

App Inventor 2

Concevez des applications Android pour mobile

Ce livre a pour objectif de vous former à la réalisation d'applications Android pour appareils mobiles à l'aide de la plateforme **App Inventor 2** initialement conçue par Google, des premières étapes de réflexion jusqu'à la mise en ligne sur le Play Store.

Il s'adresse aux non développeurs ayant besoin de monter en compétence sur ce sujet et aux passionnés avides de faire leurs propres expériences.

Le premier chapitre présente **App Inventor 2**, la popularité de cette solution, son contexte d'utilisation, ses possibilités, en se basant sur des exemples d'applications existantes.

Le deuxième chapitre vous montre comment **créer, en quelques minutes, une application pour mobile** fonctionnelle sur App Inventor 2, sans être développeur.

Le chapitre suivant présente une **méthodologie de travail** reposant sur la réflexion que vous devez avoir en amont, avant la création de toute application sur mobile, afin de mener votre projet à terme sans avoir à le reconsidérer durant la phase de conception il présente également des **outils pratiques** qui vous permettront de créer des applications robustes que vous pourrez par la suite partager autour de vous, voire commercialiser sur le Play Store de Google.

Le quatrième chapitre présente en détail **l'environnement de travail** d'App Inventor 2 et de nombreuses techniques pour être encore plus efficace au quotidien.

Les chapitres suivants, qui représentent le cœur de l'ouvrage, détaillent l'ensemble des **composants présents sur la plateforme**, ils comprennent de **nombreux exemples** que vous pourrez reproduire afin de vous approprier au mieux cette solution.

Le livre se termine par la **publication** d'une application et vous présente quelques solutions (AdMob, Android Studio) permettant d'aller plus loin dans la conception d'application mobile.

Ronan CHARDONNEAU

Ronan CHARDONNEAU est un passionné de marketing digital depuis ses premières années d'études à l'université ; depuis, il a eu l'occasion d'évoluer auprès de nombreuses agences web et annonceurs en France et à l'étranger. Aujourd'hui Maître de conférence associé à l'université d'Angers, il forme tous les ans plusieurs centaines d'étudiants au marketing en ligne. Ronan travaille sur le sujet des objets connectés depuis 2015, notamment pour montrer à ses étudiants que le marketing digital ne se limite pas aux sites internet et que les problématiques ne sont pas résolues tant que le client final n'est pas satisfait à 100%. Il est également auteur de plusieurs ouvrages parus aux Éditions ENI sur ces thématiques : Google Analytics, Piwik, Google Tag Manager.

Introduction

"Beaucoup de personnes ne sont pas d'accord avec moi, mais ça (montrant un smartphone), c'est un objet connecté"

Yannick DESSERTENNE, Société nationale des objets connectés

À propos de l'auteur

Ronan Chardonneau est un passionné de marketing digital depuis ses premières années d'études à l'université.

Il a eu l'occasion d'évoluer auprès de nombreuses agences web et annonceurs en France et à l'étranger.

Maître de conférence associé à l'université d'Angers, il forme par an plusieurs centaines d'étudiants au marketing en ligne.

Il est formateur indépendant depuis 2010 sur les solutions en analyse de données telles que Google Analytics, Piwik, Google Tag Manager.

Il est également l'auteur d'ouvrages sur ces sujets aux Éditions ENI : Google Analytics, Piwik l'alternative Open Source en web analytics, Google Tag Manager.

Il est l'organisateur du MeasureCamp pour la ville de Nantes : <http://nantes.measurecamp.org>, événement rassemblant les analystes digitaux de toute la France et soutenu par l'AADF (*Association des Analystes Digitaux Francophones*).

Il travaille sur le sujet des objets connectés depuis 2015, notamment pour montrer à ses étudiants que le marketing digital doit aller au-delà des sites Internet : référencement naturel, affiliation, achat de liens sponsorisés...

À propos de ce livre

Chers lecteurs,

Le livre que vous avez entre les mains est celui que j'aurai aimé avoir il y a cinq ans de cela quand le boom des applications pour mobiles a commencé en France. À l'époque le marché francophone était activement (et c'est toujours le cas) à la recherche de développeurs d'applications pour mobiles. Les chiffres annoncés en agence pour la réalisation de telles applications étaient de l'ordre de plusieurs dizaines de milliers d'euros et les succès de ces projets incertains quant aux retombées économiques. À côté de cela, les responsables en marketing digital voyaient de loin ces projets défiler sans savoir réellement comment ils pouvaient apporter leur pierre à l'édifice, après tout, les applications pour mobiles ne concernent que les développeurs. Le livre que j'ai rédigé sur App Inventor 2 sert justement à casser cet a priori, il permet à des non-développeurs de réaliser leur propre application sur Android et à prendre confiance sur un secteur qui, jusqu'à présent, leur semblait comme trop technique.

Dans ce contexte, l'objectif de ce livre est assez simple : "**"ratrapper le fossé intellectuel" non acquis au cours des dernières années sur le développement d'applications mobiles.** Grâce à celui-ci, vous allez passer d'un niveau débutant à un niveau intermédiaire/avancé, et ce exactement de la même manière que les CMS ont fait de vous des webmasters aguerris alors que vous saviez à peine comment mettre une page sur un serveur web.

Ce livre a également pour objectif d'aller voir plus loin et de proposer une **initiation à l'univers des objets connectés**. Après tout, un smartphone n'est qu'un super-objet connecté. Étant donné que vous l'avez dans votre poche, pourquoi ne pas l'utiliser pour vos projets domotiques, créer l'objet de vos rêves afin de gagner un temps précieux, automatiser de nombreuses tâches ?

À qui s'adresse ce livre ?

Ce livre, et j'insiste sur ce point, s'adresse à des non-développeurs. Quel que soit votre profil, vous devriez être en mesure de lire cet ouvrage sans aucune difficulté particulière. La seule "barrière" est matérielle, il est fortement recommandé d'avoir son propre ordinateur, une connexion (au minimum ADSL) Internet, un appareil mobile sous Android.

Pourquoi avoir écrit ce livre ?

App Inventor 2 fait l'objet de nombreux ouvrages dans le domaine anglophone, hispanophone, mais très peu de ressources sont disponibles en français. La demande grandissant sur les formations en marketing mobile, j'ai décidé de mettre à disposition mes connaissances sur le sujet afin d'en faire profiter le maximum d'entre vous, dont mes étudiants. À noter que la plupart des ouvrages que vous trouverez sur le marché concernant App Inventor 2 sont orientés vers des "cas pratiques", c'est pourquoi j'ai souhaité focaliser davantage mon ouvrage sur les capacités offertes par la plateforme. J'avais personnellement besoin d'un guide exhaustif sur les limites d'App Inventor 2.

Pour la petite anecdote, mon grand frère, ingénieur et responsable informatique m'avait demandé un jour de lui prouver qu'App Inventor 2 était une belle plateforme. La mission qu'il m'avait confiée était de créer une application mobile permettant d'obtenir le résultat de deux dés à six faces. Le temps de me connecter à App Inventor 2 et 30 secondes plus tard, l'application était déjà sur mon téléphone !

App Inventor 2 m'a permis d'apprendre la logique informatique. Certes, je suis loin d'être un développeur aguerri, mais cela m'a permis de comprendre de nombreux aspects liés à la connectivité et au Web. Avant App Inventor 2, j'avais à de nombreuses reprises essayé de me former au développement informatique : cours à l'université, MOOC, lecture de livres pédagogiques... Aucun ne m'a permis d'apprendre aussi vite et aussi bien qu'avec la logique d'App Inventor 2 qui allie l'aspect ludique et l'aspect pragmatique.

À la manière de Grace Hopper (<http://www.stanleycolors.com/2014/12/story-grace-hopper-aka-amazing-grace/>), j'ai attrapé le virus de décomposer la plupart des objets connectés qui nous entourent afin de comprendre leur fonctionnement : thermostat, logiciel de passage en caisse des grandes surfaces, indicateurs de présence des places de parking,

télécommande...

J'ai également réalisé que l'univers de l'Internet des objets et de la robotique pouvait facilement être approché par l'intermédiaire des applications pour mobile.

En espérant que ce livre vous permettra de réaliser vos projets du moment et qu'il vous apportera autant que ce qu'il m'a offert, je vous souhaite une très bonne lecture.

Ronan Chardonneau

Remerciements

Ce livre est le fruit d'un très grand ensemble de problématiques rencontrées et différents univers dans lesquels j'ai pu évoluer au cours de ces dernières années. Mes remerciements vont naturellement à l'ensemble de tous ces acteurs.

L'université d'Angers

Merci à mes étudiants du master 2 Marketing Technologies de l'Information et de la Communication de l'université d'Angers qui, d'année en année, me poussent sans cesse à être à la pointe des connaissances en marketing digital. Leur énergie déployée durant les cours a été ma principale motivation pour rédiger cet ouvrage. Merci également au fondateur historique de cette formation, monsieur Bruno Daucé, maître de conférences en marketing sensoriel - marketing olfactif - marketing expérientiel qui me fait confiance depuis mes débuts à l'université, notamment dans la mise en place de cours innovants et expérimentaux en marketing digital.

L'école des mines de Nantes

J'ai eu l'occasion de travailler durant près de quatre ans au sein de l'incubateur de l'école des mines dans deux start-ups : Jymeo, Sensorwake. Ces années m'ont permis de croiser et d'échanger avec des entreprises très innovantes et ainsi de pouvoir imaginer bien au-delà des prestations de services classiques, les produits de demain.

Les communautés

Par cet ouvrage je tiens à remercier l'ensemble des créateurs et animateurs d'événements de conférences (MeasureCamp, QueDuWeb, Web2Day, Matinales MTIC, WebCampDay...) qui nous permettent à nous, professionnels, de nous nourrir de nouvelles idées, de nous confronter à nos confrères et de toujours aller voir plus loin. Merci à Taifun, webmaster du site <http://puravidaapps.com> qui a su répondre à toutes mes questions techniques sur App Inventor 2 ainsi que Pierre Huguet, éducateur sur AI2 et Hossein Amerkashi de AppyBuilder.

Les professionnels

Je souhaite remercier deux entreprises en particulier, Playmoweb qui, par l'intermédiaire de Rémi Lanoës m'a permis de voir les attentes des professionnels quant au développement des applications mobiles. Ainsi que l'entreprise Webyousoon qui, par l'intermédiaire d'Alvin Berthelot, m'a permis de découvrir de plus près Android Studio et ses différences avec App Inventor 2.

