Atelier d'Introduction au numérique

1. Introduction

Le programme décrit ci-dessous propose une initiation au numérique et au développement d'applications mobiles, avec

- un atelier de 2 heures sur le son (physique)
- 5 séances d'une heure pour l'informatique :
    - introduction aux algorithmes sous forme de jeu (blockly maze)
    - réalisation collective d'un jeu (l'application mobile SMS game),
      puis réalisation individuelle (ou par 2), de l'application mobile "bonjour le monde",
    - réalisation (individuelle ou par 2) d'une application de traduction
    - réalisation d'une application web : open food facts
    - réalisation d'une application géo-localisée

  Le développement de chaque application se fait par étapes. On peut passer à la suivante sans toutes les réaliser, ce qui peut donner de la souplesse si les élèves progressent à des vitesses différentes.

2. Environnement

2.1. Environnement matériel

Pour l'atelier sur les sons il faut :

un PC, un smartphone et des petits matériels pour l'enseignant (entonnoir, bougie, enceinte BT) et - si on veut que les élèves expérimentent - un smartphone par groupe.

Pour les séances informatiques, il faut :

au moins un PC et un smartphone (ou tablette) Android pour 2 élèves avec accès à Internet.
Une dizaine de smartphones peut être mise à disposition. Les élèves pourront également utiliser leur mobile, si c'est un Android. (On pourrait aussi utiliser un simulateur à installer sur les PC).
La configuration est à vérifier au moins une semaine avant le début du projet.

2.2. Environnement logiciel

Pour l'atelier sur le son, on utilise des applications gratuites ou open source.

Pour les séances informatiques, on utilise des environnements de programmation graphique, dont les élèves sont déjà familiers avec Scratch. Ils sont gratuits et utilisés en préservant l'anonymat.

- blockly maze bien adapté à l'apprentissage des premières notions d'algorithmie.
- App Inventor pour réaliser des applications mobiles pour tablettes et smartphones. App Inventor est ouvert sur l'extérieur, il exploite les capteurs des smartphones, Internet et le Web. Comme Scratch, il vient du MIT, c'est son équivalent pour 12 ans et plus. Il permet aux élèves de créer des applis mobiles pour leur portable.

3. Ateliers sur la physique

3.1. Atelier sur le son

Principe :

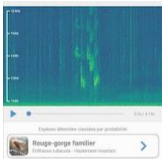
- le son est abordé comme une vibration qui se propage à une vitesse qui dépend du milieu, puis on expérimente la notion de fréquence, de graves et d'aigus, d'infra et ultra-sons puis la notion de niveau sonore mesuré en décibels, et les risques pour la santé,
- on voit ensuite à la représentation du son en fonction du temps, puis en fonction de la fréquence avec le spectre, puis en fonction des deux avec le spectrogramme ,
- enfin on applique ces notions à des cas concrets : reconnaissance des oiseaux par leur chant (birdNet), la mesure d'une distance par l'écho, ou du temps de réverbération d'une salle.

Déroulé :

- voir planches et media (dans atelierson.zip) en support et résumés : vidéoSon1 et vidéoSon2

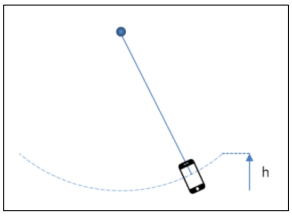
Moyens utilisés :

- l'enseignant dispose des planches avec les liens pour diffuser les sons utiles,
- il mesure le niveau sonore avec l'appli sonomètre sur son smartphone, à renvoyer à l'écran,
- une enceinte, un entonnoir, une bougie pour illustrer les dangers du niveau sonore,
- les outils utilisés sur le PC sont en ligne ou téléchargeables (audacity),
  les outils utilisés sur smartphone sont sur le playstore (sound meter, birdnet ou merlin bird), ou en ligne (sound spectrum analyzer),
- pour l'expérimentation
  - mesure du niveau sonore avec sound meter
  - reconnaissance des oiseaux par leur chant : Birdnet ou Merlin bird
  - affichage avec audacity (PC), également utilisable pour les mesures de la vitesse du son et du temps de réverbération avec un son enregistré.
  - en option pour des expériences mobiles avec un smartphone :
    - audio spectrum analyzer
    - function generator ou frequency sound generator

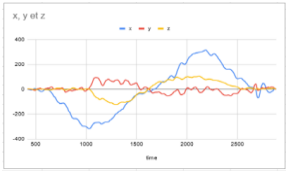


3.2. Atelier sur le mouvement et l'énergie (abandonné)

Le principe est d'utiliser le smartphone comme un pendule, de mesurer sa vitesse angulaire avec le gyroscope, d'en déduire sa vitesse linéaire, puis de mettre en correspondance les énergies cinétique et potentielle :
$½ v^2 = gh$ ou $½ v^2 – gh$ = constante.



A ce stade, la faisabilité n'est pas démontrée : le principe est simple, mais les sources de bruit ou d'erreur sont multiples. Pour l'instant, on a donc abandonné l'idée de cet atelier.



4. Séances informatiques

Sujet	temps	Activité	Points abordés	supports
Qu'est-ce qu'un algorithme ?	5	introduction	Ce qu'on va faire dans cette 1° séance et le suivantes	
	25	Blockly, niveaux 1 à 7 ou 9	Apprentissage par la pratique (analyse, codage, tests, séquences, itérations, conditions)	
	10	Revue : analyser, pseudo-code, traçage, …	Analyser/décomposer, pseudo-code intermédiaire entr notre langage et celui utlisé dans les programmes	
	10	Définition d'un algorithme	Définition d'un algorithme : comme solution à un problème Structuré par les séquences, boucles et branchements conditionnels L'algorithme pour faire des pâtes	
	?	Quiz	Quiz sur les algorithmes et la lecture (traçage) de code	Quiz
Présentation **Hello World et traduction**	10	• rappels et démarche (avec le Quiz ?) • présentation de ce que l'on va faire	Quiz sur les algorithmes et la lecture (traçage) de code applications mobiles, algorithmes, Internet, capteurs, géolocalisation	Quiz
	15	Hello world 1 • au démarrage, affichage d'un texte (design uniquement – pas de programme)	• lancement App Inventor • design, choix composants (ress.) • transfert smartphone, test	video
	15	Hello world 2 : • quand clic sur le bouton : le smartphone dit le texte à haute voix	<u>Design</u> • Interface / IHM : bouton • Media : composant texte à parole • nommage des composants <u>Blocs/programmation</u> ○ quand clic sur bouton : appeler la fonction qui lit le texte <u>Notions abordées</u> • identification des types de blocs en fonction de ○ leur couleur : évènements, propriétés, procédures (ou fonctions) ○ leur forme : obtenir ou écrire (get/set) • procédures et fonctions : certaines ne renvoient pas le résultat tout de suite • évènements • un algorithme ou un script par évènement. L'application c'est l'ensemble des scripts ou programmes exécutés en réponse à des évènements	

