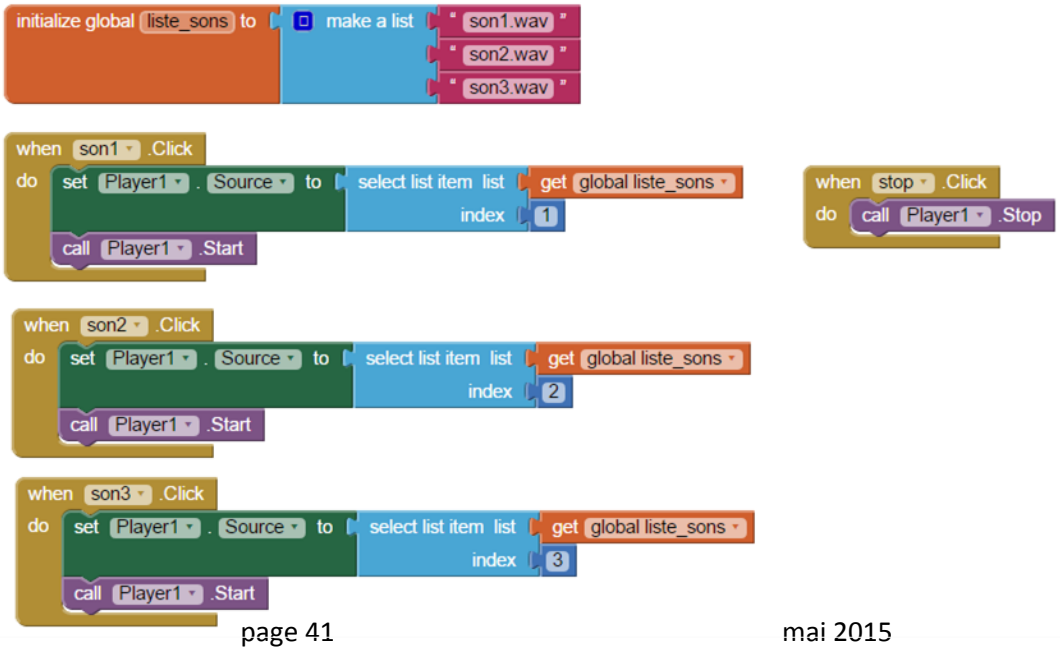
Pour cela il faudra cliquer sur les propriétés du player et cliquer dans la partie source sur Upload.





Exemple d'IHM

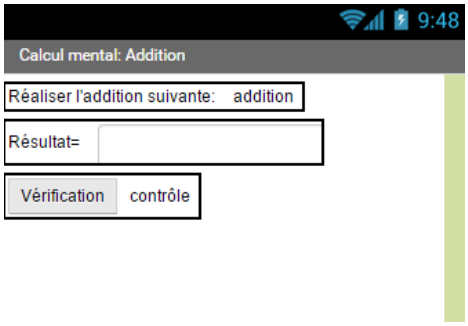
Eléments de correction



17.2 Calcul mental

Créer une application qui permettra aux élèves de s'entrainer au calcul mental. Il s'agit dans un premier temps de concevoir une application qui vérifie des additions de nombres entiers. Il s'agira alors de positionner dans une liste, les nombres à additionner. L'utilisateur tapera sa réponse. L'application vérifiera la réponse.

L'IHM pourra ressembler à :



Pour cette application il faudra utiliser une boucle itérative finie.

Pour cela vous pouvez utiliser l'une des 2 solutions suivantes

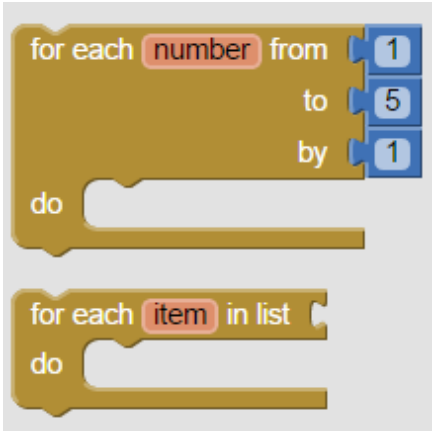
Le pseudo code associé est :

POUR nombre ALLANT de 1 à 5 pas pas de 1 FAIRE

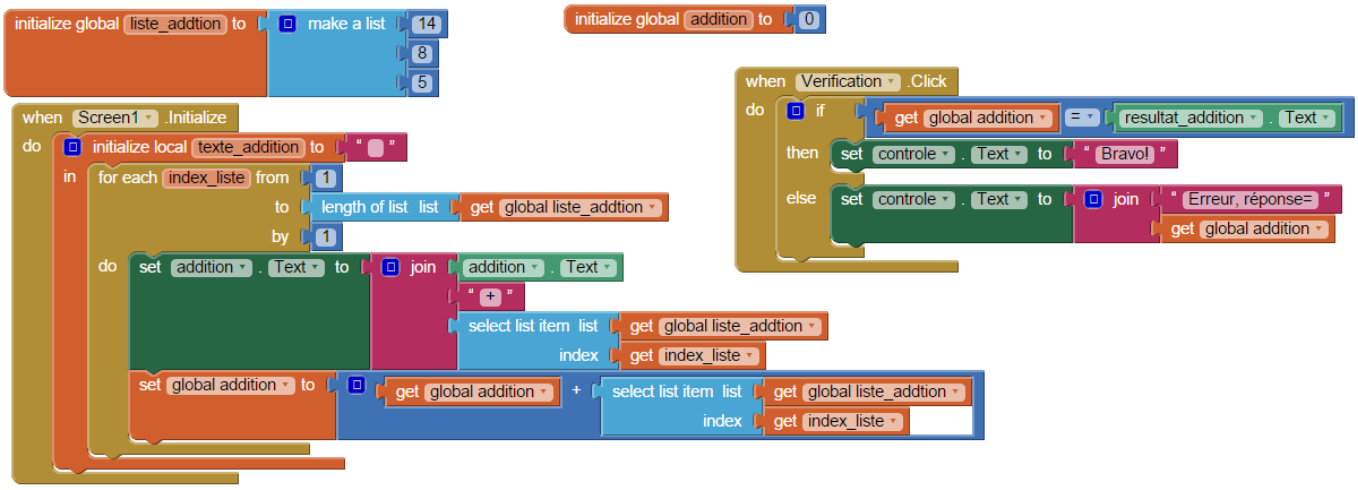
{

…….

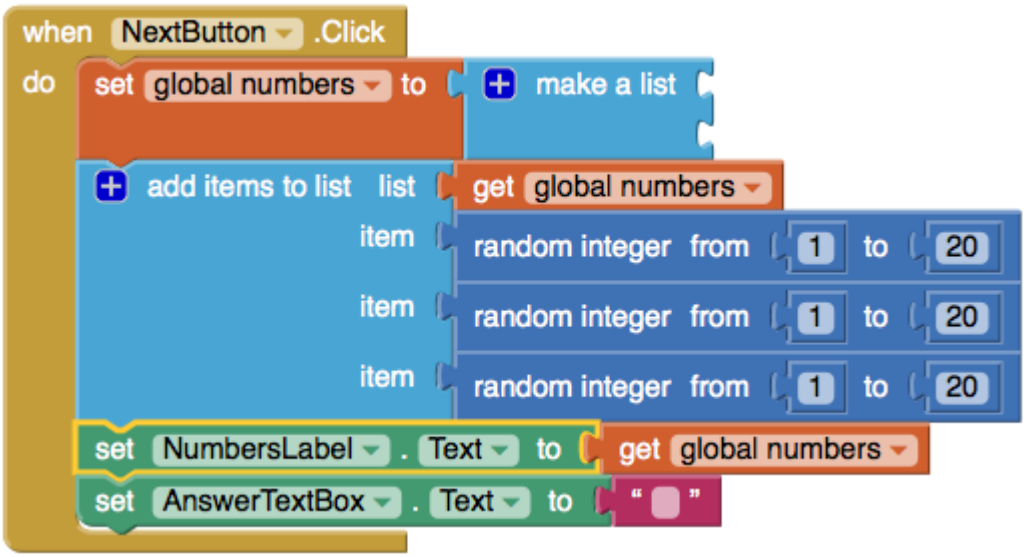
}

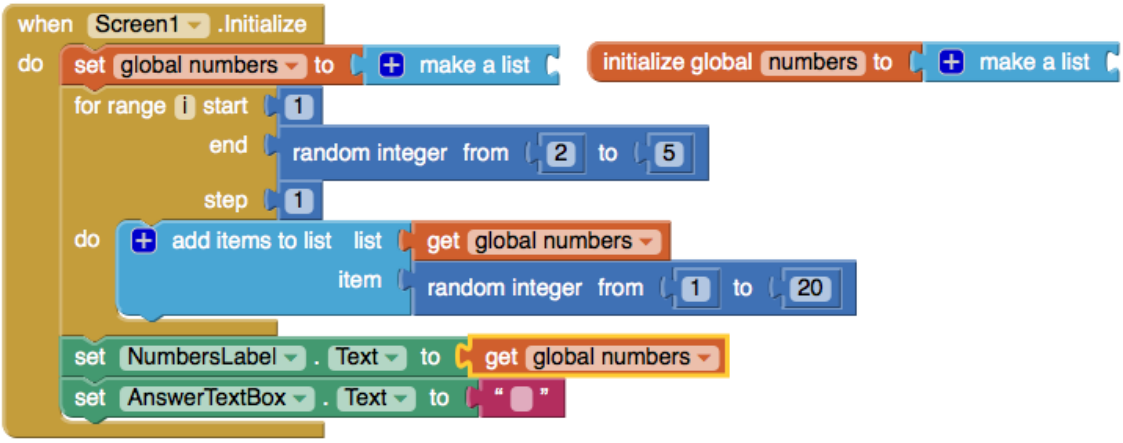


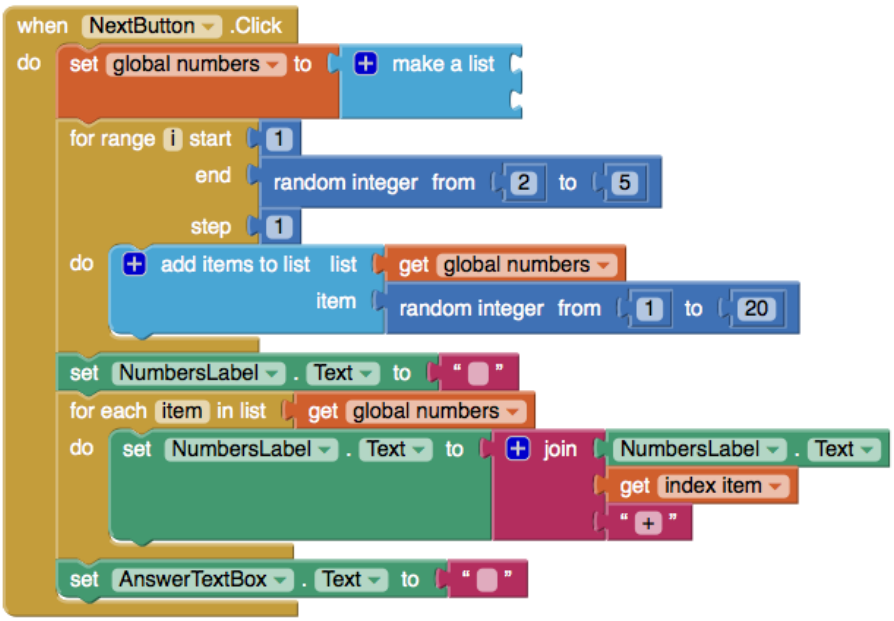
Elément de correction :