Les Éditions ENI

Merci aux Éditions ENI qui m'ont à nouveau ouvert leur porte pour la rédaction d'un cinquième ouvrage, qui plus est, bien différent des autres qui étaient rédigés sur l'analyse de données. Merci pour votre confiance, sans vous il n'y aurait pas diffusion de connaissance à grande échelle.

À ma femme

Parce que je ne tiens finalement jamais mes promesses et que quand je dis qu'il n'y aura pas d'autres livres... j'en écris un autre quelques mois plus tard. Merci de me supporter.

À vous lecteur

Merci d'avoir fait le choix de ce livre. Si vous n'avez aucune connaissance sur la création d'applications mobiles... vous ne pourrez être déçu.

Introduction

Dans ce premier chapitre, nous allons découvrir ce qu'est App Inventor 2, par qui il a été conçu, les besoins auxquels il répond ainsi que ses limites.

Les objectifs de chapitre :

- Comprendre ce qu'est App Inventor 2,
- Connaître les libertés offertes par la solution.

Présentation d'App Inventor 2

Développé initialement par Google, App Inventor 2 est un service gratuit qui vous permet de créer des applications pour téléphones mobiles/tablettes Android en quelques secondes sans avoir à programmer.

Pour ce faire, vous aurez besoin des éléments suivants :

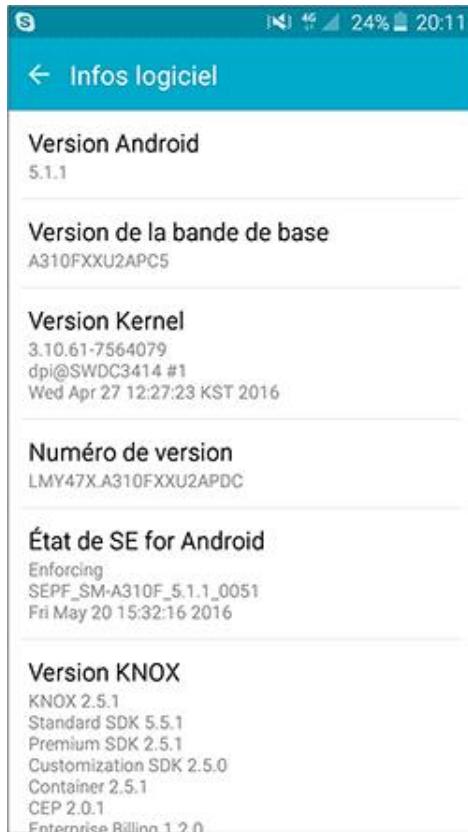
- un ordinateur : Windows, Mac, Linux
- un navigateur : Google Chrome, Firefox, Safari (Internet Explorer n'étant pas supporté)
- une connexion à Internet (même si vous pouvez installer un logiciel vous permettant de travailler hors ligne)
- un compte Google (créé automatiquement dès lors que vous utilisez déjà un service de Google tel que Gmail)
- un terminal (smartphone ou tablette) avec un système d'exploitation Android version 2.3 ou supérieur

Comment vérifier la version du système d'exploitation d'un Android ?

Il est difficile de donner des conseils précis tellement les terminaux sont nombreux. Cependant, il y a de grandes chances pour que vous trouviez la version de votre système d'exploitation Android en allant dans :

- **Paramètres**
- **À propos du téléphone**

Il est possible que vous ayez des étapes supplémentaires, dans le cas de figure ci-dessous il fallait également cliquer sur **Infos logiciel** :



Pour toutes vérifications : <http://appinventor.mit.edu/explore/content/system-requirements.html>

App Inventor 2 a été créé et est actuellement maintenu par le MIT (*Massachusetts Institute of Technology*). Par maintenance, on entend le fait que le service est mis à jour par le MIT et également hébergé. C'est-à-dire que tout votre travail est directement hébergé sur les serveurs du MIT et vous n'avez pas besoin de payer pour cela.

Bien qu'App Inventor 2 permette de créer des applications pour Android uniquement, vous pouvez utiliser leurs plateformes si vous disposez d'un ordinateur Windows, Linux, Mac, Android. À noter que tous les navigateurs Internet ne supportent pas App Inventor 2, c'est le cas par exemple d'Internet Explorer, il vous faudra alors vous orienter vers les solutions suivantes : Google Chrome, Firefox, Safari.

À la date de rédaction de ce livre, App Inventor 2 compte :

- 273 000 utilisateurs actifs
- 5 millions d'inscrits
- 16 millions d'applications créées

Pourquoi parler d'App Inventor 2 et non d'App Inventor ?

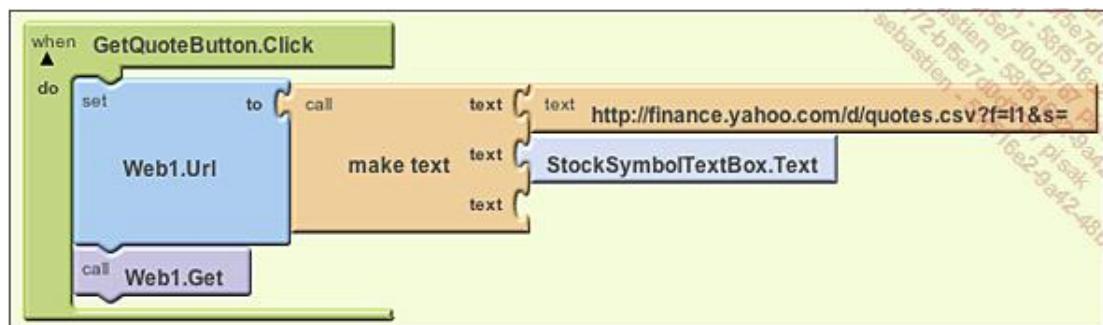
App Inventor 1 permettait tout comme App Inventor 2 de pouvoir créer des applications pour Android mais avait cette particularité de ne pas fonctionner dans le Cloud, il vous fallait alors avoir le logiciel Java d'installé sur votre ordinateur.

Qu'est-ce que Java ?

Java est un langage de programmation et une plateforme informatique qui ont été créés par Sun Microsystems en 1995. Vous avez probablement déjà croisé son logo représenté par une tasse de café.

Une mise à jour du design des blocs logiques a également été prise en considération pour cette version d'App Inventor 2.

Lorsque vous recherchez des tutoriels sur Internet, vous tombez d'ailleurs parfois sur ces anciens blocs propres à App Inventor 1 :



Présentation d'Android

Android est un système d'exploitation au même titre que Microsoft Windows, Apple ou encore Linux, c'est donc le logiciel qui permet de faire fonctionner votre terminal et de l'exploiter... d'où son nom. Il appartient à Google et est représenté par un petit robot de couleur vert pomme.

À la différence de Microsoft Windows et Mac, Android est un système d'exploitation open source, qui est d'ailleurs basé sur un noyau Linux. Cette particularité non négligeable permet à des développeurs d'analyser la manière dont est développé le téléphone et de pouvoir ainsi créer des applications dédiées. C'est justement cette propriété qui permet à App Inventor de pouvoir fonctionner et de vous permettre de tirer pleinement parti de l'ensemble des composants de votre téléphone.

Android est principalement présent sur les terminaux dits mobiles c'est-à-dire les smartphones et les tablettes. On le retrouve également sur les ordinateurs portables de Google, les Chromebook : <https://www.google.fr/chromebook/> dont la particularité réside dans le fait qu'il s'agit de terminaux extrêmement rapides mais quasiment intégralement dépendants d'Internet (la majeure partie des applications proposées ne fonctionneront pas si vous n'êtes pas connecté).

Ces caractéristiques font que sur les terminaux mobiles, Android jouit d'une très bonne place sur le marché. D'après <http://www.zdnet.fr/actualites/chiffres-cles-les-os-pour-smartphones-39790245.htm>, au premier trimestre 2016, Android possédait plus de 84 % de parts de marché au niveau mondial contre 14,8 % pour iOS.

En revanche il est quasiment inexistant sur le marché des ordinateurs classiques.

Qu'est-ce que le MIT ?

Le Massachusetts Institute of Technology (MIT) est un institut de recherche et une université américaine, spécialisée dans les domaines de la science et de la technologie.

Présentation d'applications déjà réalisées avec AI2

App Inventor 2 vous permet de créer autant d'applications que vous le souhaitez, vous êtes libre de laisser votre imagination vous emmener là où vous le désirez. À titre d'exemple, vous pouvez consulter l'ensemble des applications qui ont fait l'objet d'une distinction de la part de la communauté App Inventor 2.

Ces dernières sont disponibles dans la galerie App Inventor 2 : <http://appinventor.mit.edu/explore/app-month-gallery.html>

Congratulations to the Novemeber winners!

			
Randomizer by Bogdan Mihalca	Taking Notes by Tyson Seable	Qoogle by Arjun	A Leer Niños by Jose Sanmiguel
			
Romanian student Bogdan created a versatile app, Randomizer, with several features. It generates random numbers, lists, and coin flips. It also includes a color picker and a date picker.	Simple Notes offers an easy and intuitive way to take and store notes. It makes creative use of user interface components to create an elegant and customizable interface, which allows the user to set their desired font size and color scheme, and to require a PIN for access if desired.	Qoogle quizzes users on their knowledge of Google search shortcuts, using multiple gamification elements—including points, a timer, and a global leaderboard—in order to keep them engaged. These elements are integrated seamlessly into a sleek user interface that makes good use of layout, color scheme, and sound effects..	Jose, from Colombia, created a practical method for children and adults who are starting to read the Spanish language. He created the A Leer Niños for his son, who is learning to read. The app uses recorded phonetics to help the user learn sounds, words, and sentences.

Ces applications n'ayant pas été sélectionnées par hasard, vous verrez qu'en les installant, vous arriverez très rapidement à vous en inspirer pour de nouvelles applications.

Qu'est-ce qu'App Inventor 2 ne peut pas faire ?