	Hello world 3 (traduction) • quand on clique sur le bouton : demander la traduction du texte en anglais • puis quand la traduction arrive : afficher la traduction et la dire à haute voix	<u>Design</u> • media : traducteur /translator • Interface : zone de texte <u>Blocs/programmation</u> • initialisation d'une variable pour la langue (langueOut) • quand clic sur bouton : appel de la procédure qui traduit avec 2 paramètres la langue dans laquelle traduire, et le texte à traduire • quand la traduction revient afficher le résultat de la traduction dans la zone du texte traduit ne pas oublier de modifier la langue du locuteur (texte à parole) et appeler la procédure texte à parole avec le résultat de la traduction <u>Notions abordées</u> • variables (couleur orange) et blocs pour les lire et les mettre à jour (get/set) • chaines de caractères (couleur grenat) : get • modification par programme des propriétés d'un composant • procédures ou foncions asynchrones : le programme n'attend pas le résultat. L'arrivée du résultat donne lieu à un nouvel évènement (quand traduction reçue) • programmation évènementielle : le programme comme série de scripts qui définissent la réaction à des évènements	
	Hello world 4 (reconnaissance de la parole) • quand on clique sur le bouton Ecouter demander au smartphone d'écouter ce que dit l'utilisateur • quand l'application reçoit le texte reconnu afficher le texte reconnu dans la zone de texte et reprendre le programme précédent pour traduire et dire.	<u>Design</u> • renommer le bouton : BoutonEcouter • ajouter media : reconnaissance vocale <u>Blocs/programmation</u> • quand clic sur BoutonEcouter : ○ lancer la reconnaissance vocale • quand texte reconnu : ○ si c'est le texte complet (partial= faux) afficher le résultat dans la zone de texte ○ puis (reprise du programme précédent) : demander la traduction <u>Attention</u> Dans la reconnaissance vocale (nouvelle version), il ne faut pas traiter les résultats partiels, donc vérifier que le paramètre partial est faux.	
	Hello world 5 (choisir la langue en sortie) • choix de la langue de traduction avec un spinner (curseur animé)	<u>Design</u> • Interface : ○ ajout spinner (ou curseur animé) : "SpinnerLangueOutput" et modification de la propriété "Elements " liste des langues : fr,en,es,it,de,ar,ru	

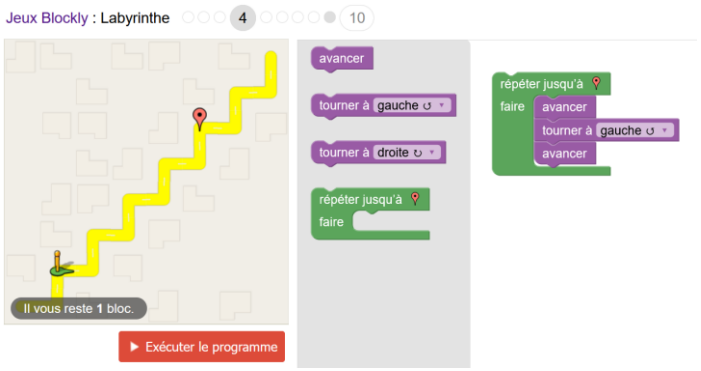
		<u>Blocs/programmation</u>
		• pour l'évènement « après sélection du spinner » :
		  ○ mettre à jour la variable langue avec la sélection du spinner
		• Au démarrage de l'application, évènement "quand screen initialize"
		  ○ mettre à jour la sélection du spinner avec la variable langue, pour que la valeur affichée par le spinner soit – dès le départ - la même que la variable langue	

	Hello world 6 (choisir la langue en entrée)
	• choisir la langue en entrée avec un spinner (curseur animé)
	• ajouter une image
		<u>Design</u>
		• Interface :
		  ○ ajouter un spinner (ou curseur animé) : SpinnerLangueInput
		  ○ ajouter une image
		<u>Blocs/programmation</u>
		• initialiser la variable langueInput : "fr"
		• au démarrage : quand screen1 Initialize
		  ○ copier la liste des langues (propriété Elements) de spinnerOutput dans la liste des langues (propriété Elements de spinnerInput
		  ○ mettre à jour la propriété sélection de spinnerInput avec langueInput (pour qu'elles correspondent dès le départ)
		• pour l'évènement « après sélection du spinnerIput » :
		  ○ mettre à jour la variable langueInput avec la sélection du spinner
		• Dans le script de l'évènement "quand clic sur bouton Ecouter" :
		  ○ mettre à jour la propriété langue du composant de reconnaissance vocale avec la variable langueInput	