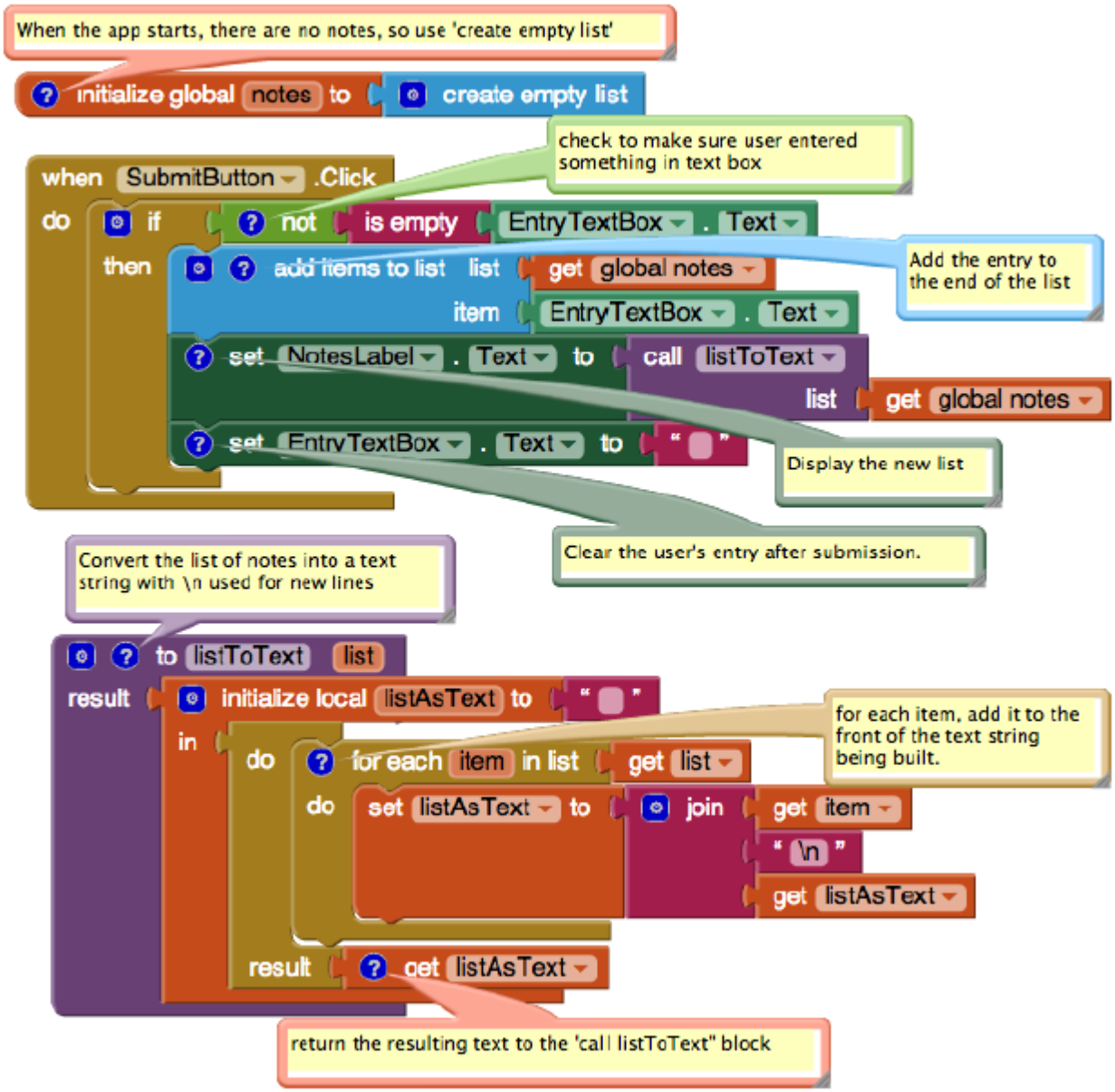


Vous pourrez améliorer l'application pour qu'elle puisse proposer un calcul basé sur des nombres aléatoires.


when NextButton .Click
do set global numbers to make a list
add items to list list get global numbers
item random integer from 1 to 20
item random integer from 1 to 20
item random integer from 1 to 20
set NumbersLabel . Text to get global numbers
set AnswerTextBox . Text to


when Screen1 .Initialize
do set global numbers to make a list
initialize global numbers to make a list
for range i start 1
end random integer from 2 to 5
step 1
do add items to list list get global numbers
item random integer from 1 to 20
set NumbersLabel . Text to get global numbers
set AnswerTextBox . Text to


when NextButton .Click
do set global numbers to make a list
for range i start 1
end random integer from 2 to 5
step 1
do add items to list list get global numbers
item random integer from 1 to 20
set NumbersLabel . Text to
for each item in list get global numbers
do set NumbersLabel . Text to join NumbersLabel . Text
get index item
+
set AnswerTextBox . Text to



17.3 Gestion d'une base de données

Il est possible de sauvegarder des données soit dans la mémoire flash du smartphone, soit sur un espace de stockage sur Internet.

Ainsi, il sera possible de converser des informations, même après avoir éteint et allumer son smartphone.

Vous allez réaliser une application qui permet aux élèves de stocker plusieurs résultats de mesures sur un smartphone.

17.3.1 Sauvegarde des données dans la mémoire Flash

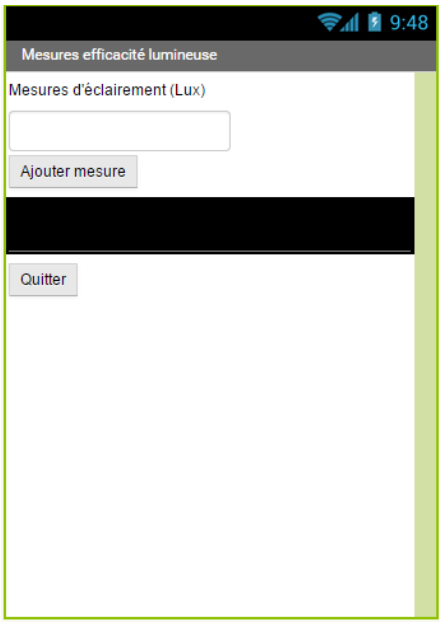
Chaque donnée est attachée à une étiquette. Pour mémoriser un élément de données, vous spécifierez l'étiquette sous laquelle elle devra être stocké sous ou rattachée à. Par la suite, vous pouvez récupérer les données qui ont été stockées en spécifiant l'étiquette donnée.

Dans la même optique, il serait possible de sauvegarder les résultats d'un jeu, le paramétrage d'une application, etc…

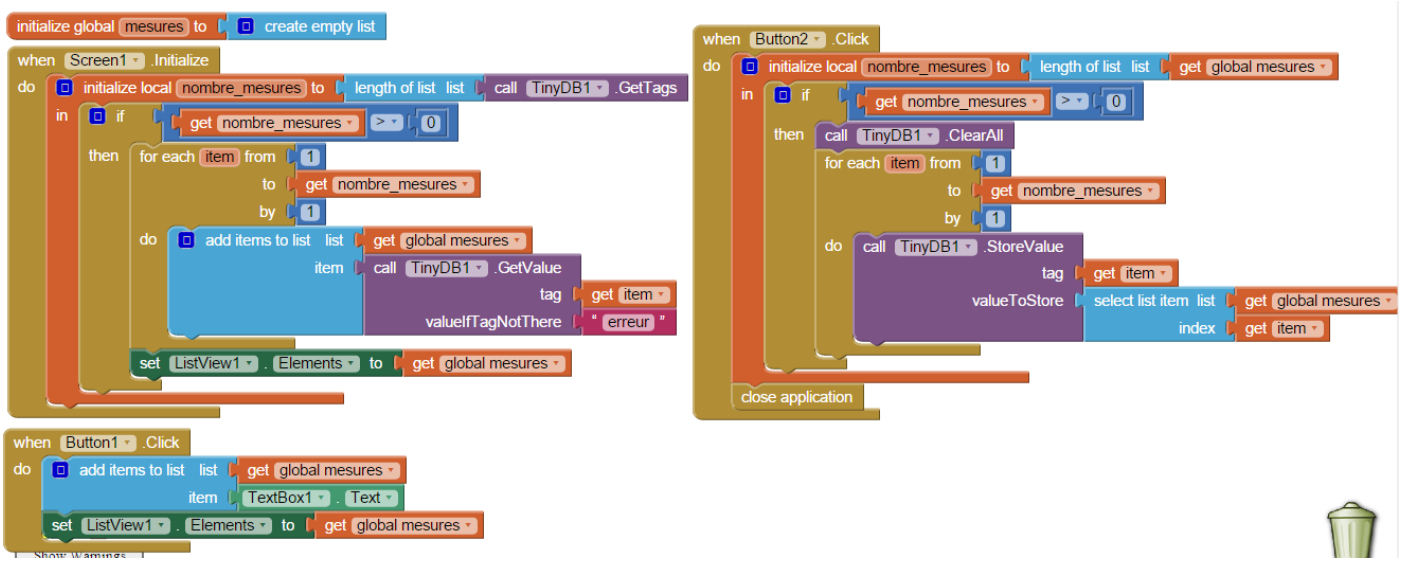
On se propose de réaliser une application qui sauvegarde des mesures d éclairement d'une lampe donnée.

Ces mesures seront sauvegardées dans le smartphone, même si on éteint ce dernier.

Au lancement de l'application, il faudra donc afficher les mesures déjà sauvegardées, puis proposer à l'utilisateur de pouvoir ajouter à ses sauvegardes une nouvelle mesure. On n'oubliera pas de créer un bouton qui permettra en fin d'application de sauvegarder la base de données dans la mémoire flash et de quitter l'application.



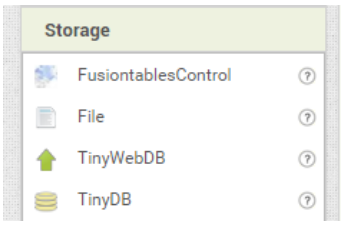
Correction du programme:



Vous pourrez améliorer l'application en ajoutant une fonction qui permet de supprimer un item sélectionné et un bouton qui permet de réinitialiser la base de données complète pour effacer toutes les valeurs.

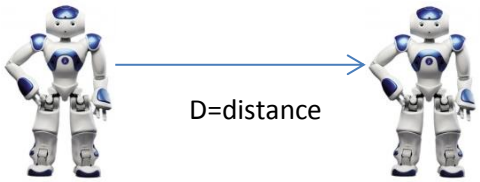
17.3.2 Sauvegarde des données sur Internet

Vous utiliserez l'outil TinyWebDB



18 Mise en œuvre d'un timer

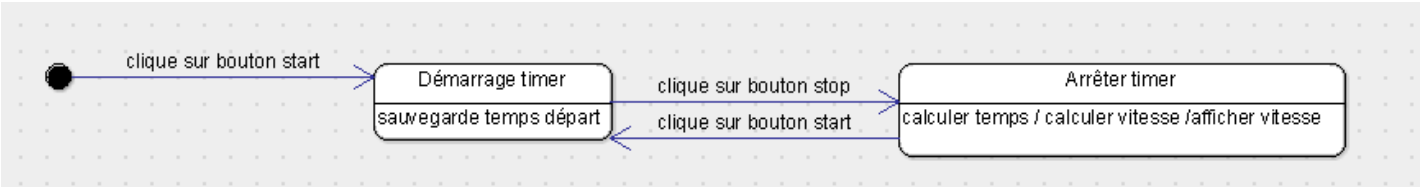
Certaines applications demandent d'être synchronisée dans le temps ou de connaître une chronologie.