App Inventor 2 ne vous permettra pas de développer toutes les applications que vous désirez. En effet, App Inventor 2 a pour objectif de rendre accessible la création d'applications mobiles à des personnes novices en programmation informatique. Ainsi, ne sont à votre disposition que les composants les plus accessibles, les plus populaires et les plus simples à utiliser. De la même manière, l'évolution technologique de cette plateforme dépend des efforts déployés par la communauté derrière celle-ci. Ainsi des fonctionnalités telles que le capteur de luminosité, le capteur de température, les notifications push, l'intégration du service de rémunération AdMob, le paiement en ligne... ne sont pas (encore) présentes dans App Inventor 2. En revanche, il n'est pas dit que vous puissiez les trouver dans d'autres distributions. Nous aurons l'occasion de revenir plus tard sur ce point. App Inventor 2 ayant pour objectif de rendre le développement informatique accessible à tous, les lignes de code sont remplacées par des briques de puzzle à imbriquer les unes dans les autres pour pouvoir décrire le comportement de l'application. Il n'est pas possible d'accéder à un terminal dans App Inventor 2 pour "ajouter à la main" des lignes de code.

Les conditions générales d'utilisation d'App Inventor 2

Tout d'abord, il est important de noter qu'App Inventor 2 fonctionne avec le service de Google du nom de "Google App Engine", c'est-à-dire que, sans la collaboration de ce dernier, le service ne pourrait voir le jour. En effet, Google a été à l'origine de ce qu'est aujourd'hui App Inventor 2.

Pour le moment, il est nécessaire d'avoir un compte Google pour créer ses applications, cependant à l'avenir, le MIT souhaite se détacher de cette obligation afin de permettre une liberté quasi totale à l'utilisateur.

À ce sujet, il est clairement indiqué dans les conditions générales d'utilisation que vous êtes propriétaire de ce que vous créez :

"MIT has no proprietary rights in the apps you create with MIT App Inventor. These apps belong to you. Your apps are stored on the MIT App Inventor server."

Le MIT ne pouvant s'engager pour vous sur la maintenance des applications il est de votre responsabilité de faire des sauvegardes régulières de vos projets sur votre ordinateur.

De la même manière, le MIT met un point d'honneur en préservant votre vie privée et en indiquant explicitement qu'aucune donnée personnelle que vous communiquerez ne sera utilisée à des fins commerciales.

Scratch : apprenez à développer sans savoir coder

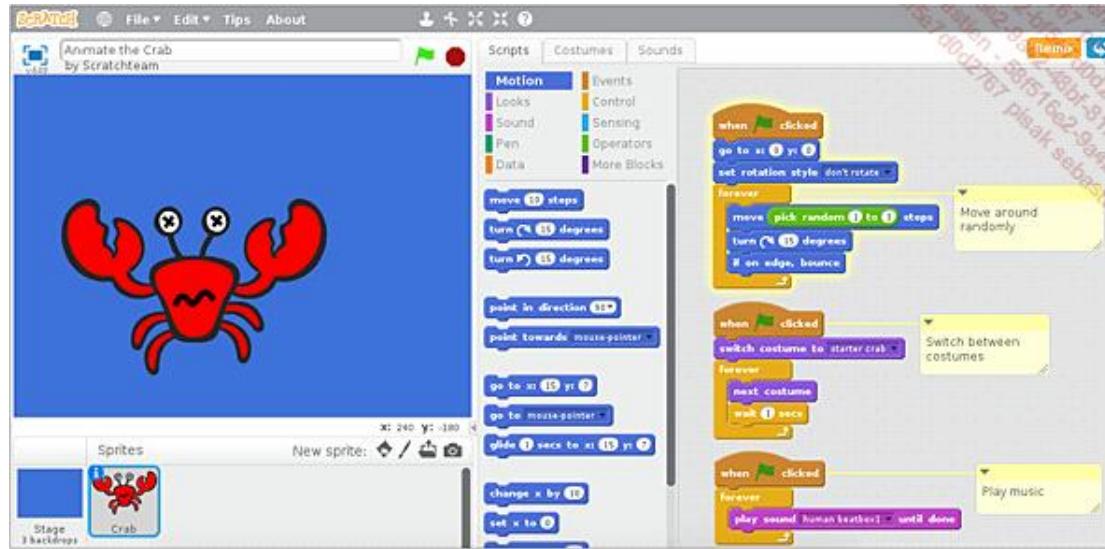
Les initiatives pour démocratiser l'utilisation de la pensée informatique sont légion. L'une d'entre elles, qui a su percer, est celle de Scratch. Scratch est un logiciel libre conçu par le MIT <https://scratch.mit.edu/> pour initier les élèves dès l'âge de 8 ans à des concepts fondamentaux en mathématiques et en informatique. Scratch est également une communauté dont l'objectif est de mettre en commun les programmes déjà développés par les uns pour permettre à d'autres des applications plus élaborées. On reconnaît notamment cette technologie car son emblème est un chat :



C'est d'ailleurs sur cette technologie que vont être formés de nombreux élèves, et ce dès le primaire en France à la rentrée 2016 dans un programme de formation appelé ClassCode : <https://pixees.fr/classcode/accueil/>.

Comme l'indique le créateur de Scratch Mitch Resnick dans cette vidéo : https://www.ted.com/talks/mitch_resnick_let_s_teach_kids_to_code?language=fr la programmation informatique concerne et va concerner toutes les couches de populations quel que soit votre âge ou votre situation professionnelle.

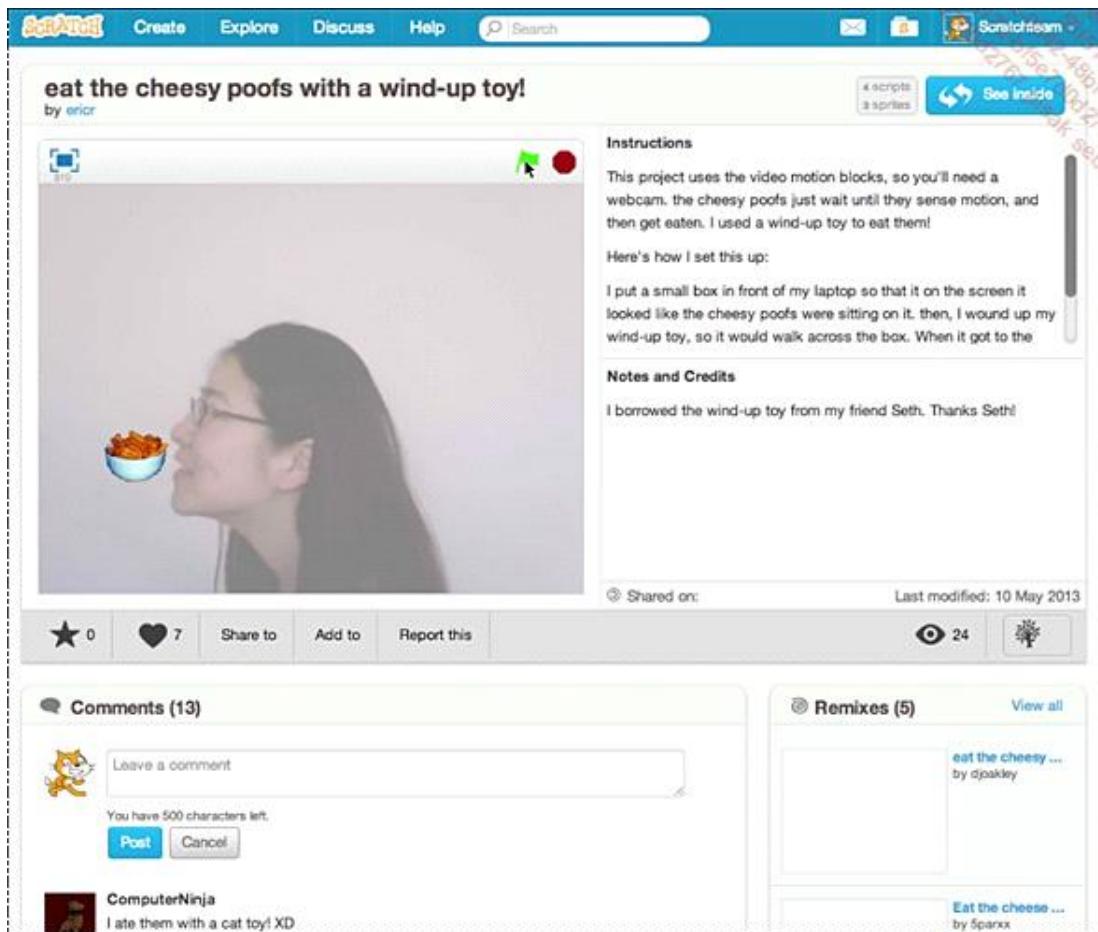
Scratch est une manière ludique et interactive d'apprendre facilement à coder en utilisant des briques logiques plutôt que des lignes de code informatique, par exemple ici dans le premier bloc d'instructions en haut à droite de l'écran :



Lorsque nous allons cliquer sur le drapeau vert (when clicked), notre crabe va se positionner aux coordonnées x et y indiquées, à savoir 0 et 0, et de façon continue celui-ci va effectuer des petites rotations de l'ordre de 15 degrés sans pour autant faire le tour de lui-même.

Scratch est une technologie passionnante qui nécessite la réalisation de nombreux projets pour en cerner toutes les possibilités. Ce qu'il faut principalement retenir, c'est que c'est accessible et que cela permet de réaliser des programmes complexes sans être développeur et quel que soit votre âge et vos connaissances.