	Hello world 7 (traduction bidirectionnelle)
	• quand on clique sur le bouton
	  demander au smartphone d'écouter ce que dit l'utilisateur et le traduire en texte
	• quand l'application reçoit le texte reconnu
	  affichage du texte reconnu dans la zone de texte
	  et reprise du programme précédent (traduction et vocalisation)
		<u>Design</u>
		• interface :
		  ○ renommer "BoutonEcouter" en "BoutonDirectEcouter", mettre couleur en gris
		  ○ créer un bouton "BoutonReverseEcouter" et mettre couleur en rouge
		  ○ créer un arrangement horizontal et mettre les 2 boutons à l'intérieur
		  ○ ajouter un label entre les deux, avec un texte vide en largeur 25%
		<u>Blocs/programmation</u>
		• Dans l'évènement quand clic sur "boutonDirectEcouter", ajouter
		  ○ mettre la couleur de fond de screen1 à gris
		  ○ mettre" langueInput" à SpinnerInput.Selection
		  ○ mettre" langueOutput" à SpinnerOutput.Selection
		• dupliquer le script quand clic sur "boutonDirectEcouter" et le recommmer en quand clic sur "boutonReverseEcouter", et modifier son contenu (couleur et inversion input/output)
		  ○ mettre la couleur de fond de screen1 à rouge
		  ○ mettre" langueOutput" à SpinnerInput.Selection
		  ○ mettre" langueInput" à SpinnerOutput.Selection	

Open food facts (scan de produits au super-marché)		Conception et réalisation d'une application Qu'est-ce qu'Internet ?	Web et Internet (URL) Capteurs	video
Géolocalisation et cartographie		Exercices sur la géolocalisation avec un smartphone Présentation des types de géolocalisation (GPS, Wifi, …) Réalisation d'une application qui affiche sa position sur une carte	Géolocalisation (GPS, Wifi, …) Serveurs de cartes (open street map)	
Application géolocalisée Pokemons		~~Lecture et affichage d'une liste d'objets géolocalisés (format geoJSON)~~ ~~Édition d'objets géolocalisés sur un serveur (Git)~~ ~~Calcul de distance entre l'utilisateur et un objet~~ ~~Sélection d'objet conditionné à la distance~~	~~Décomposition fonctionnelle~~ ~~Représentation et abstraction de données~~ ~~Utilisation d'un serveur pour partager des données~~ ~~Édition de données~~ ~~Calculs de distance~~ ~~Utilisation d'un serveur (Git)~~	

4.1. Séance 1 : algorithmes / labyrinthes



cliquer sur l'image pour lancer

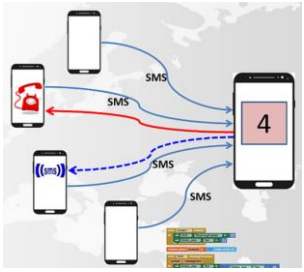
- Video de présentation
- notions abordées :
  - algorithmes, décomposition, diagramme fonctionnel ou pseudo-code
- objectifs pédagogiques :
  - comprendre la notion d'algorithme comme méthode de résolution de problème, et comprendre sa structure en séquences d'actions, boucles et conditions.
  - appréhender les étapes d'un développement : décomposition, passage du langage naturel au langage informatique (pseudo code), développement et test progressifs
- Déroulé : cf. supports
  - 35 mn : jeu blockly maze : labyrinthe
    - séquences (niveaux 1-2), boucles (niveaux 3-5), conditions (niveaux 6-8)
      (pour info : ce jeu similaire est plus difficile, mais basé sur la géométrie)
  - 10 mn revue de définition d'un algorithme : planches (1a et1b planches 3 à 34)
  - 10 mn écrire un algorithme / décomposition, pseudo code, codage, traçage
    - écrire – en classe entière - le pseudo code du niveau 9 (ou 8 selon le niveau)
    - exercice de lecture/traçage de code : https://docs.google.com/forms/d/e/1FAIpQLSfqwk17gyssRa-ztWGRXQb_K8Jh48ZjgqkkvvrDujmVO4MqYw/viewform
  - 5 mn debrief
    - qu'est-ce qu'un algorithme ?
    - comment écrire un algorithme : décomposer, écrire, tester
    - exercices à préparer le cours suivant : niveaux 9 et 10 ou niveau 10 https://blockly.games/maze?lang=fr&level=9&&skin=0
  - quiz sur les algorithmes (peut être fait au début du cours suivant
    - Quiz Google forms sur les algorithmes : https://docs.google.com/forms/d/e/1FAIpQLSf6v7jlArX32jfIS26okCdFk9tGQRMxlt8MH3J8SS5F1SYQhg/viewform

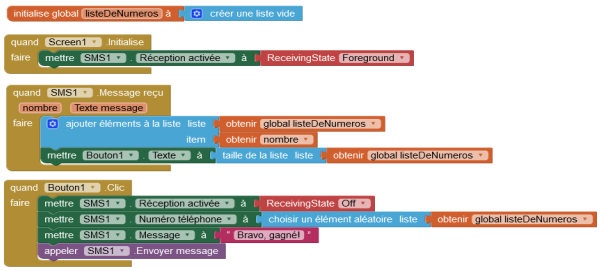
- exercice de lecture/traçage de code : https://docs.google.com/forms/d/e/1FAIpQLSfqwk17gyssRa-ztWGRXQb_K8Jh48ZjgqkkvvrDujmVO4MqYw/viewform

4.2. Séance 2 : Bonjour le monde, open food facts

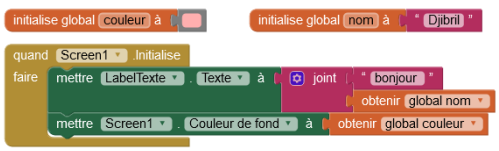
Dans cette séance on commence par concevoir et réaliser ensemble une première application très simple qui consiste à afficher, "bonjour le monde", puis avec le nom de l'utilisateur, puis avec une couleur de fond aléatoire, puis en rafraichissant l'affichage en fonction d'un capteur. Elle permet parcourir des notions diverses.