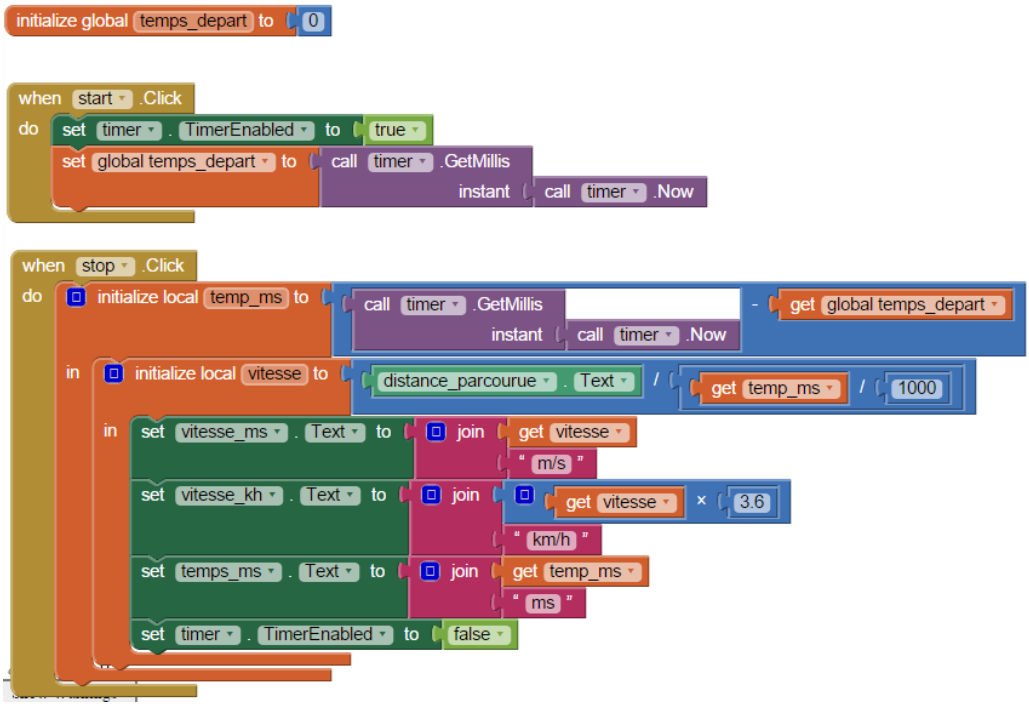
Pour cela Appinventor met à disposition un outil timer.

Vous allez réaliser une application qui permet de mesurer la vitesse de déplacement d'un robot.

Pour cela vous aurez tracé au sol deux lignes dont vous avez mesuré la distance qui les sépare. Pour mesurer la vitesse moyenne de déplacement, il suffira que le robot se déplace entre ces 2 lignes et de démarrer un chronomètre dès que le robot a croisé la première ligne et d'arrêter le chronomètre dès qu'il a croisé la deuxième ligne. A l'aide de l'expression v=d/t, vous calculerez la vitesse moyenne de déplacement en m/s et km/h du robot.



Elément de correction



Vous pourrez améliorer l'application en utilisant l'un des capteurs suivants :

- Capteur NFC avec pastille associée
- Capteur de proximité
- Accéléromètre

Ainsi la mesure se fera automatiquement. On pourra encore améliorer le système réaliser plusieurs mesures et en les sauvegardant dans la mémoire flash, puis en les affichant sur un graphique et en calculant la moyenne et l'écart type par exemple.

19 Communication de données via Bluetooth

19.1 Méthode d'analyse

Analyse d'un cahier des charges

Information d'entrée et de sortie

IHM, nommage des objets, recherche de leurs propriétés

Diagramme de cas d'utilisation

Diagramme d'état

Analyse des évènements associés à des tâches

Algorithme

Programmation codage, type de variables, recherche de documentation sur un objet, propriété ou méthode

Simulation

Compilation

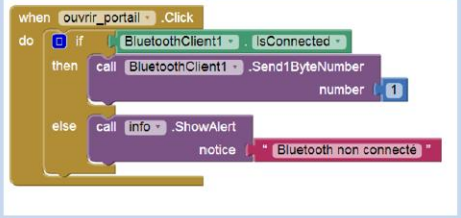
Téléversement

19.2 Application pilotage d'un portail via liaison Bluetooth

Cette application permet de piloter la commande d'ouverture et de fermeture d'un portail coulissant.

Le système va envoyer un octet au portail. En fonction de l'octet reçu par le portail, il y aura ouverture ou fermeture.

Design et codage de l'application avec App Inventor



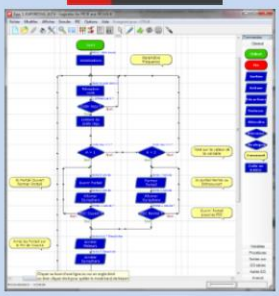
Programmation Smartphone

ANDROID

Bluetooth

Programmation de l'interface AutoProg ou AutoProgUno

PICAXE



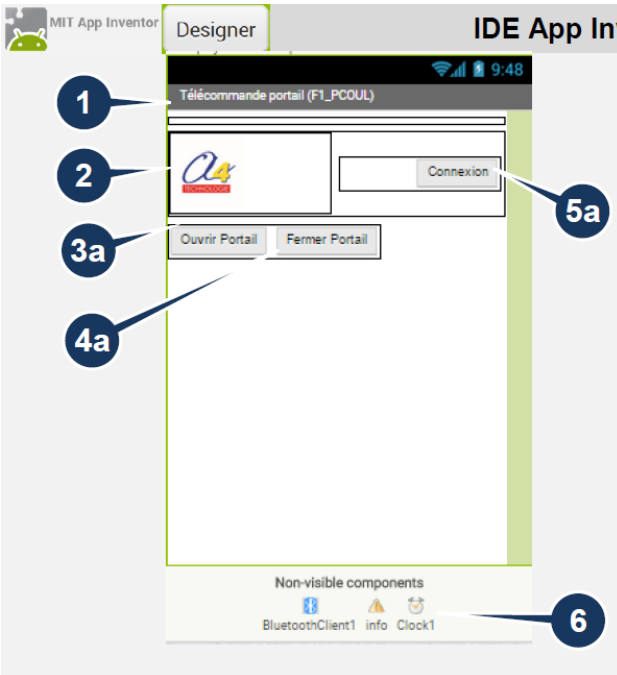
ARDUINO

Programmation Interface

AutoProg

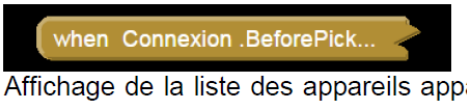
AutoProgUno

IDE App Inventor2 (fenêtre Designer)



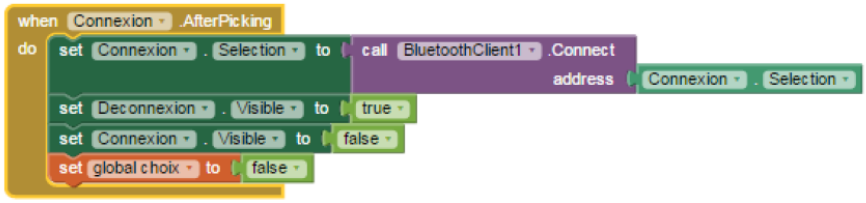
1. Titre de l'application.
2. Image insérée dans l'application.
3a. Déclenchement de l'ouverture en envoyant le code **« 1 ».**
4a. Déclenchement de la fermeture en envoyant le code **« 2 ».**
5a. Gestion de la communication en Bluetooth.
6. Éléments non visibles de l'application. Des explications sont données en Annexe.

Affichage des appareils appairés



Affichage de la liste des appareils appairés avec le smartphone.

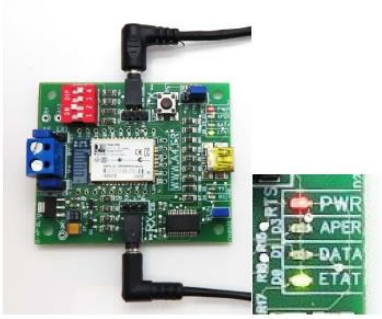
Sélection de l'appareil à connecter au smartphone



Établissement de la connexion avec l'appareil sélectionné et affichage du bouton **Déconnexion**.

Mise en fonctionnement de la carte Bluetooth.

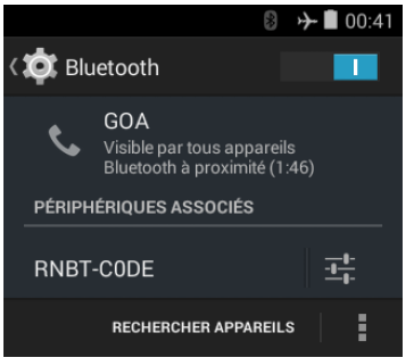
1



Configurer le module Bluetooth comme indiqué au chapitre 3 et le connecter à l'interface AutoProg ou AutoProgUno en respectant le plan de câblage proposé au chapitre 4.

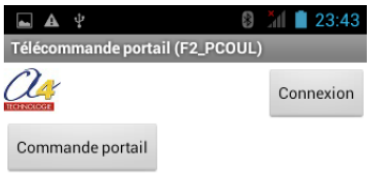
À la mise sous tension du module (au travers de l'interface AutoProg ou AutoProgUno), le témoin rouge **PWR** est allumé et le témoin vert **ETAT** clignote.

2



Activer le Bluetooth et cliquer sur **Rechercher Appareils**.
Sélectionner le module Bluetooth du portail coulissant pour procéder à l'appairage.

3



Lancer l'application puis appuyer sur le bouton **Connexion** en haut à droite de votre écran.

4

Sélectionner l'identifiant du module Bluetooth auquel il faut connecter le smartphone.
L'identifiant correspond à l'inscription **MAC ID** que l'on peut lire sur le composant Bluetooth qui équipe le module.

Le témoin vert **ETAT** cesse de clignoter et reste allumé en permanence.
Le témoin vert **APER** indique que la connexion entre le smartphone et le module Bluetooth est effectuée.

4bis

Si ce message d'erreur apparaît, vérifier que le module est sous tension.
Quitter éventuellement l'application, appuyer pendant 5 s sur le bouton **RESET** de la carte Bluetooth, redémarrer le smartphone.
Répéter l'opération de connexion à partir de l'étape N°3.

5

Lorsque la connexion est réalisée, le bouton **Déconnexion** apparaît.
Le témoin vert **DATA** s'allume dès qu'une donnée est émise ou reçue par le module Bluetooth.
L'appui sur le bouton d'envoi de données, dans cet exemple **Commande portail**, déclenche l'allumage fugitif de ce témoin.

Il faudra aussi faire un programme au niveau du système de réception du portail pour piloter le portail à partir des commandes reçues par le module Bluetooth.

Vous pouvez utiliser différents microcontrôleurs comme un système Tinsy, Arduino, Picaxe, Microchip, etc…

Pour programmer la cible, il est possible d'utiliser des IDE graphiques comme Ardublock qui s'intègre à l'IDE arduino très facilement. La programmation se fait alors avec la même méthode que celle vue actuellement, puisqu'Ardublock utilise la base Blockly.

Exemple de codage pour gérer la réception d'un octet dans une cible microcontrôleur type Arduino.

Ce programme permet de tester un octet reçu sur la carte Bleutooth vec Ardublock.