Par exemple ci-dessous, l'impression écran vous présente un programme écrit avec Scratch permettant de faire reconnaître votre webcam et de la faire interagir avec des objets déposés dans l'interface :



Comme vous pouvez le constater ci-dessous, le code, l'ordre d'arrangement des briques est logique et compréhensible :

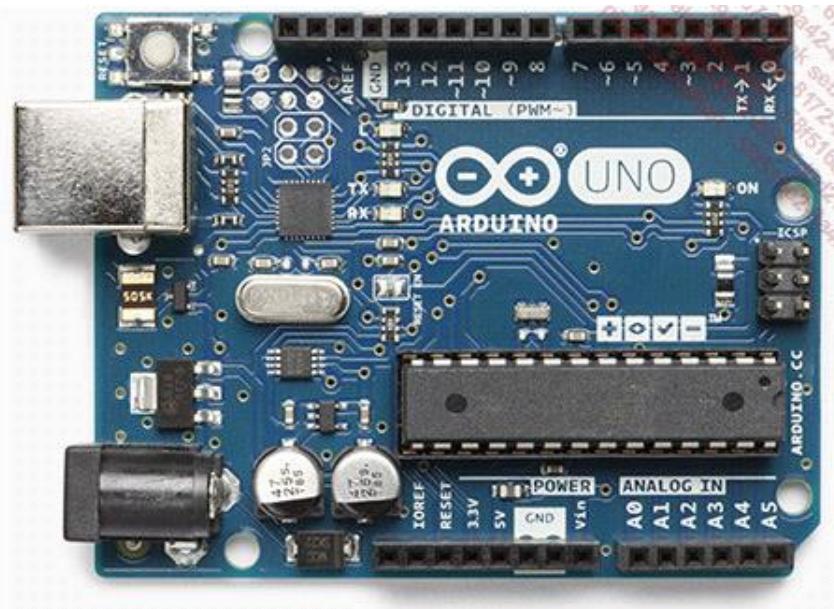


Vous pouvez retrouver directement cette application à l'adresse suivante et ainsi la tester :
<https://scratch.mit.edu/projects/10002240/>

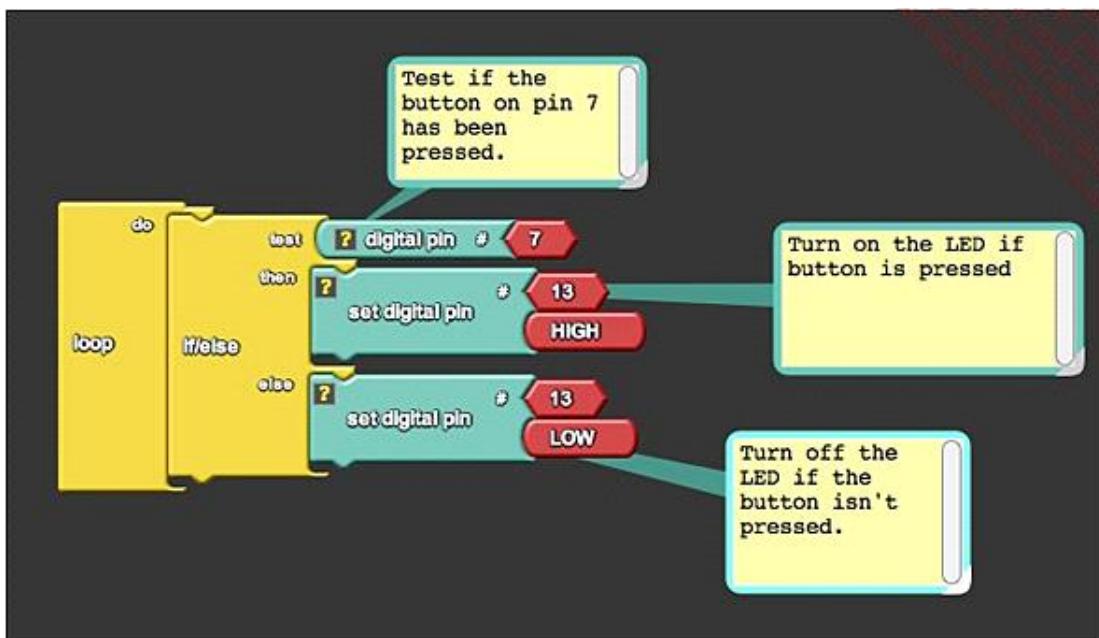
Le cours de formation relatif à Scratch en français qui sera présenté aux élèves du primaire est disponible ici :
<https://openclassrooms.com/courses/decouvrir-la-programmation-creative>.

Vous n'aurez pas nécessairement besoin de connaître le fonctionnement de Scratch pour avancer dans App Inventor 2, mais cela vous aidera à avancer bien plus vite et surtout à comprendre la logique derrière l'application que vous allez réaliser.

Par ailleurs, la logique que l'on a dans Scratch se retrouve aujourd'hui embarquée dans de nombreuses autres suites logicielles, par exemple dans l'univers des objets connectés, pour ceux et celles d'entre vous qui utilisent des cartes Arduino (l'un des matériaux de base lorsque l'on démarre dans l'univers des objets connectés) :



Il existe un programme appelé Ardublock, vous permettant d'écrire des séquences avancées sans en rédiger une seule ligne :



Conclusion

Dans ce chapitre, vous avez découvert ce qu'est App Inventor 2. Voyons désormais comment, en pratique, vous pouvez réaliser une application sur cette plateforme et directement en profiter.

Introduction

L'objectif de ce chapitre est simplement que vous puissiez prendre vos marques sur App Inventor 2. Vous allez réaliser votre première application sur Android en un temps record. Cette dernière n'aura rien d'extraordinaire en termes de fonctionnalités mais va vous permettre de voir les différentes étapes nécessaires à sa conception, de rentrer directement dans le vif du sujet.

Création d'une application sur AI2 en quelques minutes

L'application que nous allons créer va afficher une infobulle lorsqu'un utilisateur cliquera sur un bouton. Pour ceux et celles d'entre vous qui sont déjà familiers avec le langage de programmation JavaScript, l'application que nous allons réaliser va ressembler à :



Source de l'image : http://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_alert

Pour réaliser cette application, vous aurez besoin :

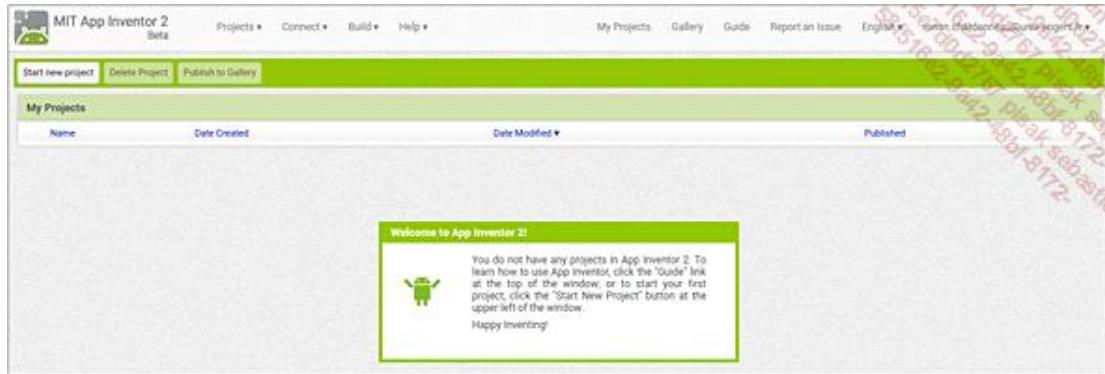
- d'un terminal mobile disposant d'Android 2.3 ou supérieur,
- d'un ordinateur avec une connexion à Internet,
- d'un compte Google,
- de l'application AIStarter téléchargeable sur le Play Store ou un lecteur de QR code classique.

→ La première étape de réalisation va consister à se connecter à l'URL suivante <http://ai2.appinventor.mit.edu/> à l'aide d'un navigateur.

Lors de votre connexion, il vous sera directement demandé le compte Google que vous souhaitez utiliser :



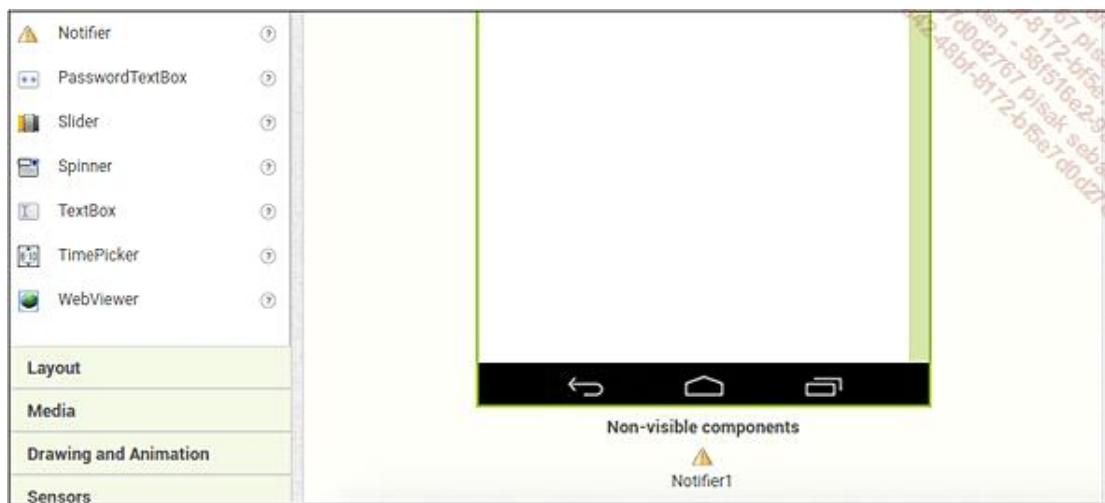
→ Une fois cela fait, vous devrez simplement accepter les conditions générales d'utilisation d'App Inventor afin de pouvoir accéder à l'interface d'App Inventor 2 :



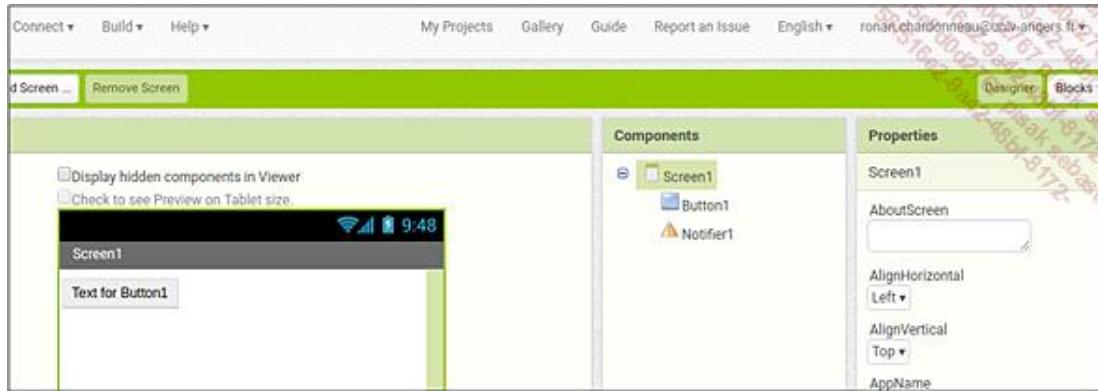
- Pour démarrer la création de votre application, cliquez sur **Start new project** et nommez votre projet, par exemple "MaPremiereApplication" (les espaces, les accents et les caractères spéciaux n'étant pas autorisés).
- Lorsque vous visualiserez l'écran ci-dessous, sélectionnez le composant bouton (**Button**) (le premier dans la liste de gauche) et déplacez-le vers l'écran blanc :