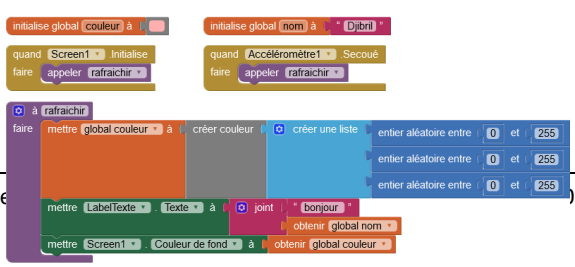
- notions abordées :
  - l'environnement de développement App Inventor
  - les étapes de design et de programmation,
  - les blocs : forme, couleur et catégorie
  - les évènements, scripts, variables et procédures
- objectifs pédagogiques :
  - savoir réaliser une application mobile simple
  - comprendre
    - les étapes : design avec le choix des ressources, puis programmation et tests
    - la nature des blocs : get/set ou obtenir/mettre (correspondant à leur forme)
    - la structure évènementielle d'une application (event driven)
    - les catégories de blocs : variables, propriétés, procédures et fonctions, évènements, texte, booléens (associées à leur couleur)
    - la notion de variable
    - la notion de procédure
    - la notion de capteur (moyen de perception de l'environnement)
- déroulé sommaire
  - SMS game : réalisation d'une loterie, faite et expliquée par l'enseignant.
    Les élèves jouent à la fin en envoyant un SMS, le gagnant tiré au sort reçoit un SMS
    cf. vidéo de développement de cette appli





  - "bonjour le monde" réalisation d'une première application par les élèves
    - V1 : Bonjour le monde
    - V2 : Bonjour + nom utilisateur
    - V3 : Bonjour + nom utilisateur
      + fond d'écran de couleur aléatoire





- V4 : ajout du rafraichissement d'écran quand on secoue l'appareil
    - déroulé détaillé : voir chapitre

4.3. Séance 4 : Application web (open food facts)

Rédaction en cours

4.4. Séance 5 : Application géolocalisée

Rédaction en cours

5. Déroulés détaillés

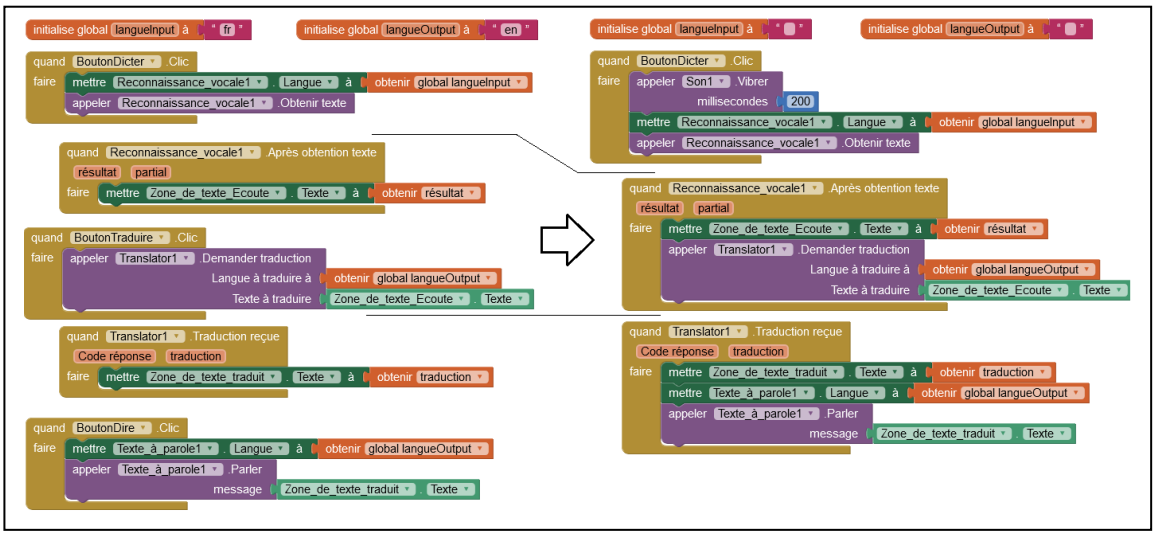
- séance 2 – "bonjour le monde"

5.1. Séance 3 : Application de traduction

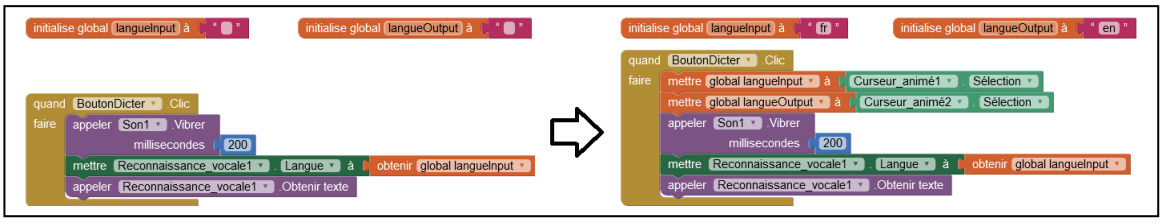
Dans cette séance les élèves réalisent une application qui va écouter ce que dit l'utilisateur, le traduire dans une autre langue et le redire dans cette langue. L'application fait appel à des fonctions de reconnaissance vocale, de traduction et de diction (texte à parole).

- notions abordées :
  - décomposition fonctionnelle : services et procédures asynchrones (on n'attend pas la réponse, son arrivée donne lieu à un évènement traité dans un autre script)
  - utilisations de services externes et/ou en ligne : reconnaissance de la parole, traduction, …
  - fonctions dites "asynchrones" : dans un script, le programme n'attend pas la réponse à une demande. L'arrivée de la réponse donnant lieu à un évènement traité par un nouveau script (programmation évènementielle).
- objectifs pédagogiques :
  - décomposer ce que doit faire l'application et trouver les composants pour le faire
    - 1. Convertir parole en texte, 2. Traduire texte, 3. dire le texte traduit
  - voir la différence entre une fonction qui donne la réponse tout de suite et une fonction (ou procédure) asynchrone où on n'attend pas la réponse.
  - utiliser une fonction avec des paramètres en entrée
  - modifier les propriétés d'une fonction
  - utiliser les données ou paramètres transmis par un évènement
  - utiliser une liste déroulante (spinner) pour choisir la langue
- déroulé sommaire (voir déroulé détaillé en annexe)
  - <u>V1 : version initiale (un bouton par étape)</u>
  - Design
    - 1 bouton pour écouter et convertir en texte ce que dit l'utilisateur
    - 1 bouton pour traduire le texte dans une autre langue
    - 1 bouton pour lire à haute voix le texte traduit
    - 2 zones de texte pour afficher les textes reconnus et traduits
    - 3 composants : reconnaissance vocale, traduction et 'texte à parole'
  - Codage/programmation : 5 scripts
    - Quand clic sur 1° bouton : lancer la reconnaissance vocale
    - Quand le texte a été reconnu : l'afficher
    - Quand clic sur 2° bouton : lancer la traduction
    - Quand le texte a été traduit : afficher ce texte traduit
    - Quand clic sur 3° bouton : convertir le texte traduit en parole