Ici l'algorithme est assez simple :

TANT QUE vrai

SI (nombre d'octets reçus > 0) ALORS

Lire l'octet reçu le sauvegarder dans reception_bluetooth

SI le code de reçu (reception_bluetooth) = 1 ALORS

Allumer la Led du port n°13

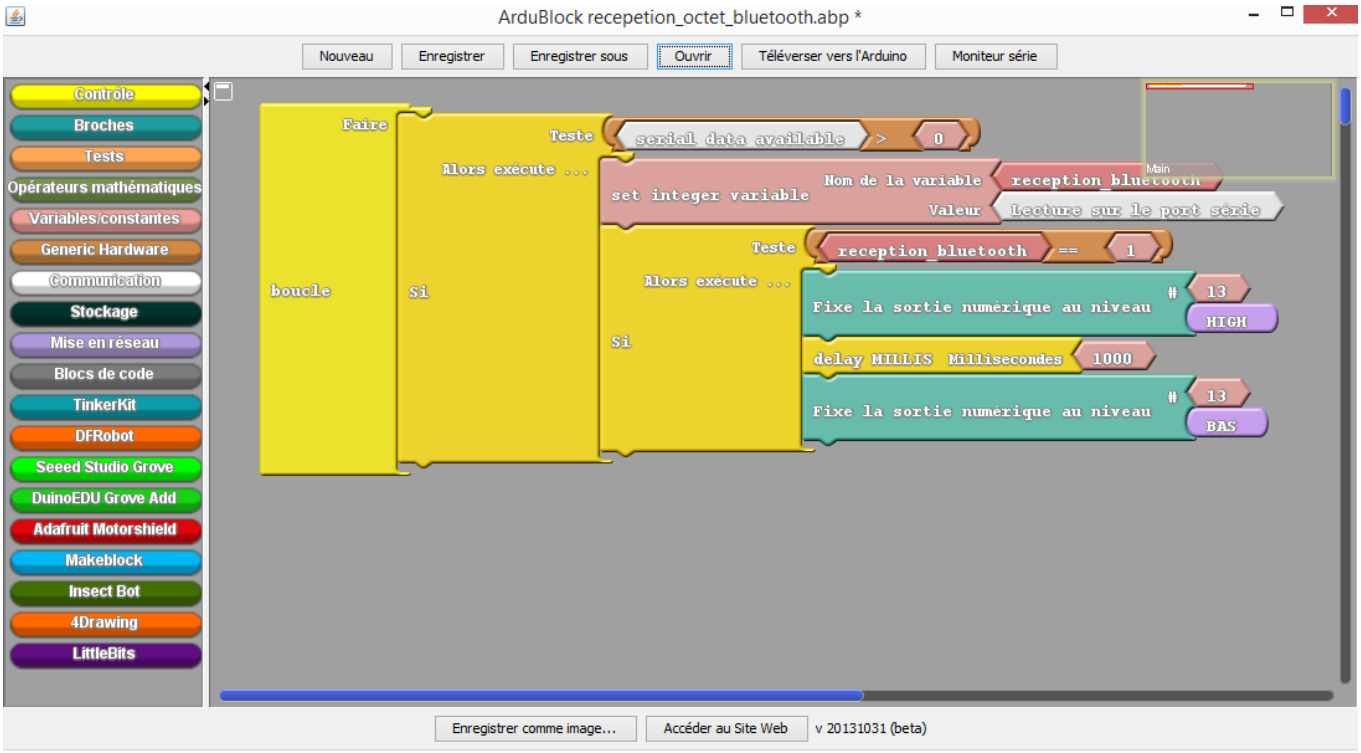
Attendre 1000ms

Eteindre la Led du port n°13

FIN SI

FIN SI

FIN TANQUE



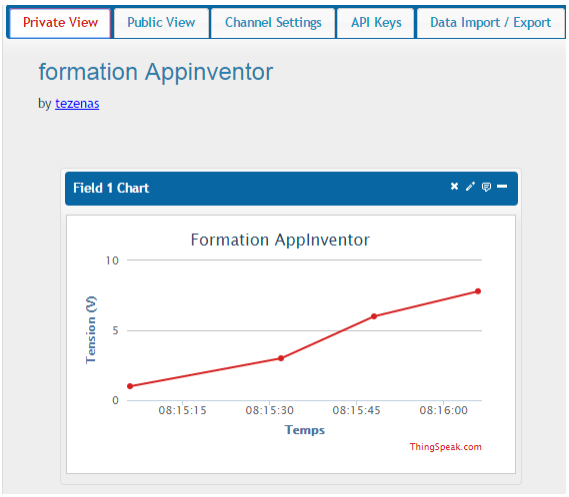
20 Accès au Web

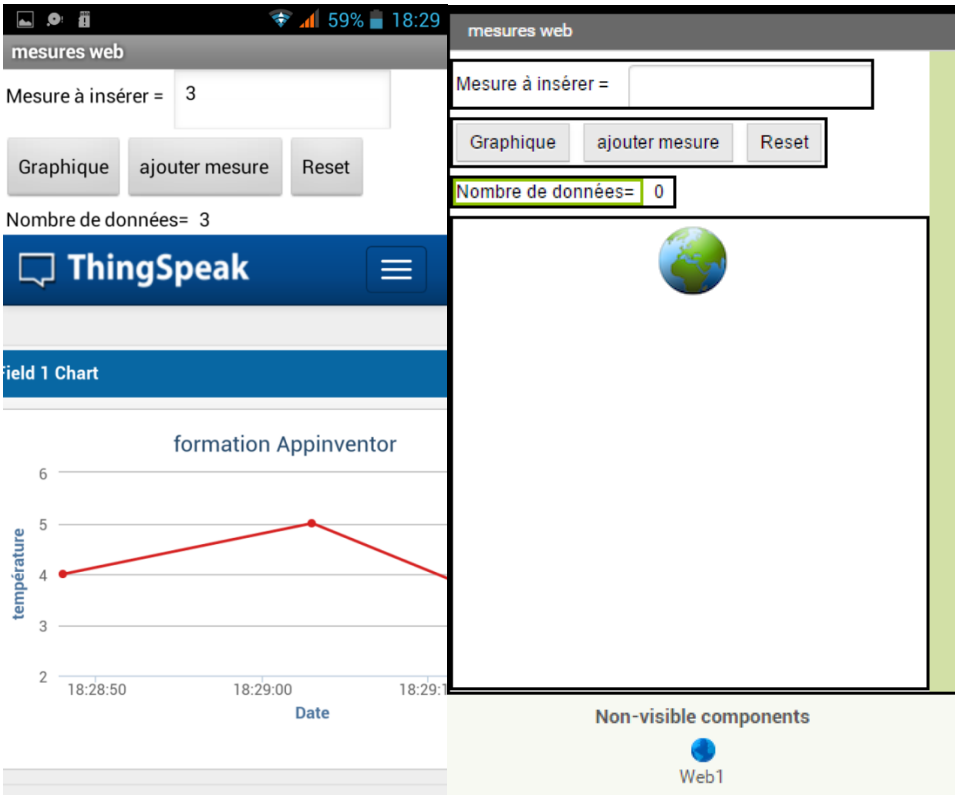
20.1 Graphique sur site web

Il est possible de dialoguer entre un smartphone et un serveur web via Wifi.

L'application suivante permet de sauvegarder des mesures et de les visualiser sur un graphique.

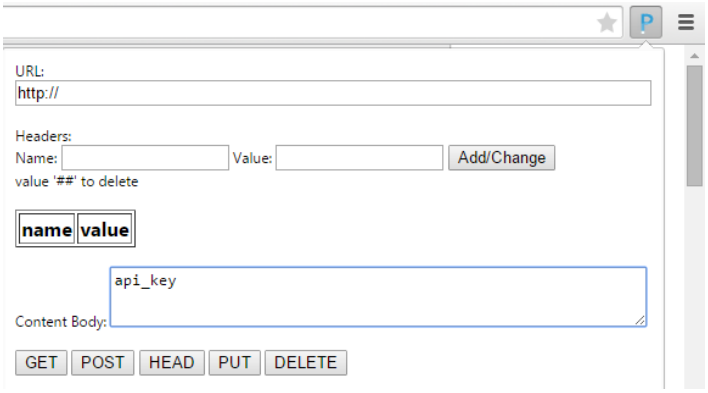
Ici on utilise une application Web très pratique proposée par le site thingspeak.com très performant.





On utilise les requêtes http de type GET et DELETE pour accéder au site Thingspeak et envoyer les données au serveur.

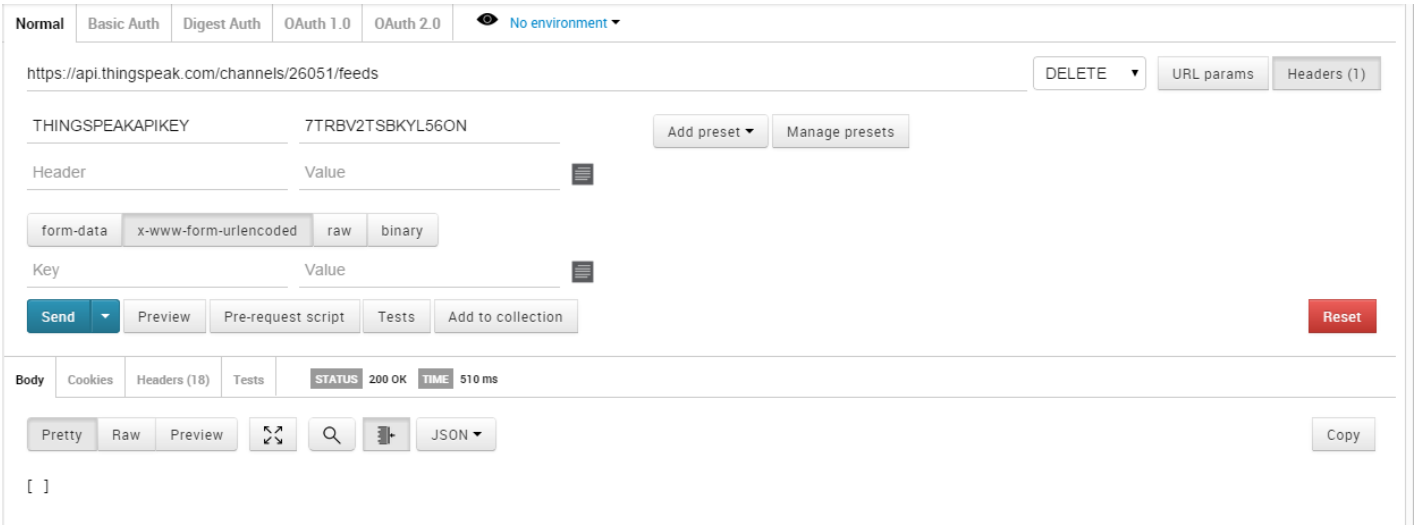
On peut utiliser une application (Chrome Poster) très pratique pour envoyer des requêtes http précises.



Il est possible de faire des essais de requête avec l'extension chrome que l'on peut télécharger ici :

https://chrome.google.com/webstore/detail/postman-rest-client-packa/fhbjgbiflinjbdggehcddcbncdddomop?hl=en

Et en saisissant les paramètres suivants pour envoyer par exemple une demande de suppression des données.



On trouve la documentation des détails de la requête à concevoir ici.

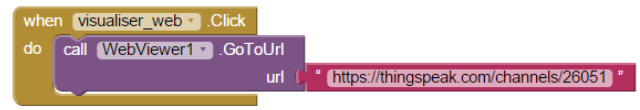
https://thingspeak.com/docs/channels#clear

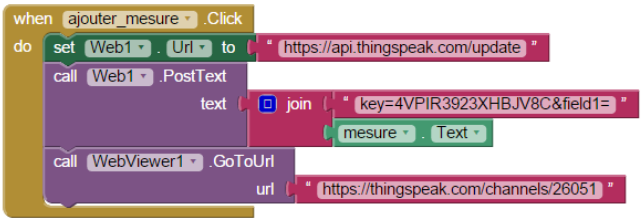
https://thingspeak.com/docs#headers

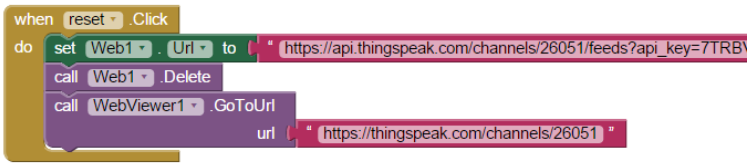
On remarque au passage que la requête DELETE proposée renvoie un coderéponse de 200 (ici status).