- Refaites la même manipulation, mais avec le composant **Notifier**. Il s'agit du huitième composant (le panneau triangulaire d'alerte ⚠) :



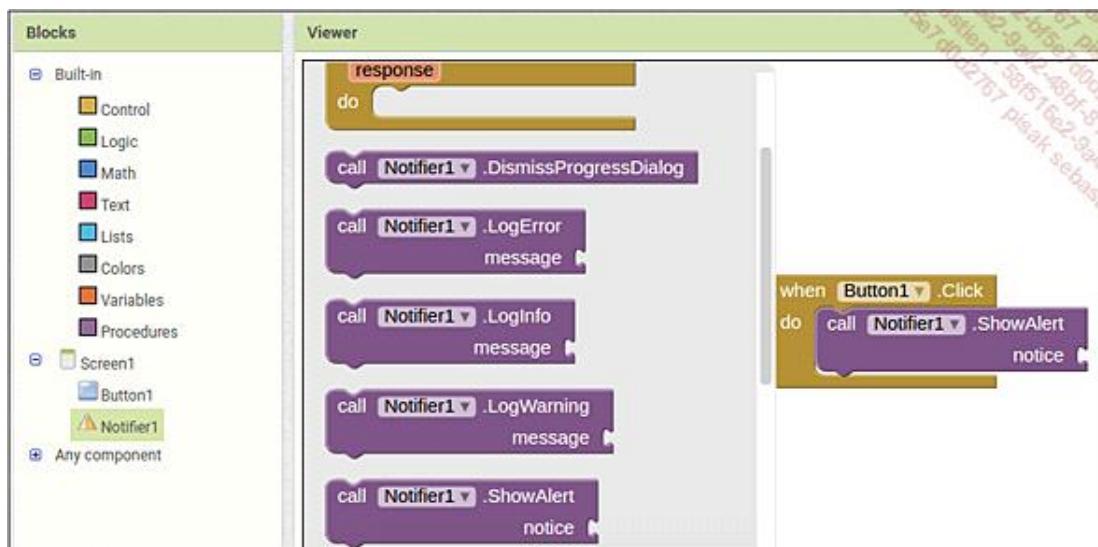
- Nos deux composants étant désormais mis en place dans notre application, nous n'avons plus qu'à définir leurs comportements : pour ce faire, cliquez en haut à droite sur le bouton **Blocks** :



- Une fois cela fait, cliquez sur l'élément **bouton** (**Button**) dans la colonne de gauche et faites-glisser le premier élément (**when Button1.Click do**) dans la zone blanche sur la droite :



- Une fois cela fait, cliquez sur l'élément **Notifier1** et cliquez-déposez le sixième élément **Notifier1.ShowAlert notice** en l'imbriquant dans l'élément **Button1.Click** :



- Dans la colonne de gauche, cliquez maintenant sur le carré violet **Text** et sélectionnez le premier élément pour l'imbriquer dans l'élément **Notifier1.ShowAlert.notice**. Une fois cela fait, cliquez dans l'élément texte

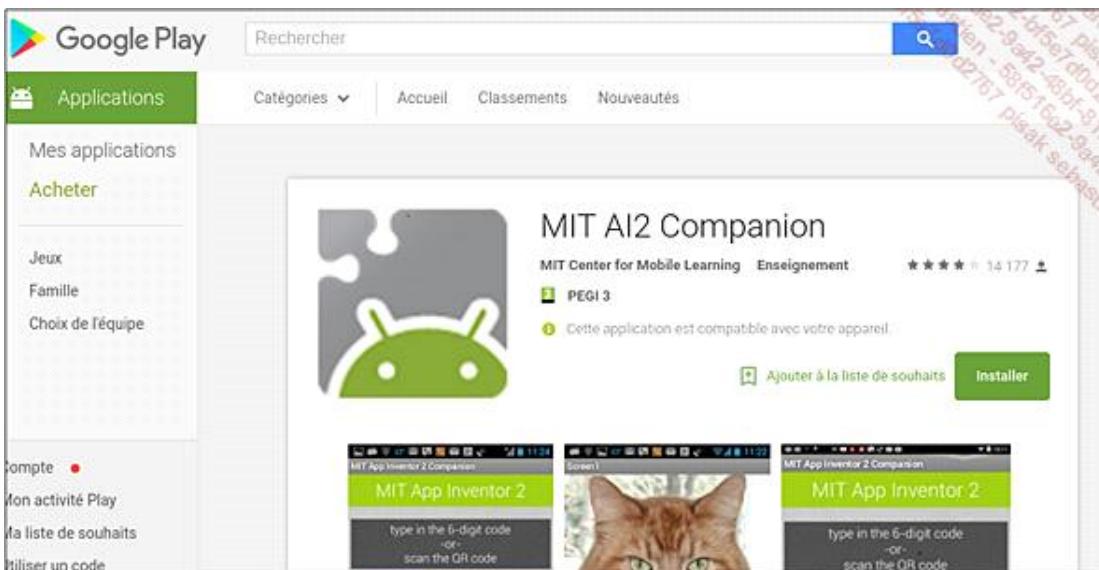
que vous venez d'ajouter et écrivez la phrase suivante : **Vous venez de réaliser votre 1ère application**, vous devriez désormais avoir la configuration suivante :



Celle-ci nous indique que lorsque le bouton est cliqué, l'application va afficher une fenêtre d'alerte avec le message suivant : "Vous venez de réaliser votre 1ère application".

Notre application est désormais réalisée en termes de design et de code, il ne nous reste plus qu'à lui donner vie.

- Afin de pouvoir installer cette application sur votre téléphone ou tablette, vous allez devoir scanner un QRCode, vous pouvez pour cela télécharger l'application MIT AI2 Companion sur le Play Store de Google à l'adresse <https://play.google.com> :



- Une fois installée sur votre terminal, cliquez sur **Build** puis **App (provide QR code for .apk)**. Cela va alors générer un QR Code qu'il faudra flasher avec l'application MIT AI2 Companion :



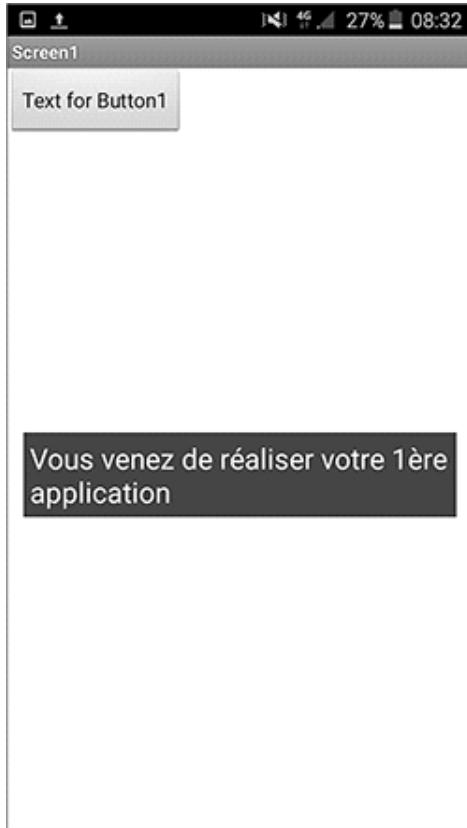
- Une fois l'application téléchargée, votre téléphone vous demandera si vous souhaitez vraiment installer cette application.



- Cliquez alors sur le bouton **PARAMÈTRES**.
- Dans la partie relative à la sécurité, activez l'option autorisant les **Sources inconnues** :



- Cliquez ensuite sur **Installer** puis **Ouvrir** pour accéder à votre application. Vous n'avez plus qu'à cliquer sur le bouton pour voir la fenêtre apparaître :



Félicitations, vous venez officiellement de créer votre première application Android grâce à App Inventor 2.

Voici ce que vous avez dû constater lors de la réalisation de ce tutoriel :

- la création d'une première version d'une application se fait en quelques minutes ;
- la réalisation de la partie code semble tout à fait logique ;
- App Inventor 2 semble facile à prendre en main ;
- de nombreuses autres fonctionnalités semblent vous permettre d'aller beaucoup plus loin ;
- le design de l'application n'est pas très avancé ;
- votre smartphone possède un minimum de sécurité et ne permet pas d'installer des applications en lesquelles il n'a pas confiance.

Nous allons justement voir en détail l'ensemble de ces fonctionnalités dans les chapitres suivants.

Conclusion

Ce que vous avez appris durant ce chapitre :

- Comment réaliser une première application en quelques minutes avec App Inventor 2.

Ce chapitre est maintenant terminé. À partir de celui-ci vous pouvez déjà décider de prendre votre envol dans App Inventor 2 et commencer à créer vos propres applications. Cependant, et comme vous pourrez le découvrir par la suite, de nombreuses connaissances vous seront nécessaires pour aller plus vite, mieux structurer votre travail et surtout découvrir les ressources cachées de cette technologie.

Introduction

Ce chapitre a pour objectif de vous aider à structurer votre travail. Comme le dit le vieil adage, un problème bien défini est un problème à moitié résolu, et cela s'applique à l'univers des applications pour mobile. Nous allons voir dans ce chapitre comment aborder les points essentiels permettant le développement d'applications sans contrainte. Vous aurez appris à la fin de celui-ci comment gagner un temps considérable et éviter les principaux écueils.

Pourquoi réaliser un cahier des charges ?

Tout comme la création d'un site Internet, une application sur mobile nécessite d'avoir en amont un "cahier des charges", en effet celui-ci va vous permettre d'éviter de nombreux écueils. Par exemple, en mettant "sur le papier" vos idées, vous allez vous rendre compte qu'il existe des manières plus simples de réaliser votre projet. Vous allez également constater qu'en listant vos besoins, des éléments qui vous paraissaient complexes ne sont en fait pas du tout insurmontables. Vous allez constater que la manière dont vous aviez initialement prévu de réaliser votre travail peut en réalité être faite de manière différente. Vous pourriez également réaliser qu'il va vous falloir des ressources, que ce soit en termes de finance (PlayStore, graphiste) ou en termes de temps.

Travailler à la rédaction d'un cahier des charges est donc un gain de temps considérable.

Les différentes manières de réaliser une application sur mobile

Vous pouvez travailler directement sur App Inventor 2 et voir où cela va vous mener. C'est notamment la méthode de travail que vous utilisez lorsque vous êtes passionné et que vous développez des petits projets personnels.