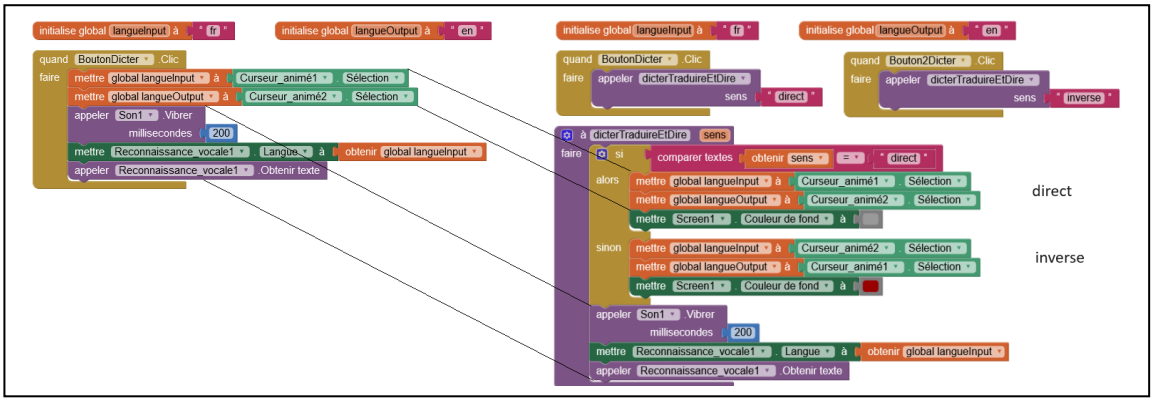
- V2 : enchainement des opérations sur un bouton
  - Codage/programmation : 3 scripts
    - Quand clic sur 1° bouton : lancer la reconnaissance vocale
    - Quand le texte a été reconnu : l'afficher et lancer la traduction
    - Quand texte a été traduit : afficher traduction et convertir en parole



- V3 : choisir les langues en entrée et en sortie
  - Design
    - Ajout de 2 spinners (curseurs animés) pour choisir chaque langue dans une liste. On définit la liste dans les propriétés et la langue sélectionnée au départ.
  - Codage
    - Quand l'utilisateur a sélectionné une nouvelle langue, mise à jour de la variable qui porte cette langue : langueInput ou langueOutput

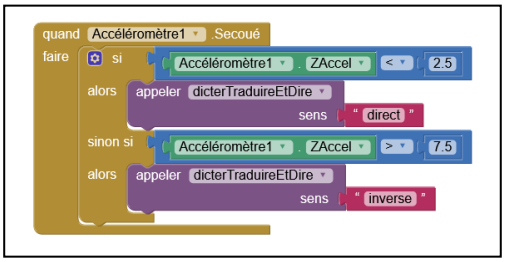


- V4 : traduction bidirectionnelle, selon l'orientation du smartphone
  - Design
    - ajout d'un bouton pour dicter dans l'autre sens
    - accéléromètre pour détecter la position du smartphone
  - Codage
    - Quand clic sur bouton1 Dicter
      - langueInput ← langue sélectionnée par le curseur 1
      - langueOutput ←langue sélectionnée par le curseur 2
        - appeler la procédure dicter
    - Quand clic sur bouton1 Dicter
      - langueInput ← langue sélectionnée par le curseur 2
      - langueOutput ←langue sélectionnée par le curseur 1
        - appeler la procédure dicter



Bonus avec l'accéléromètre

- Quand téléphone secoué
  - si (acc z >7.5)
    appeler la procédure dicter (direct)
  - sinon sii (acc z <2.5)
    appeler la procédure dicter (inverse)



5.2. Séance 4 : Application web (open food facts)

Rédaction en cours

5.3. Séance 5 : Application géolocalisée

Rédaction en cours

6. Déroulés détaillés

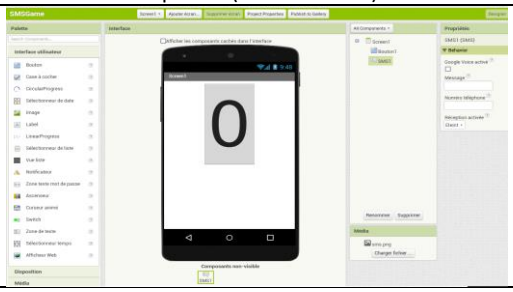
6.1. séance 2 – "bonjour le monde"

6.1.1. Introduction avec SMS game :

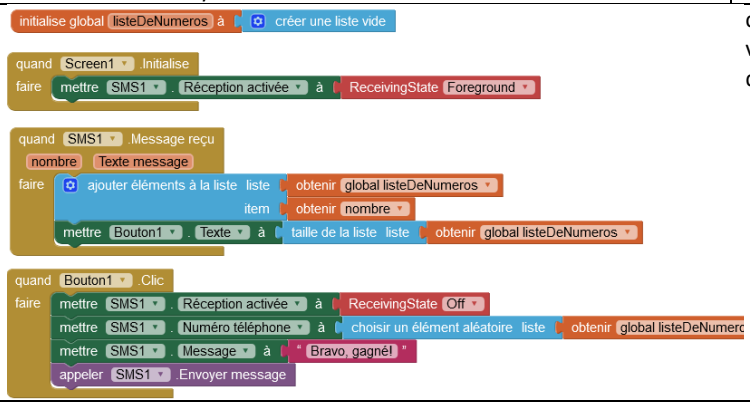
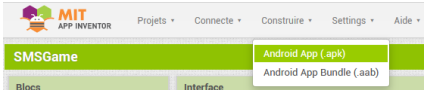
L'animateur réalise une application qui attend et enregistre les SMS, puis - quand on clique sur un bouton – tire un numéro au hasard et lui envoie un SMS "vous avez gagné".