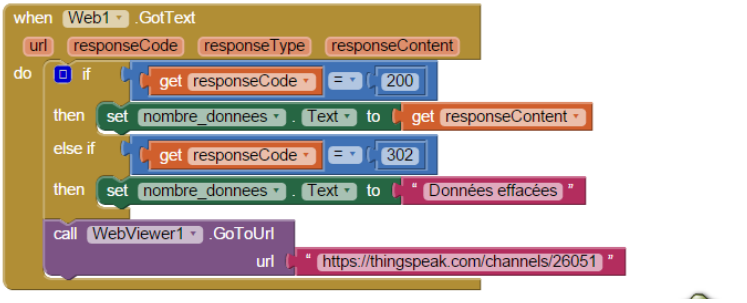
On peut de la même manière analyser les autres requêtes et vérifier les coderéponse de chacune des requêtes. Cela permettra de trier les réponses reçues.

Eléments de réponses









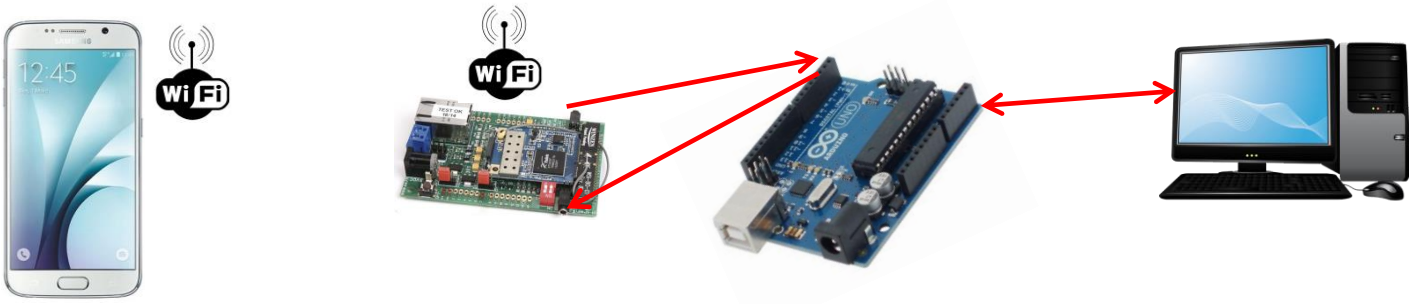
20.2 Accès au Web

20.2.1 Serveur web

Appinventor permet d'utiliser des requêtes http pour pouvoir interagir avec un serveur web.

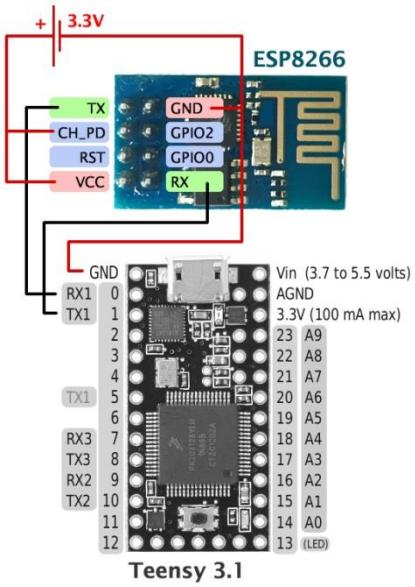
Pour cela il est possible d'utiliser différentes requêtes comme les requêtes GET, POST, PUT, DELETE.

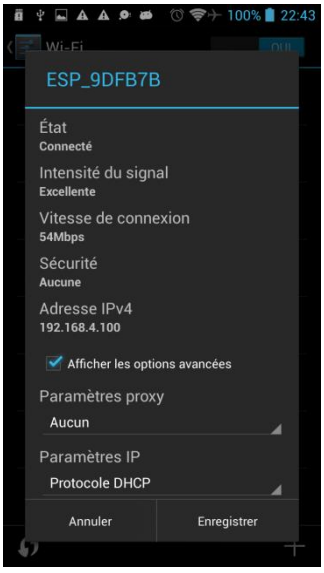
L'application proposée permet de dialoguer et d'échanger des données entre un système embarqué (carte wifi + µC)



IHM du projet. On peut modifier à loisir l'adresse IP de la carte wifi côté µC.

Exemple de connexion entre un µC (ici Teensy compatible Arduino) et un module ESP8266.

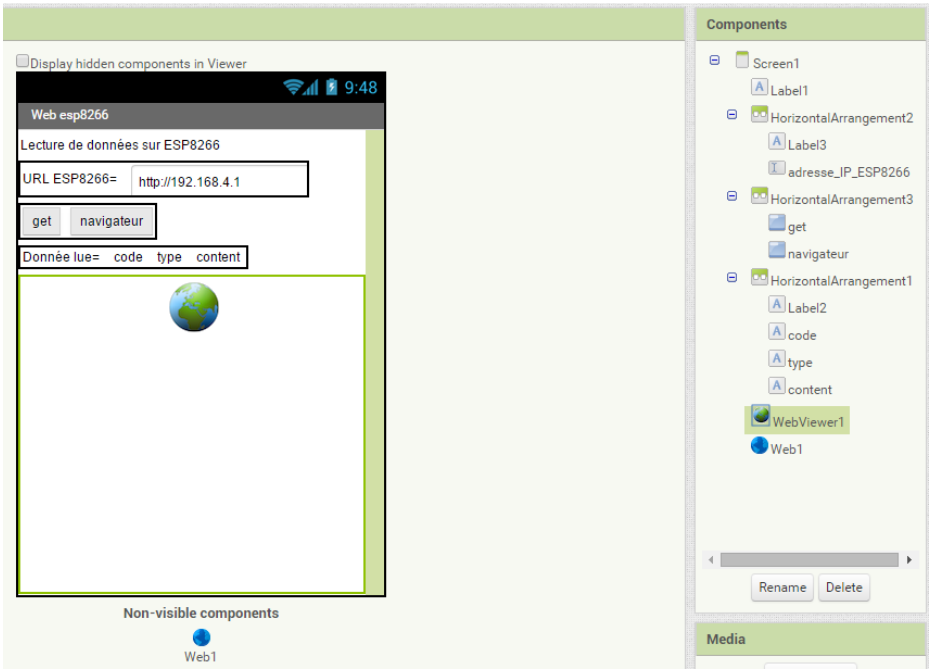




Il sera possible de passer en IP fixe sur le smartphone si nécessaire. Mais le smartphone est ici en DHCP, une adresse IP lui est attribuée, ici 192.168.4.100. On vérifiera bien qu'elle se trouve bien dans la même classe d'adresse que l'ESP8266.

Il est aussi possible de modifier l'adresse IP fixe du module ESP8266à l'aide de la commande AT : AT+CIPSTA=ip

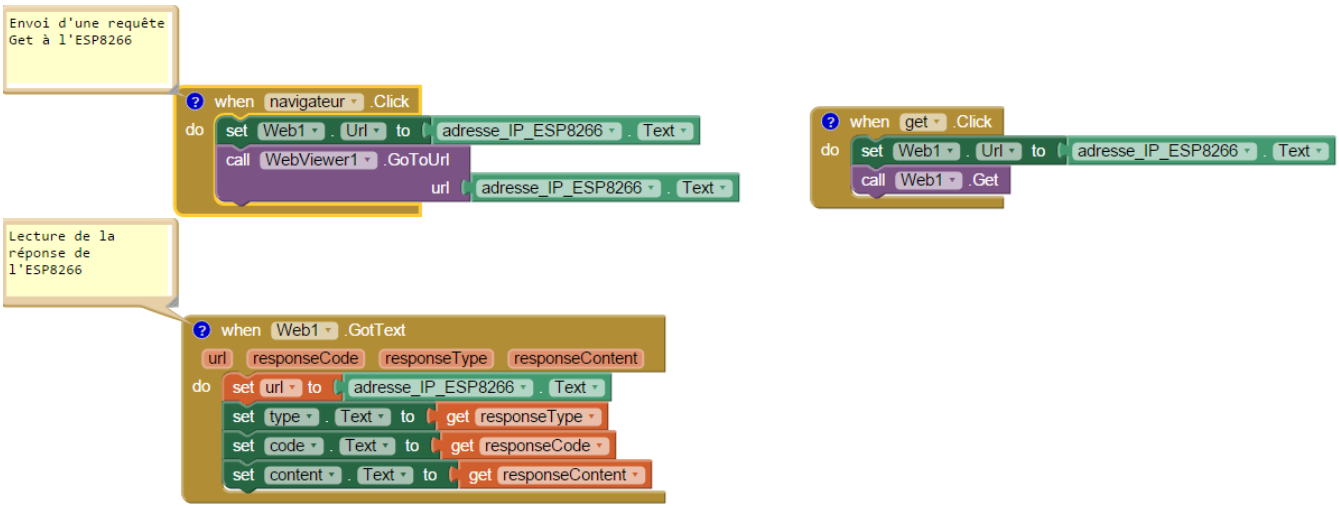
Le bouton GET permet d'envoyer une simple requête GET. Les informations reçues en retour sont affichées dans code/type/content. Le bouton Navigateur envoie, lui aussi, une commande GET, mais avec l'affichage de la page web reçue dans le navigateur intégré à l'application Android.



Lorsqu'une requête est envoyée, soit par une commande GET pure, soit en tapant l'adresse d'un site web, le smartphone va envoyer une requête http GET.

Dans cet exemple, le module ESP8266 est utilisé comme module d'interface Wifi avec le µC. On prendra le soin de connecter le TX de l'ESP8266 sur la pin3 et le RX de l'ESP8266 sur le pin2 et non sur le pin 0 et 1 ! En effet, nous allons réserver ces pins pour la communication entre le µC Arduino et le moniteur série Arduino.