Avantage :

- Vous allez rapidement apprendre et obtenir un résultat.

Inconvénient :

- Votre application va malheureusement être très différente de ce que vous souhaitez obtenir initialement.

Vous pouvez également travailler en méthode agile, c'est-à-dire que vous allez réaliser un cahier des charges très succinct en priorisant les fonctionnalités les plus importantes. Il s'agit d'une méthode de travail très pratiquée dans le domaine du numérique.

Avantages :

- Vous avez un minimum de planification devant vous et pouvez prévoir les évolutions futures de votre produit.
- Vous aurez assez rapidement une application à montrer.

Inconvénient :

- Il va vous falloir un minimum de rigueur et de méthodologie de travail.

Vous pouvez également utiliser la méthode dite "waterfall", en réalisant un cahier des charges très exhaustif. Il s'agit principalement de la méthode utilisée pour de gros projets ou dans le monde professionnel afin de se protéger juridiquement.

Avantage :

- Vous avez une vision très claire de l'ensemble de votre projet, il n'y a plus qu'à suivre le guide.

Inconvénient :

- Les temps de réalisation sont généralement très longs et vous risquez de ne jamais voir sortir votre projet.

Que doit contenir un cahier des charges pour la réalisation d'une application pour mobile ?

Il est toujours difficile de déterminer avec exactitude tout ce que doit contenir un cahier des charges, certaines informations étant applicables au monde professionnel, d'autres pas. Dans notre cas de figure, nous avons identifié les grandes catégories suivantes pour le développement d'une application mobile :

- Informations sur le donneur d'ordre
- Les besoins du donneur d'ordre
- Identification de la cible
- Aspect éditorial
- Aspect fonctionnel
- Aspect graphique
- Aspect juridique
- Business model
- Promotion de l'application
- Ressources consacrées

Informations sur le donneur d'ordre

Ces informations vous permettent de prendre du recul par rapport à la personne pour qui vous réalisez l'application mobile. Par exemple, si vous la créez pour une association, vous allez vous rendre compte que vous pouvez collecter sur cette dernière tout un ensemble d'informations : brochures, cartes de visite, logo, activité de l'entreprise, description des produits et services... Le recueil de ces informations va vous permettre déjà de vous projeter et d'imaginer la manière dont vous allez concevoir l'application aussi bien sur la partie graphique que sur les contenus à apporter.

Les besoins du donneur d'ordre

Il s'agit ici d'identifier les besoins auxquels il faut que l'application réponde. Il peut y avoir deux types de besoins, à la fois ceux du donneur d'ordre et ceux du consommateur final. Ces besoins vont vous permettre de définir les objectifs que doit réaliser votre application. Par exemple : permettre une réservation facile d'une chambre d'hôtel, calculer facilement la distance parcourue par jour d'un utilisateur, garder une trace d'appels téléphoniques passés.

Vérifiez bien que ces objectifs sont SMART, c'est-à-dire que vous êtes en mesure de les réaliser aussi bien au niveau fonctionnel qu'en termes de délai. Ici le secret est de vous projeter et de vous assurer que l'application que vous allez réaliser répond vraiment à un besoin. Projetez-vous, faites comme si vous la possédiez déjà. Vous identifierez ainsi si elle vaut le coup d'y consacrer votre temps et votre énergie.

Identification de la cible

Il faut savoir vous mettre à la place des consommateurs finaux, ceux qui utiliseront l'application que vous allez développer. Mieux vous connaîtrez cette cible, mieux vous serez à même de réaliser une application performante. Faites le dessein de votre cible et créez une application pour eux. Identifiez la manière dont ils vont chercher votre application, ce qu'ils taperaient dans les moteurs de recherche, la manière dont ils l'utiliseront, les lieux qu'ils sont susceptibles de visiter et où devrait apparaître votre communication, les supports que vous leur diffuserez.

Aspect éditorial

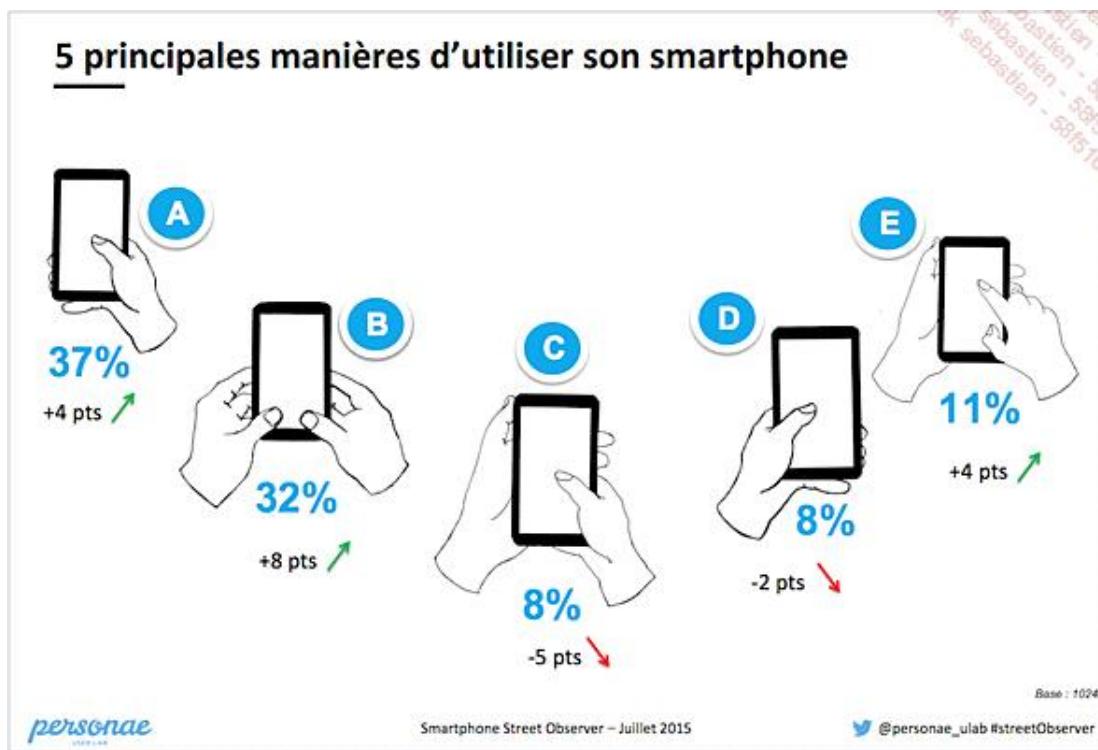
Vous allez dans tous les cas devoir produire du contenu textuel dans votre application (écran d'accueil, écran de configuration, crédits, divers textes). Tous ces aspects doivent être déjà pensés afin d'avoir une vision claire de ce que vous souhaitez obtenir.

Aspect fonctionnel

Quels sont les composants que vous souhaitez voir inclus au sein de votre application ? Quelles sont les fonctionnalités que vous souhaitez avoir ? Pouvez-vous déjà imaginer la manière dont le code va être écrit ? Dans cette partie du cahier des charges, vous pourriez même déjà réaliser votre application en dressant sur le papier sa composition interne.

Aspect graphique

L'aspect graphique va vous permettre d'accompagner le développement de l'aspect éditorial et fonctionnel. Beaucoup d'utilisateurs vont vous juger sur les aspects graphiques de votre application. De la même manière, l'expérience utilisateur va jouer un rôle prépondérant. Par exemple, où allez-vous placer les éléments graphiques de votre application ?



L'image ci-dessus est issue de l'étude de l'usage du smartphone en France, réalisée par l'entreprise Personae.

Ainsi, vous êtes libre d'utiliser des logiciels tels que GIMP, Photoshop... pour matérialiser vos attentes ou d'autres logiciels en ligne tels que Moqups (<https://moqups.com/>), Balsamiq (<https://balsamiq.com>) ou encore Marvel (<https://marvelapp.com/>). Dans tous les cas, le plus important sur cette partie est de pouvoir dessiner ce qui vous trotte dans la tête. Et dans ces moments-là, peu importe l'outil, un stylo et un papier font également l'affaire. En mettant à plat votre projet, vous réaliserez probablement que ce que vous aviez en tête initialement n'est finalement pas possible, peut être amélioré et va déboucher sur une bien meilleure application. C'est justement tout le but de ce cahier des charges.

Aspect juridique

Avez-vous le droit de faire ce que vous êtes en train de développer ? Peut-être que oui, peut-être que non. L'aspect juridique de votre cahier des charges va englober tout un ensemble de variables : l'utilisation finale de votre application, le droit des images, la sécurité de votre application... Essayez-vous de copier une œuvre existante ? Tout cela doit nécessairement faire l'objet d'une réflexion en amont.

Business model

S'il s'agit d'une application développée dans un but personnel ou à destination d'une association, vous n'aurez pas de souci à vous faire sur cette partie. En revanche, si votre objectif est d'ordre économique, vous allez devoir penser au modèle économique de votre application. Comment allez-vous rentabiliser cette dernière ? Est-ce une application avec un prix fixe ? Un abonnement ? Avec de l'affichage publicitaire ? Des dons ? Tout doit être pensé et défini à l'avance.

Promotion de l'application

Comment allez-vous faire connaître votre application ? Bouche-à-oreille ? Affichage publicitaire ? Affiliation ? Par l'intermédiaire d'un site Internet ? En vous associant à d'autres applications ?

Ressources consacrées

Il faut ici imaginer le mot "ressource" au sens large du terme. Ici, ce qui nous intéresse c'est de savoir si le projet que vous souhaitez réaliser est cohérent en fonction des ressources que vous allez mettre en place. Si vos ressources se résument à 60 minutes de travail tous les soirs sur une semaine, alors vous savez déjà qu'il sera probablement difficile de réaliser une application commercialisable au bout d'un mois capable de se positionner parmi les best-sellers. De la même manière, afin de pouvoir référencer votre application dans le Play Store, vous verrez qu'il y a "un pas de porte" à franchir de l'ordre de 25 €. Peut-être qu'App Inventor ne possédera pas toutes les fonctionnalités dont vous avez besoin pour mener à bout votre projet et il vous faudra engager un développeur.

Vous connaissez désormais l'intérêt de rédiger un cahier des charges et êtes en mesure de compléter le vôtre afin d'envisager le développement d'une application mobile sur Android.