étape	Description	Notions

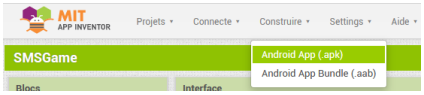
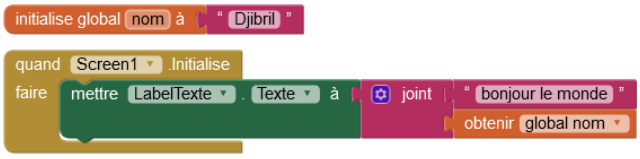
Design conception	Design / décomposition:
	• Dessiner l'interface et écrire une phrase qui décrit l'application
	• Identifier
	  ○ les ressources nécessaires : bouton + composant sms
	  ○ les évènements
	  ○ et données manipulées (liste numéros)
		maquette
		évènements
		composants visibles (bouton) et cachés : SMS
		variable de type liste

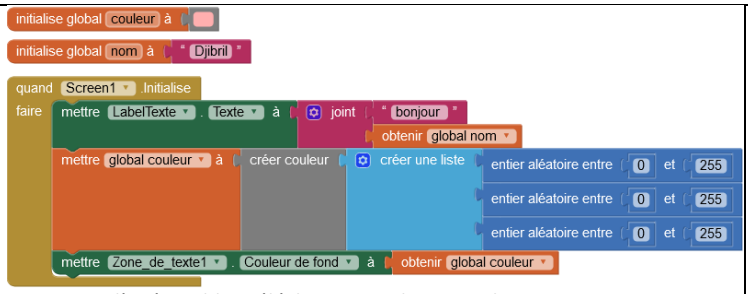
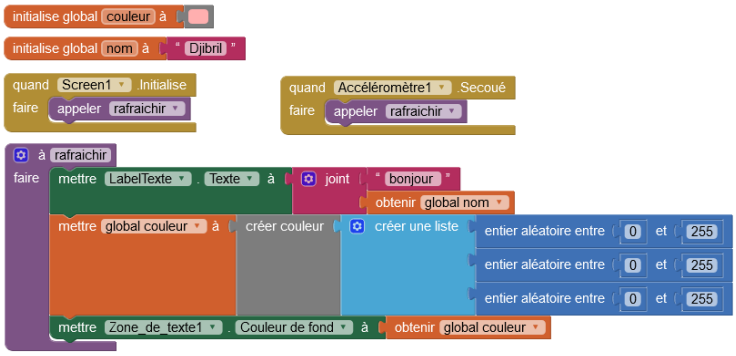
Design réalisation		Ajout et paramétrage des composants, Mise en page

Pseudo code	• déclarer et initialiser la liste des numéros
	• quand application démarre
	  ○ activer SMS
	• quand SMS reçu
	  ○ ajouter numéro à liste numéros
	• quand bouton cliqué
	  ○ choisir un numéro au hasard
	  ○ envoyer un sms à ce numéro
		Pseudo code de chaque évènement

codage		codage vérification, traitement des erreurs
Création exécutable	• Installation d'AI2 companion depuis le playstore • Construire l'exécutable  • Télécharger l'application sur le smartphone 	création de l'exécutable (.apk) transfert sur smartphone essais

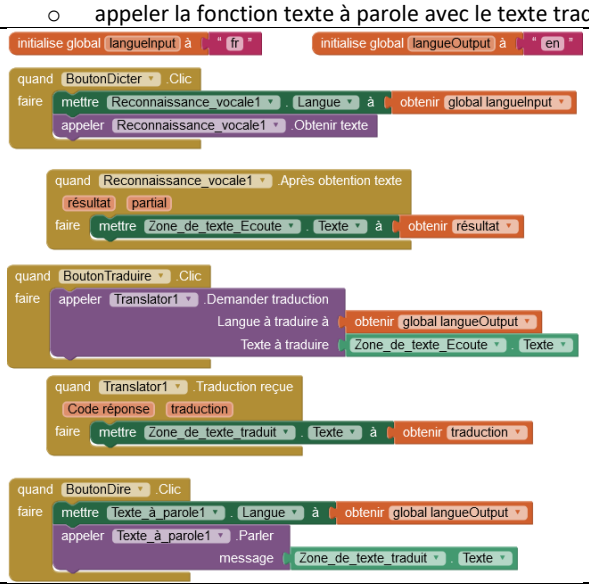
6.1.2. Application "bonjour le monde" :

étape	Description	Notions vues
version 1	Préparation : installer l'appli AI2 companion sur le smartphone Description : au démarrage, l'application affiche 'bonjour le monde' Design : Ajout d'un composant label : label texte réglage des propriétés à droite : Centrage verticale et horizontal de Screen1 Réglage de la taille et de la couleur des caractères de labelTexte Evènements à traiter: quand l'application démarre (screen.initialize) Pseudo code : *quand Screen1. Initialize* *mettre le texte de labelTexte à "bonjour le monde"* Codage :  Construire l'exécutable :  Télécharger sur le smartphone :  Tester	Réalisation d'une application complète minimale Design : • Choix et ajout d'un composant label pour afficher un texte Mise en page : • modification des propriétés de Screen1 : centrage horizontal et vertical, choix de la couleur de fond • modification des propriétés de labelTexte, taille et couleur des caractères compilation du code, transfert smartphone, exécution et tests
Version 2	Description : au démarrage, afficher 'bonjour' + le nom de l'utilisateur Design : sans changement Evènements : sans changement Pseudo code : *Initialiser la variable nom avec une valeur* *quand screen1. Initialize* *mettre le texte de labelTexte à "bonjour le monde" + nom* Codage :  Construire l'exécutable : Télécharger sur le smartphone : Tester	Variable de type texte Bloc de jointure de textes
Version 3	Description : comme précédent + choisir une couleur aléatoire pour le fond d'écran couleur du fond d'écran de façon Design : sans changement Evènements : sans changement Pseudo code : *Initialiser la variable couleur* *Initialiser la variable nom avec une valeur* *quand screen1. Initialize* *mettre le texte de labelTexte à "bonjour le monde" + nom* *mettre dans la variable couleur, une dose aléatoire de rouge, vert et bleu* *affecter la couleur à la propriété couleur de fond d'écran* Codage :	Définition d'une couleur depuis une liste de base Définition d'une couleur par ses composantes RVB entre 0 et 255 Fonction qui renvoie un entier aléatoire Modification d'une propriété de Screen1