Programme sur Appinventor.



```
/*
Application test pour le module wifi ESP8266 connecté à un smartphone programmé avec APPinventor
carte UNO Arduino
*/
#include <SoftwareSerial.h>
#define pinEN   9
#define ESP_Rx 2
#define ESP_Tx 3
#define LED 13
#define DEBUG true //permet de


SoftwareSerial wifi(ESP_Rx,ESP_Tx);

void print_debug(String data)
{
  Serial.print(data);
}
void setup()
{
  pinMode(LED, OUTPUT);
  digitalWrite(LED,HIGH);
  delay(500);
  digitalWrite(LED,LOW);
  Serial.begin(9600);
  wifi.begin(9600);
  Serial.println("Test ESP8226 avec smartphone");
  //initialisation du module wifi, envoi des commandes AT
  envoi_commande_AT("AT+RST\r\n",2000,DEBUG);  //on reset le module wifi et ses config précédentes
  envoi_commande_AT("AT+CWMODE=2\r\n",1000,DEBUG); //configuration en mode point d'accès
  envoi_commande_AT("AT+CIFSR\r\n",1000,DEBUG); //on affiche l'adresse IP du module wifi ESP8266
  envoi_commande_AT("AT+CIPMUX=1\r\n",1000,DEBUG); //configuration en mode multi connexion
  envoi_commande_AT("AT+CIPSERVER=1,80\r\n",1000,DEBUG);  //on se met en mode serveur port 80
}

int a=0; // compteur à chaque commande GET
void loop()
{
   if(wifi.available()) //Si on reçoit une requête
   {
     Serial.println("demande entrante");
     if(wifi.find("+IPD,")) //si le serveur web à reçu une demande de page web
      {
        Serial.println("Page web reconnue");
        delay(2000);
        int connectionId = wifi.read()-48; //on est en mode multiconnexion, il faut donc connaitre le canal
        Serial.println("id page="+String(connectionId)); //on affiche le n° de canal
        digitalWrite(LED,HIGH);

        String pagewebhtml;
        String commande_AT_envoi;
        pagewebhtml="HTTP/1.1 200 OK\n\n"; //envoi du code/type au smartphone
        String cipSend = "AT+CIPSEND=";
        cipSend += connectionId;
        cipSend += ",";
        cipSend +=pagewebhtml.length();
        cipSend +="\r\n";

        envoi_commande_AT(cipSend,3000,DEBUG);
        delay(20);
        envoi_commande_AT(pagewebhtml,3000,DEBUG);

        pagewebhtml="Test Wifi ESP8266 A"+String(a); //envoi de la page web très basique
        cipSend = "AT+CIPSEND=";
        cipSend += connectionId;
        cipSend += ",";
        cipSend +=pagewebhtml.length();
        cipSend +="\r\n";
```

```
        envoi_commande_AT(cipSend,3000,DEBUG);
        delay(20);
        envoi_commande_AT(pagewebhtml,3000,DEBUG);
        delay(20);

        cipSend = "AT+CIPCLOSE=";
        cipSend += connectionId;
        cipSend += ",";
        cipSend +=pagewebhtml.length();
        cipSend +="\r\n";

        envoi_commande_AT(cipSend,3000,DEBUG);
        delay(20);
      }
      a+=1;
   }
}


String envoi_commande_AT(String commande, const int timeout, boolean debug)
{
    String reponse="";
    wifi.print(commande); //envoi de la commande
    long int time=millis();
    while((time+timeout)>millis())//on boucle sur la durée voulue pour une attente de réponse
    {
       while (wifi.available())
       {
         //on lit la donnee envoyée par le module wifi pour debug
         char c=wifi.read();
         reponse += c;
       }
    }
    if (debug) // si debugage activé
    {
      Serial.print(reponse); //on affiche dans le moniteur série la réponse de l'ESP8266
    }
    return reponse;
}
```

Résultats obtenus sur le moniteur série d'Arduino.

```
Test ESP8226 avec smartphone
AT+RST

OK
bB×†@ùRcâüR‰š")ÊNÿÊÉ¤□ÌÿS‰‡
[Vendor:www.ai-thinker.com Version:0.9.2.4]

ready
AT+CWMODE=2
no change
AT+CIFSR
192.168.4.1

OK
AT+CIPMUX=1

OK
AT+CIPSERVER=1,80

OK
```

Initialisation

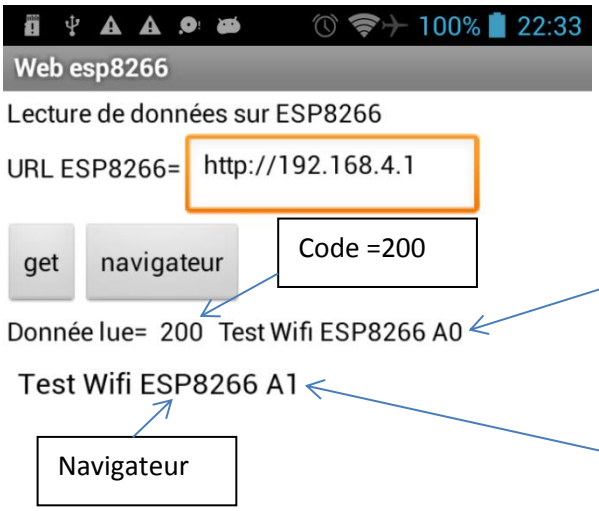
Défilement automatique

```
demande entrante
Page web reconnue
id page=0
,169:GET / HTTP/1.1
User-Agent: Dalvik/1.6.0 (Linux; U; AndroA> HTTP/1.1 200 OK
SEND OK
AT+CIPSEND=0,20
> Test Wifi ESP8266 A0
SEND OK
AT+CIPCLOSE=0,20

OK
Unlink
```

Réponse à l'appui sur le bouton GET

```
demande entrante
Page web reconnue
id page=0
,527:GET / HTTP/1.1
Host: 192.168.4.1
Connection: keep-aliveA> HTTP/1.1 200 OK
SEND OK
AT+CIPSEND=0,20
> Test Wifi ESP8266 A1
SEND OK
AT+CIPCLOSE=0,20

OK
Unlink
```

Réponse à l'appui sur le bouton navigateur



On remarque bien que la valeur du compteur a à bien été incrémenter entre l'appui sur le bouton GET et le bouton Navigateur. On passe de 0 à 1

Wireshark permet de sniffer le protocole http et tcp et de vérifier l'envoi et la réception d'informations autour de la commande GET.

```
4 0.004571000 192.168.4.52      192.168.4.1      HTTP      439 GET / HTTP/1.1
```

```
⊞ Frame 4: 439 bytes on wire (3512 bits), 439 bytes captured (3512 bits) on interface 0
⊞ Ethernet II, Src: HonHaiPr_81:e8:a8 (08:3e:8e:81:e8:a8), Dst: 1a:fe:34:9d:fb:7b (1a:fe:34:9d:fb:7b)
⊞ Internet Protocol Version 4, Src: 192.168.4.52 (192.168.4.52), Dst: 192.168.4.1 (192.168.4.1)
⊞ Transmission Control Protocol, Src Port: 57714 (57714), Dst Port: http (80), Seq: 1, Ack: 1, Len: 385
⊟ Hypertext Transfer Protocol
  ⊞ GET / HTTP/1.1\r\n
    Host: 192.168.4.1\r\n
    Connection: keep-alive\r\n
    Cache-Control: max-age=0\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\n
    User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/43.0.2357.65 Safari/537.36\r\n
    Accept-Encoding: gzip, deflate, sdch\r\n
    Accept-Language: fr-FR,fr;q=0.8,en-US;q=0.6,en;q=0.4\r\n
    \r\n
    [Full request URI: http://192.168.4.1/]
```

Avec utilisation de chrome comme navigateur en tapant d'adresse ip de l'ESP8266 (192.168.4.1)

```
11 11.21597500(192.168.4.1      192.168.4.52      TCP      74 [TCP segment of a reassembled PDU]
```

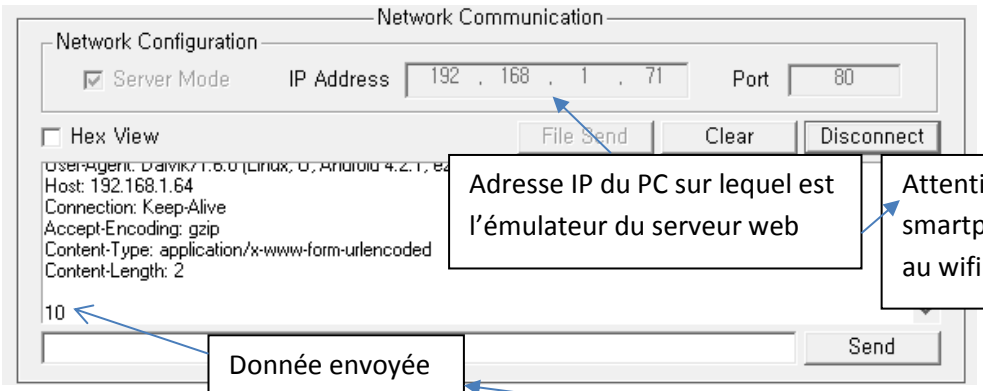
```
⊟ Transmission Control Protocol, Src Port: http (80), Dst Port: 57714 (57714), Seq: 18, Ack: 386, Len: 20
    Source port: http (80)
    Destination port: 57714 (57714)
    [Stream index: 0]
    Sequence number: 18    (relative sequence number)
    [Next sequence number: 38    (relative sequence number)]
    Acknowledgment number: 386    (relative ack number)
    Header length: 20 bytes
  ⊞ Flags: 0x018 (PSH, ACK)
    Window size value: 5455
    [Calculated window size: 5455]
    [Window size scaling factor: -2 (no window scaling used)]
  ⊞ Checksum: 0xe39d [validation disabled]
  ⊞ [SEQ/ACK analysis]
    TCP segment data (20 bytes)

0010  00 3c 00 2c 00 00 ff 06  32 0a c0 a8 04 01 c0 a8   .<.,.... 2.......
0020  04 34 00 50 e1 72 00 00  7d 93 0d d7 c6 4a 50 18   .4.P.r.. }....JP.
0030  15 4f e3 9d 00 00 54 65  73 74 20 57 69 66 69 20   .O....Te st Wifi
0040  45 53 50 38 32 36 36 20  41 34                    ESP8266  A4

TCP segment data (tcp.data), 20 bytes    Packets: 14 Displayed: 14 Marked: 0 Dropped: 0
```

Réponse renvoyée par le µC via le module ESP8266 au navigateur du PC (192.168.4.52)

20.2.2 Envoi d'un donnée sur le Web

Il est possible d'ajouter l'envoi d'une donnée du smartphone au module µC.

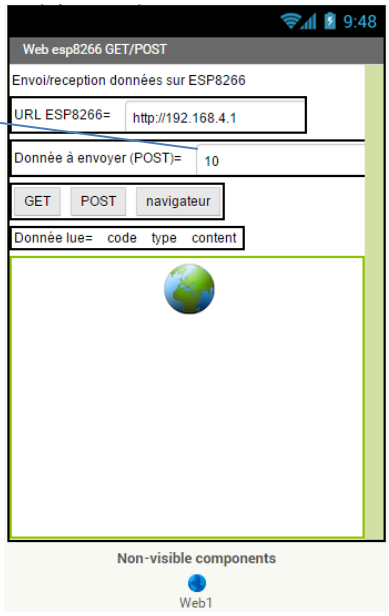
On utilise ici un émulateur de serveur web comme avec Device Terminal ou networkstuff sur son PC sans utiliser l'ESP8266



Adresse IP du PC sur lequel est l'émulateur du serveur web

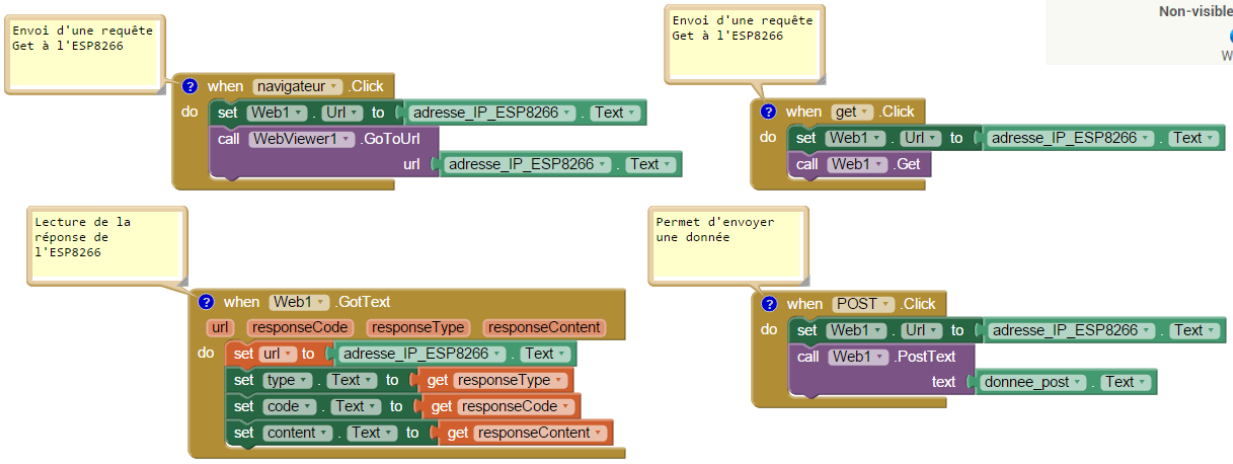
Attention de bien mettre dans l'appli du smartphone cette adresse et de s'associé au wifi du PC !