Exemple de cahier des charges

Le cahier des charges que nous allons vous présenter ci-dessous est celui d'une application pour mobile simple et fonctionnelle. Il s'agira de réaliser un niveau :



Il existe déjà beaucoup d'applications de ce type, mais ce qui nous intéresse surtout ici c'est le cahier des charges.

Informations sur le donneur d'ordre

Une entreprise du secteur de la construction qui souhaite mettre à disposition de son personnel un outil de poche simple et fonctionnel à l'image de leur entreprise.

Les besoins du donneur d'ordre

Avoir une application permettant en un clic de vérifier si un plan est droit.

Identification de la cible

Professionnels du bâtiment : commercial, ingénieur, ouvrier ayant oublié leurs outils de chantier.

Aspect éditorial

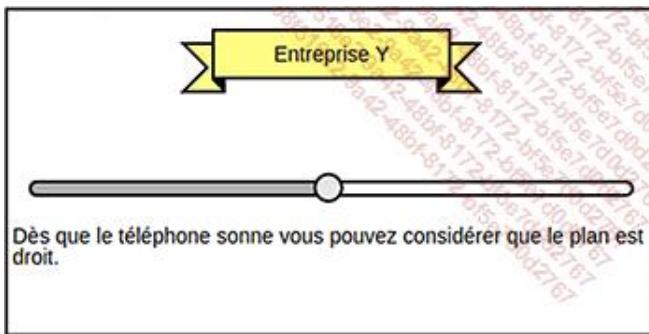
Un simple texte d'explication du fonctionnement de l'application accompagné du logo de l'entreprise.

Aspect fonctionnel

Dès le lancement de l'application, celle-ci doit montrer un curseur qui bouge en continu en fonction de l'inclinaison du mobile. Dès que celui-ci indique une inclinaison comprise entre -1 et un degré alors le téléphone doit se mettre à vibrer pour indiquer que le plan mesuré peut être considéré comme droit.

Aspect graphique

Simple, coloré et à l'image de l'entreprise :



Dès que le téléphone sonne vous pouvez considérer que le plan est droit.

Aspect juridique

L'application utilisera l'ensemble des technologies qui composent App Inventor 2.

Business model

L'idée de l'entreprise est de fournir une application propre à son entreprise afin que son personnel n'utilise pas devant ses clients des applications qui ne viennent pas de concurrents. Il s'agit avant tout d'une question d'image de marque.

Promotion de l'application

Afin que l'entreprise puisse se faire connaître, il est prévu de référencer cette dernière sur le Play Store afin que n'importe quel utilisateur de mobile puisse utiliser cette dernière. Elle sera également présentée dans le catalogue offert à tous les clients de l'entreprise dans la catégorie Outils.

Ressources consacrées

Une demi-journée de travail pour le "développeur".

Conclusion

Vous savez désormais comment documenter et mettre sur papier vos idées pour le développement d'une application pour mobile. Voyons désormais plus en détail comment bien démarrer avec la plateforme App Inventor 2.

Introduction

Il existe de nombreuses manières de travailler avec App Inventor 2. Vous verrez d'ailleurs que vos besoins vont évoluer au cours de son utilisation. Par exemple, d'une simple application à créer sur un coin de table, au développement d'une application dans un milieu professionnel, en passant par un projet que vous souhaitez développer en toute tranquillité, coupé du monde extérieur.

Dans ce chapitre, nous allons vous présenter les différentes configurations sous lesquelles vous pouvez travailler avec App Inventor 2.

Attention : certains points de ce chapitre pourront vous paraître très techniques ou du moins difficiles à réaliser concrètement. Le secret est de prendre son temps et à chaque fois de relire calmement les consignes.

À la fin de celui-ci, vous serez en mesure de connaître les différentes manières de travailler avec App Inventor 2, telles que la mise en place d'AI 2 sur votre machine via un serveur local, la mise en place d'un émulateur, ou émuler directement sur votre smartphone l'application que vous êtes en train de créer.

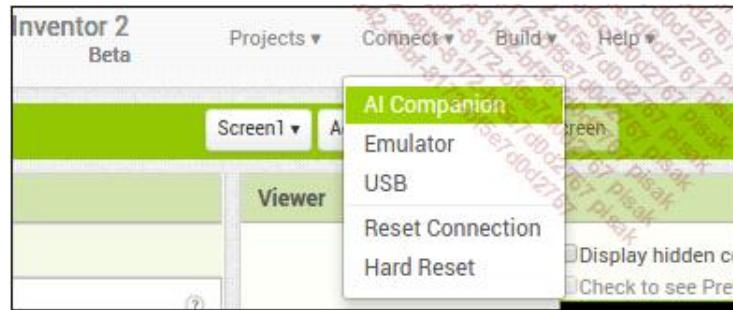
Mise à jour de l'application en temps réel

Dans le chapitre précédent, Création d'une application sur AI 2 en quelques minutes, nous avons vu comment installer très rapidement une application sur mobile en flashant un QR code. Sachez que vous pouvez aller bien plus vite en connectant votre ordinateur directement à votre mobile via un câble USB ou en mettant directement en place un émulateur sur votre ordinateur. L'une et l'autre de ces solutions vont vous permettre d'être plus rapide dans le développement de vos applications.

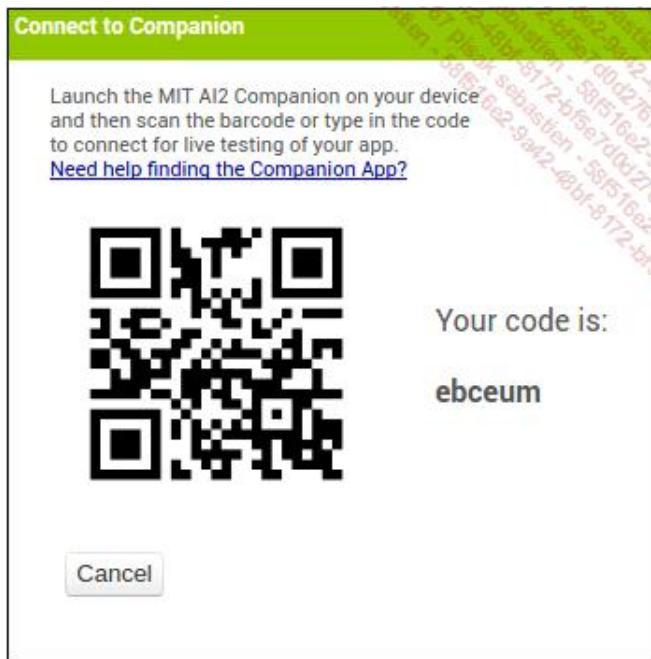
La manière la plus rapide de voir l'évolution de votre application est de connecter votre appareil mobile via un code grâce à l'application AI Companion.

- Pour ce faire, vous devrez utiliser l'application officielle que vous trouverez sur le PlayStore à l'adresse suivante : <https://play.google.com/store/apps/details?id=edu.mit.appinventor.ai companion3&hl=fr>

Lorsque vous réaliserez votre application, cliquez sur **Connect** puis sur **AI Companion** :



Une fenêtre va apparaître vous indiquant un code à six caractères :



Ce code de six caractères est alors à entrer dans l'application officielle d'App Inventor 2 pour vous connecter directement en temps réel à l'application que vous êtes en train de développer.



Durant cette phase, il est fort possible que vous rencontriez divers messages d'erreurs, vous trouverez des éléments de réponse sur le lien suivant : <http://appinventor.mit.edu/explore/ai2/support/troubleshooting.html>. Le problème le plus souvent rencontré est lorsque votre ordinateur n'utilise pas le même réseau Wi-Fi que celui de votre appareil mobile. Pour résoudre le problème, connectez les deux appareils sur le même réseau Wi-Fi.

Vous pouvez également relier votre ordinateur à votre smartphone ou tablette via un câble USB. Cependant, les manipulations à effectuer sont bien plus complexes et dépendent de votre smartphone et du système d'exploitation de votre ordinateur. Par ailleurs, de par sa grande complexité, les créateurs d'App Inventor 2 ne recommandent pas son utilisation et vous conseillent la connexion via Wi-Fi. Pour les plus téméraires d'entre vous, voici la documentation officielle : <http://appinventor.mit.edu/explore/ai2/setup-device-usb.html>

Pour les utilisateurs Windows

Prérequis pour l'installation : Windows XP ou supérieur ainsi que le navigateur Google Chrome ou Firefox. Il vous faudra également avoir les droits d'administration sur la machine Windows que vous allez utiliser.

Voici les étapes à suivre pour l'installation d'App Inventor 2 sur votre ordinateur :

- Téléchargez le fichier à l'adresse suivante : http://appinv.us/aisetup_windows
Un fichier du nom de MIT_App_Inventor_Tools_2.3.0 ou similaire devrait être téléchargé, taille approximative : 80 MB.
- Une fois téléchargé, double cliquez dessus pour lancer l'installation.
Nous vous conseillons de laisser le chemin d'installation tel que proposé.
- Lors de l'installation, il vous sera demandé si vous autorisez l'installation d'un programme inconnu, cliquez alors sur **oui**.

App Inventor 2 est désormais installé sur votre ordinateur ; si jamais vous rencontrez des problèmes lors de ces étapes, le MIT a mis à votre disposition un lien d'aide à la page suivante : <http://appinventor.mit.edu/explore/ai2/support/troubleshooting.html>

Installation de App Inventor 2 sur GNU/Linux

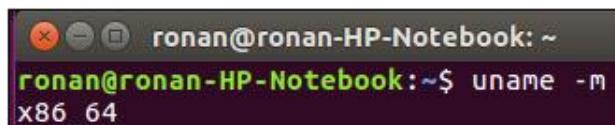
La procédure d'installation sur le site est en anglais : <http://appinventor.mit.edu/explore/ai2/linux.html>. Du coup, voici la procédure à suivre pour l'installer, rédigée en français :

Comme pour toute installation sur Linux il faudra vous mettre en super-administrateur pour effectuer l'installation.

Le programme AI 2 pour Linux fonctionne pour un ordinateur en 32 bits. Si votre ordinateur est en 64 bits, vous devrez installer la librairie lib32z1 avec la commande `sudo apt-get install lib32z1`.