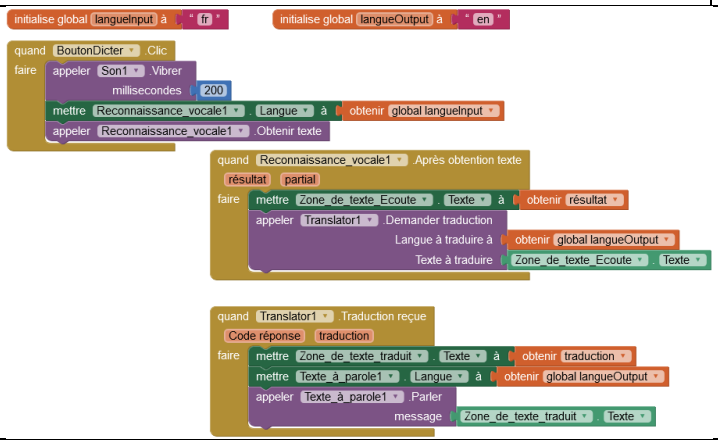
	 Construire l'exécutable, Télécharger sur le smartphone, Tester	
Version 4	Description : idem + rafraichir quand on secoue l'appareil Design : Ajouter un composant non visible accéléromètre Evènements : Nouvel évènement quand accéléromètre secoué Pseudo code : *Initialiser la variable couleur* *Initialiser la variable nom avec une valeur* *quand screen1. Initialize* *appeler la fonction rafraichir* *quand accéléromètre. secoué* *appeler la fonction rafraichir* ***procédure rafraichir*** *mettre labelTexte.texte à "bonjour le monde" + nom* *mettre dans la variable couleur, une couleur qui contient une dose aléatoire de rouge, de vert et de bleu* *affecter la couleur à la propriété couleur de fond d'écran* Codage :  Construire l'exécutable, Télécharger sur le smartphone, Tester	Utilisation d'un composant accéléromètre Utilisation d'une procédure pour : 1. éviter les doublons 2. Généraliser / abstraire nota : on commence par dupliquer le code, puis on crée la procédure pour éviter les doublons

6.2. séance 3 – "application de traduction"

6.2.1. V1 : écouter, traduire et dire dans une autre langue

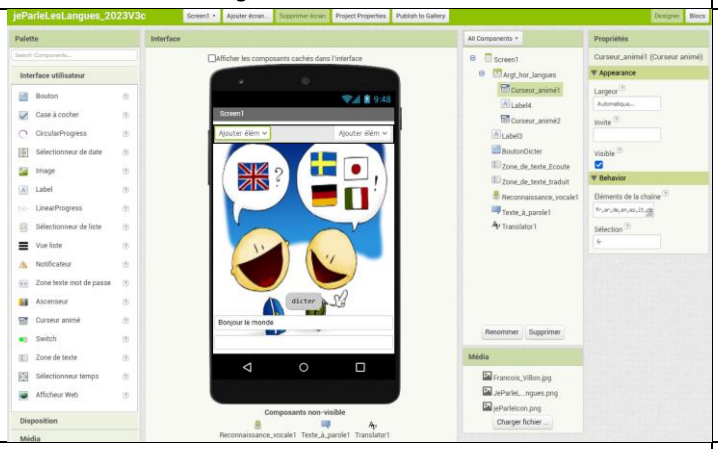
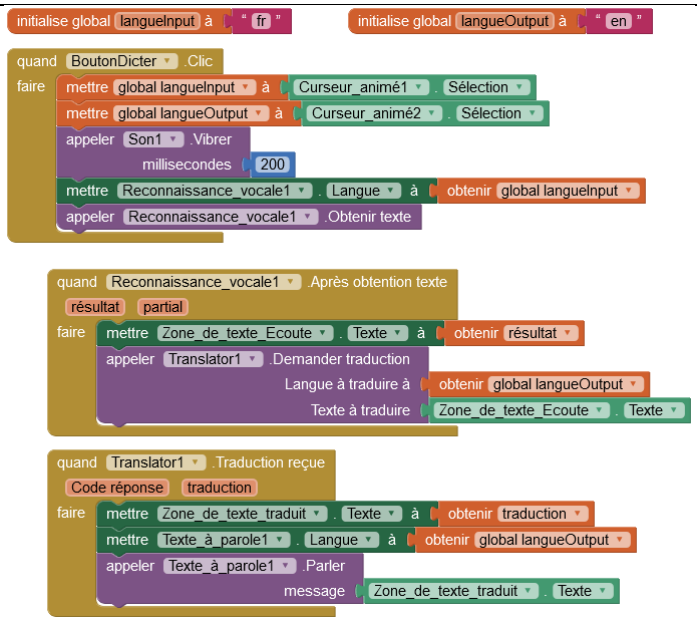
étape	Description	Notions
Description	l'application doit permettre d'écouter ce que dit l'utilisateur, le traduire dans une autre langue et le redire dans cette langue	
Décomposition	• "dicter" et convertir en texte à afficher (reconnaissance vocale) • "traduire" et afficher traduction qui est affichée • "lire » le texte traduit à haute voix	Analyse, décomposition fonctionnelle
Design réalisation	Composants visibles • 3 boutons : "écouter", "traduire", "lire" • une zone de texte pour le texte reconnu • une zone de texte pour le texte écouté traduit Composants invisibles • reconnaissance vocale, translator, texte à parole Propriétés • ajouter une image au format png (transparent) en fond d'écran • choisir le nom et la taille des boutons 	Ajout et paramétrage des composants, Mise en page
Données et variables	Données et variables de l'application • langueInput : la langue à écouter en entrée • langueOutput : la langue de traduction en sortie • texte écouté (propriété du composant texte) • texte traduit (propriété du composant texte)	
Pseudo code	• définir/initialiser langueInput et langueOutput • quand clic sur bouton dicter ○ mettre la propriété langue à écouter à langueInput ○ appeler la fonction de reconnaissance vocale • quand texte reconnu ○ afficher le texte dans la zone de texte ecoute • quand clic sur traduction ○ appeler la fonction de traduction avec en paramètres, le texte à traduire et la langue (langueOutput) • quand traduction obtenue ○ afficher la traduction dans la zone de texte trad. • quand clic sur lire ○ mettre la propriété langue à lire à langueOutput ○ appeler la fonction texte à parole avec le texte traduit	
codage		