Donnée envoyée



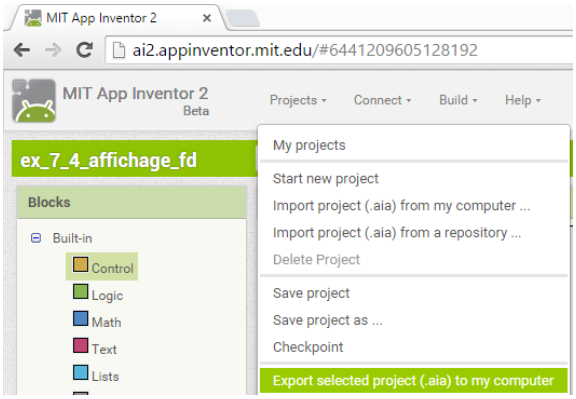

```
Link

+IPD,1,240:POST / HTTP/1.1
User-Agent: Dalvik/1.6.0 (Linux; U; Android 4.2.1; e2001_v89_jbl1a698 Build/JOP40D)
Host: 192.168.4.1
Connection: Keep-Alive
Accept-Encoding: gzip
Content-Type: application/x-www-form-urlencoded
Content-Length: 2

10
OK
```

Infos reçue à l'envoi du POST. Visualisation issue du moniteur série d'Arduino avec l'ESP8266



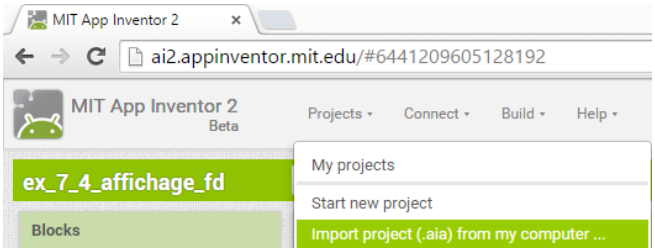
21 Sauvegarde des projets Appinventor dans un fichier

Il est possible d'enregistrer le projet sauvegardé sur les serveurs du MIT. Pour cela cliquez sur



L'extension du fichier créé est aia.

De la même manière, il est possible d'importer un projet Appinventor



22 Ressources

22.1 Informations générales

http://appinventor.mit.edu/explore/ai2/tutorials.html

Tester une application Android APK sur son PC

https://developer.chrome.com/apps/getstarted_arc

http://www.bluestacks.com/

Documentation sur tous les blocs d'AppInventor

http://appinventor.mit.edu/explore/ai2/support/blocks.html

Tous les types d'objets d'Appinventor

http://ai2.appinventor.mit.edu/reference/components/

Installation AppInventor en français sur un PC

https://sites.google.com/site/appincarthage/

22.2 Visualiser écran Android sur PC

http://www.mymobiler.com/(Ne fonctionne pas sur windows 8 ?)

http://www.mightypocket.com/2010/09/installing-android-screenshots-screen-capture-screen-cast-for-windows/

22.3 Logiciel diagramme d'état

http://statecharts.org/download.html

22.4 Initiation

https://interstices.info/upload/docs/application/pdf/2014-06/csunplugged2014-fr.pdf

22.5 Programmation

http://studio.code.org/s/frozen/stage/1/puzzle/1

22.6 Algorithme

http://studio.code.org/s/course1/stage/6/puzzle/1

22.7 Installation d'Appinventor en local

http://sourceforge.net/projects/ai2u/

lancer votre navigateur et tapez localhost :8888 dans votre navigateur. Vous pouvez utiliser ce serveur à partir d'autres postes en tapant l'adresse ip du poste sur lequel est installé ce serveur sans oublier le port 8888.

23 Notions d'apprentissages :

Evènement

Un événement est une action sur un objet capable de déclencher une tâche

Objet

Propriété

Méthode

Classe (clonage)

Algorithme

Programme

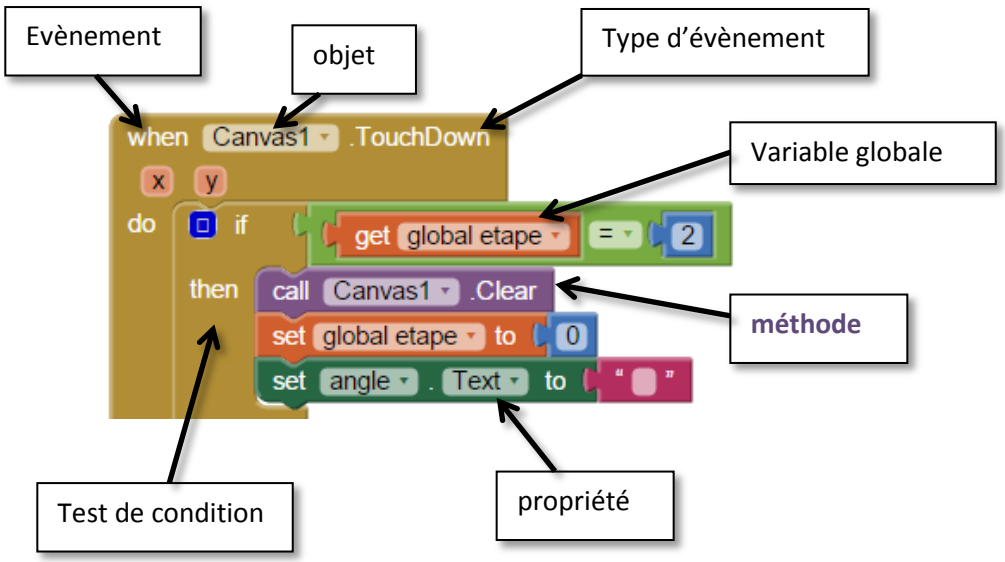
Compilation

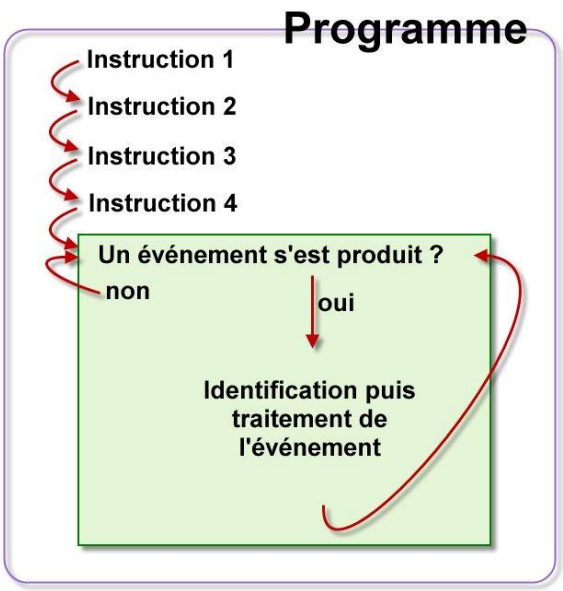
Diagramme d'état

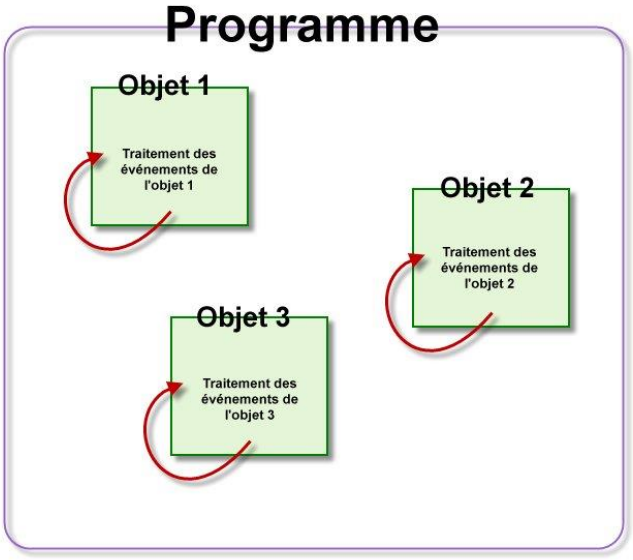
Tâche

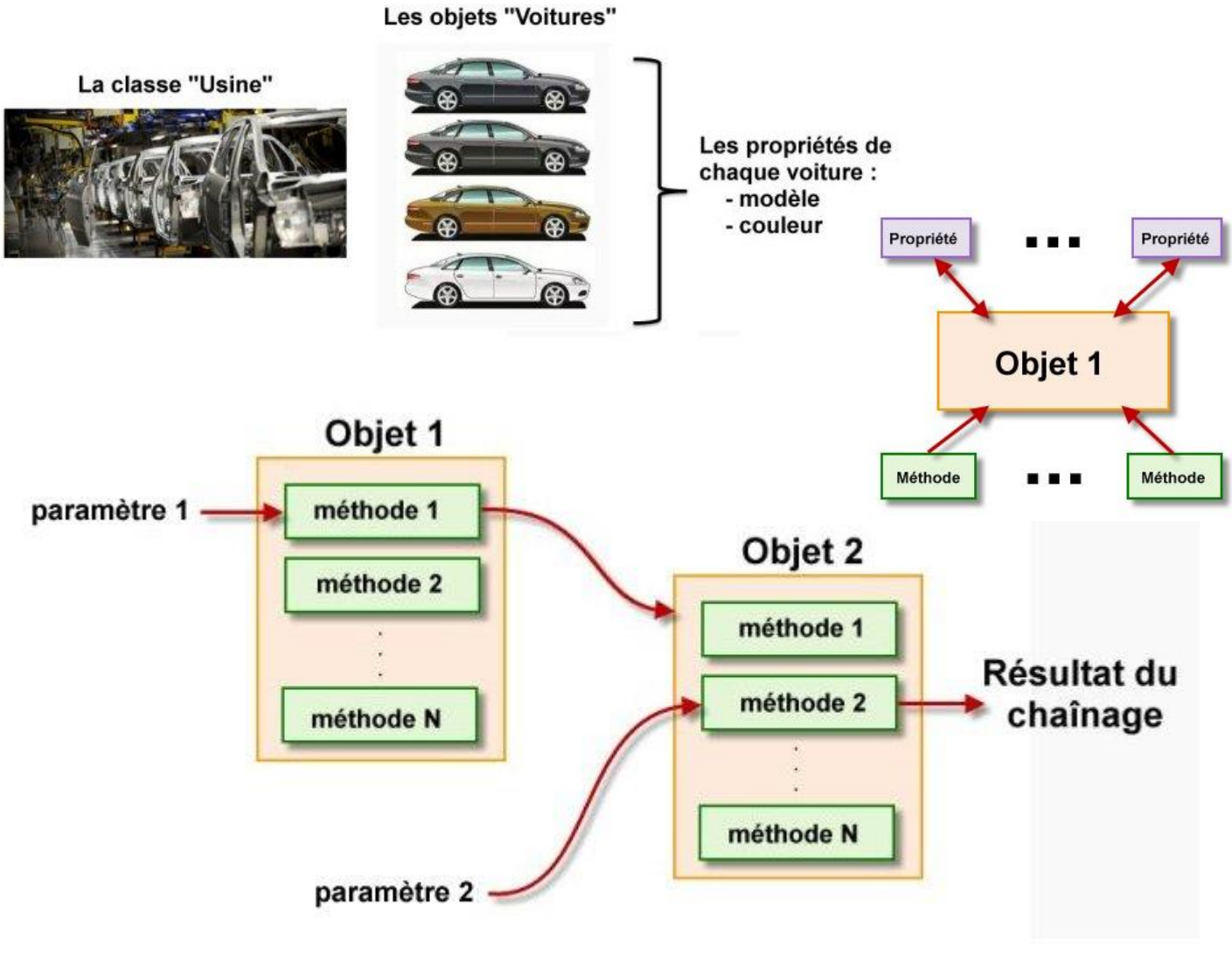
Fonction, procédure

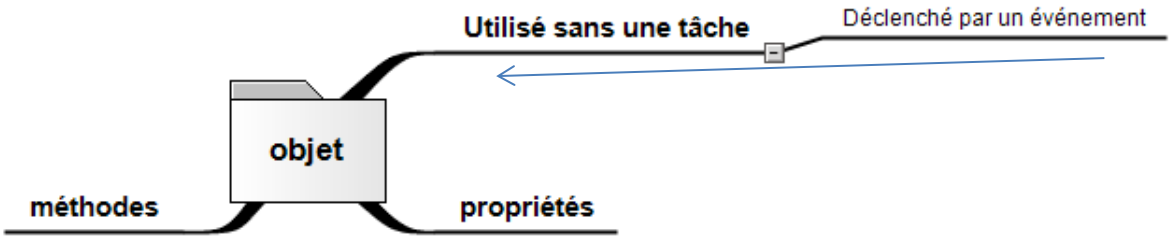
Types de variables




Programme
Instruction 1
Instruction 2
Instruction 3
Instruction 4
Un événement s'est produit ?
non
oui
Identification puis traitement de l'événement


Programme
Objet 1
Traitement des événements de l'objet 1
Objet 2
Traitement des événements de l'objet 2
Objet 3
Traitement des événements de l'objet 3


Les objets "Voitures"
La classe "Usine"
Les propriétés de chaque voiture :
- modèle
- couleur
Propriété
Propriété
Objet 1
Méthode
Méthode
Objet 1
paramètre 1
méthode 1
méthode 2
méthode N
Objet 2
méthode 1
méthode 2
méthode N
Résultat du chaînage
paramètre 2



Classe (description d'un ensemble d'objets, c'est un modèle d'objet) / objet (entité virtuelle ou réelle qui partagent les mêmes caractéristiques, un objet est associé à une classe)/ états (étape de la vie d'un objet) / tâches

On peut se rapprocher de certains diagrammes UML.

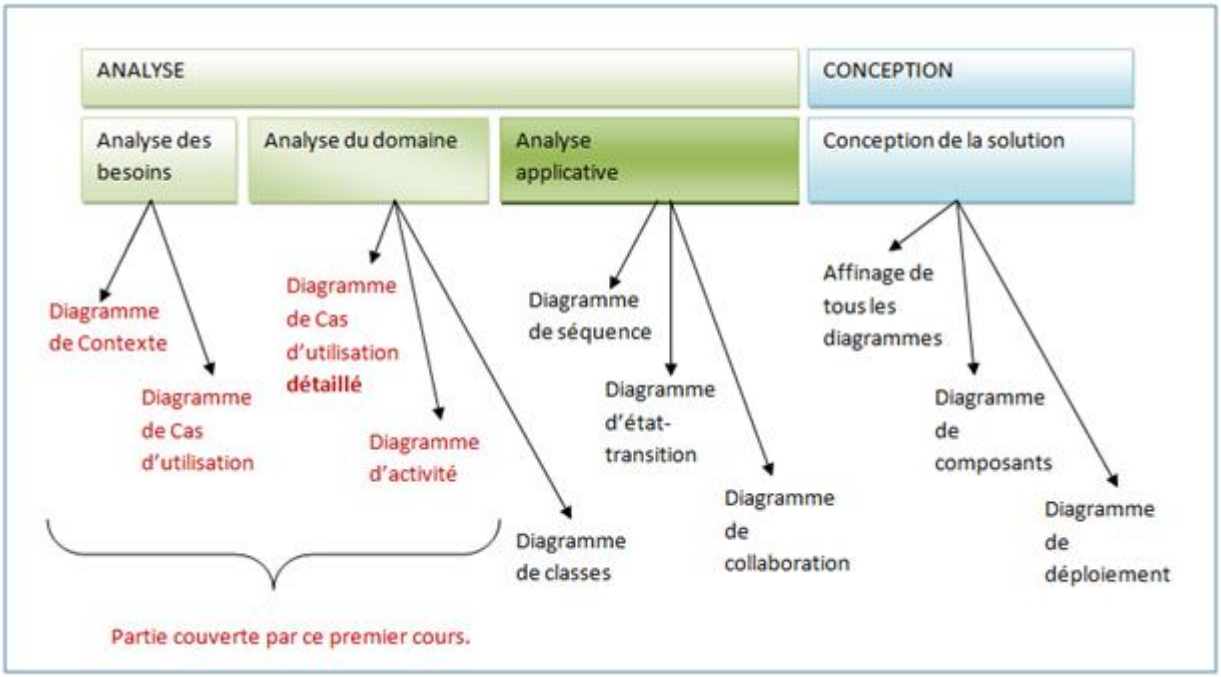


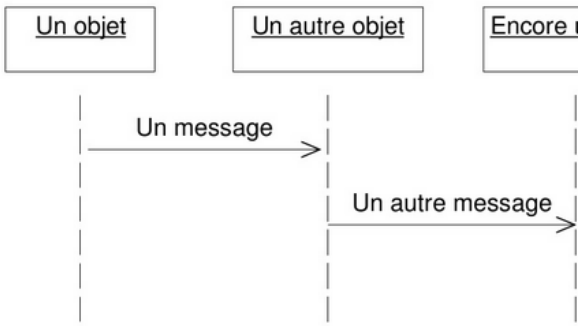
Diagramme de Cas d'utilisation (fonctions du système du point de vue des utilisateurs)

Diagramme d'objets

Diagramme de séquences

Diagramme d'états

Diagramme d'activité



23.1 Méthode

1. Analyse du cahier des charges
2. Informations d'entrée nécessaires/ informations de sorties
3. Dessin d'une IHM papier, insertion des objets et nommage des objets
4. Liste des objets nécessaires
5. Diagramme de cas d'utilisation
6. Propriétés utilisés sur les objets
7. Diagramme tâche/objets/évènement. Quel évènement déclenche une tâche donnée.
8. Diagramme de séquence
9. Diagramme d'état (que j'ai commencé à tester avec des 4eme et assez simple à faire passer car très naturel et découle de l'analyse précédente)
10. Puis algorithme d'une tâche (logigramme / pseudo code?)
11. Bilan des variables nécessaires et type de variables
12. Dessin sur un IDE donné de l'IHM et définition des objets et insertions
13. Programmation de l'algorithme par tâche associé à un évènement
14. Tracking d'une variable en mode pas à pas, visualisation du déroulement de l'algorithme dans un programme codé.
15. Débogage
16. Publication

23.2 Fiche méthode pour programmer une application :

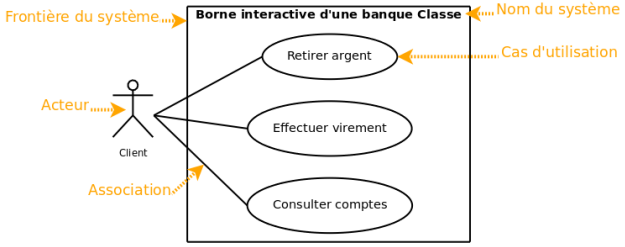
Titre de l'application

Informations d'entrée Informations de sortie

Dessin de l'IHM / type d'objets / noms de objets

Nom de l'objet	Type d'objet	Propriétés utilisées de l'objet

Diagramme de cas d'utilisation



Tâche	Objets	Evènement déclencheur

Diagramme de séquence

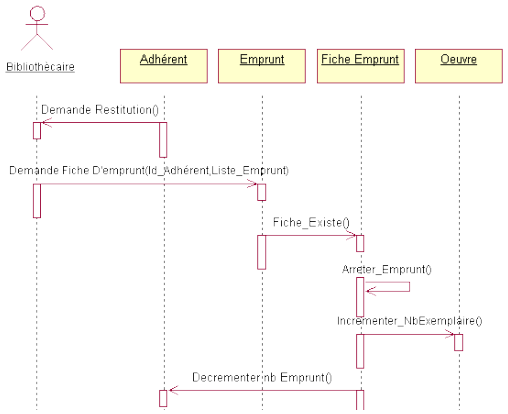
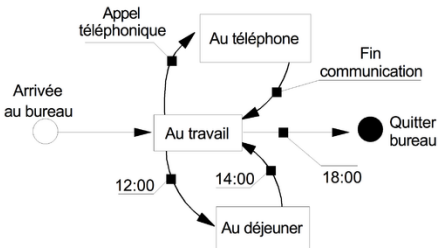
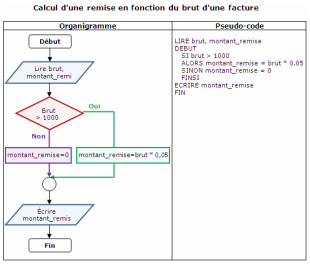


Diagramme d'état



Algorithme (logigramme / pseudo code)



Nom de la variable	Type de variable (booléen / texte / entier / réel)	utilisation

IHM sur l'IDE / Programmation sur l'IDE / Simulation / débogage / tracking mode pas à pas/ visualisation du contenu des variables / Essais sur cible / Publication.

24 Propositions pédagogiques

24.1 Méthode

Pour pouvoir aborder les différentes notions demandées par les programmes, il est important de définir une progression pédagogique de ces différentes notions.

Il serait intéressant de commencer par une application pour descendre à plus bas niveau.

Avant de se lancer dans un algorithme, les élèves peuvent analyser le fonctionnement d'une application pour justement s'initier au déroulement d'un programme. Cela fait d'ailleurs partie d'une des compétences proposées par le programme.

24.2 De l'algorithme vers le programme

Pour traduire un algorithme en code, il existe différents langages de programmation.

L'apprentissage du code semble facilité avec un premier apprentissage à l'aide d'IDE graphiques qui permettent de limiter les erreurs de syntaxe, de vocabulaire d'un langage donné.

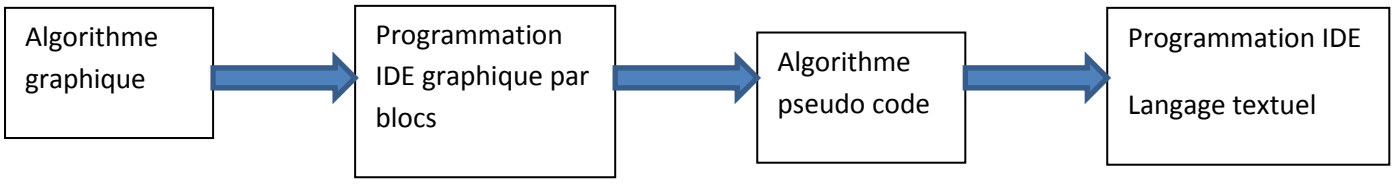
Une fois que les élèves auront acquis les compétences de bases demandées avec un système de type graphique, alors il sera possible de passer à la programmation textuelle dans un langage donné.

Il existe un grand nombre de langages de programmation comme :

- Langage Python (utilisé dans l'enseignement)
- Langage C (utilisé avec les systèmes Arduino)
- Langage Php
- Langage Html
- Langage Java

Mais pour que les élèves basculent progressivement vers ces langages, il parait intéressant qu'ils puissent s'initier au pseudo code en utilisant par exemple l'outil microalg ou algobox

En résumé :



Bon courage à vous avec vos élèves…

Table des matières