6.2.2. V2 : automatiser (tout enchainer sur un seul clic)

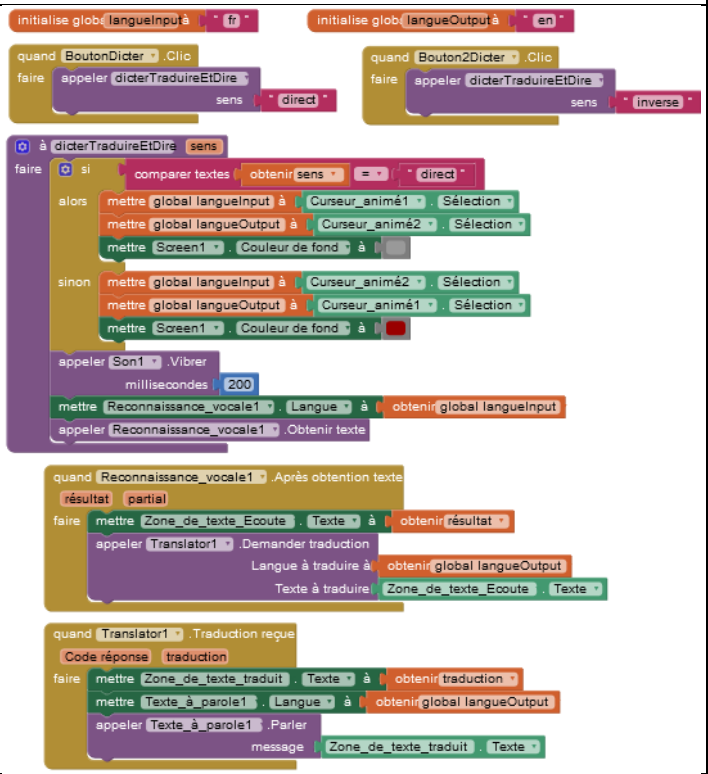
étape	Description	Notions
Description	Dans cette version, on va enchainer les actions après avoir déclenché la reconnaissance vocale avec le bouton dicter, donc lancer la traduction dès que le texte est reconnu et lancer la lecture dès que la traduction est disponible. Les boutons traduire et lire deviennent inutiles. On va aussi améliorer l'ergonomie en faisant vibrer le smartphone quand on demande la reconnaissance vocale	Analyse, décomposition fonctionnelle
Description Décomposition	• Quand l'utilisateur clique sur le bouton « écouter » écouter ce que dit l'utilisateur, • quand le texte est reconnu afficher le résultat et demander la traduction • quand la traduction arrive, l'afficher et demander la lecture à haute voix	Analyse, décomposition fonctionnelle
Design	• Ajouter un composant 'son' pour pouvoir faire vibrer • On n'utilisera plus les boutons traduire et lire, mais avant de les supprimer, récupérer les lignes de code 	
Pseudo code	• Quand l'utilisateur clique sur le bouton "dicter" ○ faire vibrer ○ et demander la reconnaissance vocale • quand le texte est reconnu ○ afficher le résultat ○ demander la traduction • quand la traduction arrive, ○ afficher la traduction ○ demander la lecture à haute voix	
codage		

6.2.3. V3 : choisir la langue à écouter et à traduire dans une liste

○

étape	Description	Notions
Description	Dans cette version, l'utilisateur peut choisir la langue en entrée et la langue en sortie	
Décomposition	On va utiliser un curseur animé (spinner ou liste déroulante) pour choisir chacune des langues. La liste des langues sera définie dans les propriétés des 2 curseurs animés (on peut aussi les définir dans le programme, quand Screen1.initialize)	
Design/ composants	• Composants visibles : ajouts ○ un arrangement horizontal (largeur remplir parent) ○ avec 2 curseurs animés (spinner) In et Out ▪ sélection 'fr' pour in, 'en' pour out ▪ element : fr,ar,de,en,es,it pour les deux ○ séparés par un label (largeur fill parent) • invisibles : inchangés	
Design		
Pseudo code	• Après sélection dans la liste des langues à écouter : ○ Mettre la langue sélectionnée comme langue en entrée • Après sélection dans la liste des langues de traduction : ○ Mettre la langue sélectionnée comme langue de traduction	
code		

6.2.4. V4 : traduction bidirectionnelle

étape	Description	Notions
Description	• l'application doit traduire dans un sens quand c'est l'utilisateur qui parle et dans l'autre quand c'est l'interlocuteur qui répond.	
Design / analyse	• utiliser l'orientation du smartphone (vertical ou tendu vers l'utilisateur) pour déterminer qui parle. • à chaque changement changer la couleur de l'écran	
Design / composants	• visibles : sans changement • invisibles : ajout d'un composant accéléromètre	
Design		
Pseudo code	○	
codage		
Bonus accéléro	• quand smartphone secoué change ○ si accélération z < 2.5 ▪ appeler dicterTraduireEtDire(direct) ○ sinon accélération z < 2.5 ▪ appeler dicterTraduireEtDire(inverse)