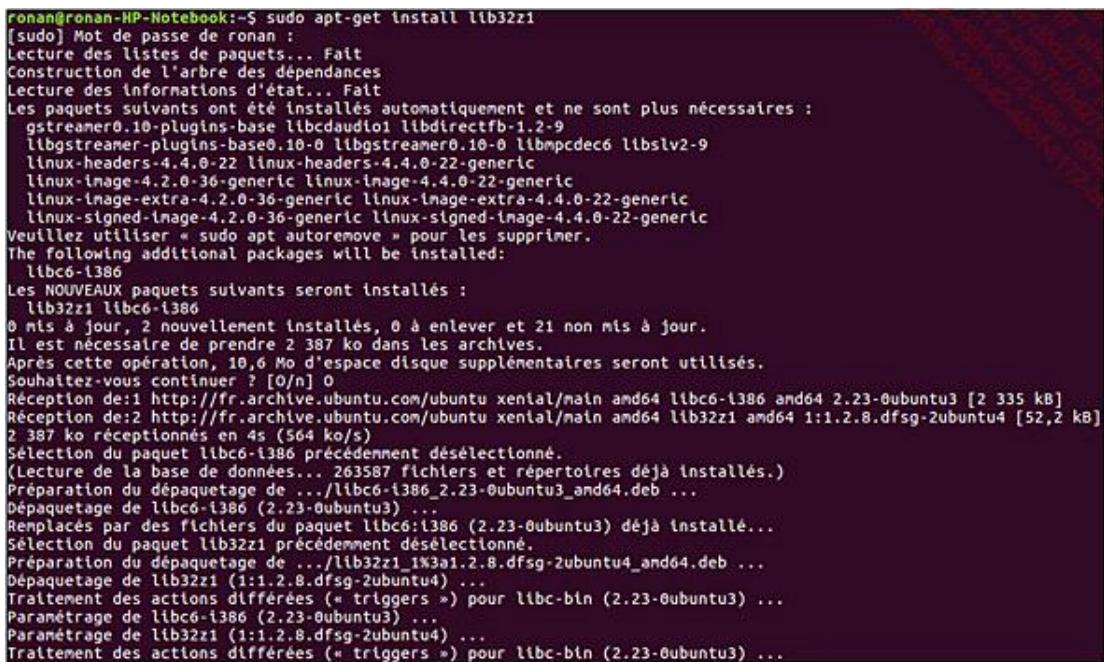
Comment vérifier si mon ordinateur est en 32 bits ou 64 bits ?

- Entrez simplement la commande `uname -m` (elle permet d'afficher les informations système sur la machine sur laquelle elle est exécutée) :



```
ronan@ronan-HP-Notebook:~$ uname -m
x86_64
```

qui vous indiquera directement votre configuration ; dans le cas ci-dessus il s'agit d'un 64 bits et va nécessiter l'installation des librairies ci-dessus.



```
ronan@ronan-HP-Notebook:~$ sudo apt-get install lib32z1
[sudo] Mot de passe de ronan :
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
gstreamer0.10-plugins-base libcdaudio1 libdirectfb-1.2-9
libgststreamer-plugins-base0.10-0 libgststreamer0.10-0 libmpcdec6 libslv2-9
linux-headers-4.4.0-22 linux-headers-4.4.0-22-generic
linux-image-4.4.0-36-generic linux-image-4.4.0-22-generic
linux-image-extra-4.4.0-36-generic linux-image-extra-4.4.0-22-generic
linux-signed-image-4.4.0-36-generic linux-signed-image-4.4.0-22-generic
Veuillez utiliser « sudo apt autoremove » pour les supprimer.
The following additional packages will be installed:
libc6-i386
Les NOUVEAUX paquets suivants seront installés :
lib32z1 libc6-i386
0 mis à jour, 2 nouvellement installés, 0 à enlever et 21 non mis à jour.
Il est nécessaire de prendre 2 387 ko dans les archives.
Après cette opération, 10,6 Mo d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] O
Réception de:1 http://fr.archive.ubuntu.com/ubuntu xenial/main amd64 libc6-i386 amd64 2.23-0ubuntu3 [2 335 kB]
Réception de:2 http://fr.archive.ubuntu.com/ubuntu xenial/main amd64 lib32z1 amd64 1:1.2.8.dfsg-2ubuntu4 [52,2 kB]
2 387 ko réceptionnés en 4s (564 ko/s)
Sélection du paquet libc6-i386 précédemment désélectionné.
(Lecture de la base de données... 263587 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de .../libc6-i386_2.23-0ubuntu3_amd64.deb ...
Dépaquetage de libc6-i386 (2.23-0ubuntu3) ...
Remplacés par des fichiers du paquet libc6:i386 (2.23-0ubuntu3) déjà installé...
Sélection du paquet lib32z1 précédemment désélectionné.
Préparation du dépaquetage de .../lib32z1_1%3a1.2.8.dfsg-2ubuntu4_amd64.deb ...
Dépaquetage de lib32z1 (1:1.2.8.dfsg-2ubuntu4) ...
Traitement des actions différées (« triggers ») pour libc-bin (2.23-0ubuntu3) ...
Paramétrage de libc6-i386 (2.23-0ubuntu3) ...
Paramétrage de lib32z1 (1:1.2.8.dfsg-2ubuntu4) ...
Traitement des actions différées (« triggers ») pour libc-bin (2.23-0ubuntu3) ...
```

- Si jamais vous avez essayé par le passé d'installer App Inventor sur votre ordinateur, vous devez désinstaller les programmes concernés. Pour ce faire, entrez les lignes de commandes suivantes :

```
sudo rm -rf /usr/google/appinventor (permet d'enlever les répertoires liés au programme)
```

```
sudo rm -rf ~/.appinventor (permet d'enlever les fichiers d'exécution)
```

```
sudo apt-get remove appinventor-setup (permet d'enlever le programme d'installation)
```

- Une fois ces actions effectuées, téléchargez le fichier situé à l'adresse suivante : http://appinv.us/aisetup_linux_deb

en ligne de commande : wget http://appinv.us/aisetup_linux_deb

```
ronan@ronan-HP-Notebook:~$ wget http://appinv.us/aisetup_linux_deb
--2016-07-13 16:23:14--  http://appinv.us/aisetup_linux_deb
Résolution de appinv.us (appinv.us)... 23.236.48.82
Connexion à appinv.us (appinv.us)|23.236.48.82|:80... connecté.
requête HTTP transmise, en attente de la réponse... 302 FOUND
Emplacement : http://128.52.184.65/appinventor2-setup_2.3_all.deb [suivant]
--2016-07-13 16:23:15--  http://128.52.184.65/appinventor2-setup_2.3_all.deb
Connexion à 128.52.184.65:80... connecté.
requête HTTP transmise, en attente de la réponse... 200 OK
Taille : 75112754 (72M) [application/x-debian-package]
Enregistre : >aisetup_linux_deb

aisetup_linux_deb          100%[=====] 71,63M  1,01MB/s   in 87s

2016-07-13 16:24:42 (840 KB/s) - `aisetup_linux_deb` enregistré [75112754/75112754]
```

Si la distribution Linux que vous utilisez accepte l'installation par clic, alors cliquez simplement sur le fichier, sinon suivez les indications suivantes :

- Veuillez vous rendre en ligne de commande dans le répertoire où le fichier se trouve et effectuez la commande suivante : sudo dpkg --install appinventor2-setup_2.3_all.deb



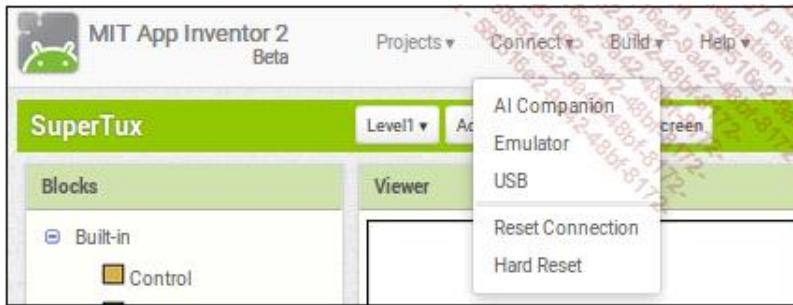
Le programme est désormais installé au chemin suivant : /usr/google/appinventor

- À partir de cette étape, vous devriez déjà être en mesure de lancer l'émulateur avec la commande suivante :

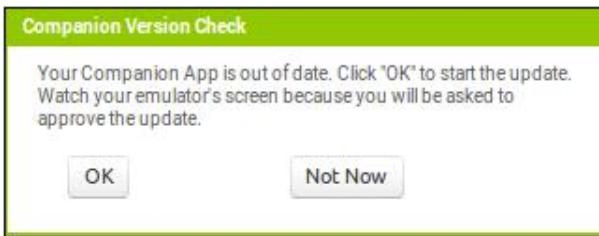
/usr/google/appinventor/commands-for-Appinventor/aiStarter &

```
ronan@ronan-HP-Notebook:~/AI2$ /usr/google/appinventor/commands-for-Appinventor/aiStarter &
[1] 4333
ronan@ronan-HP-Notebook:~/AI2$ Bottle server starting up (using WSGIRefServer())...
Listening on http://127.0.0.1:8004/
Hit Ctrl-C to quit.
```

- Lorsque vous visualisez le message indiquant **Listening on http://127.0.0.1:8004/**, cela signifie que vous pouvez vous rendre à l'adresse <http://ai2.appinventor.mit.edu> afin de connecter App Inventor 2 à votre ordinateur qui va alors l'émuler. Pour ce faire, allez dans une application que vous avez créée et cliquez sur **Connect** puis **Emulator** :



Soyez patient lors de la connexion ; en effet, émuler App Inventor 2 sur votre ordinateur sera gourmand en termes de ressources. Cela peut prendre jusqu'à cinq minutes pour que le tout s'installe. Il est également possible qu'un message vous demandant de mettre à jour votre émulateur s'affiche :



Pour faire reconnaître l'appareil par le système si vous êtes sous Linux :
<https://developer.android.com/studio/run/device.html>

Il va falloir ajouter ce que l'on appelle une règle "udev". Pour en savoir plus : <https://doc.ubuntu-fr.org/udev>. Pour faire simple, cela sert à faire reconnaître votre périphérique.

Afin de permettre que votre appareil soit trouvé :

- Accédez à un terminal en mode root, en utilisant la commande `su root` :

```
ronan@ronan-HP-Notebook:~$ su root
Mot de passe :
root@ronan-HP-Notebook:/home/ronan# 
```

- Une fois cela fait, entrez la ligne de commande suivante :

```
> /etc/udev/rules.d/51-android.rules
```

Ceci va permettre la création du fichier 51-android.rules dans le répertoire rules.d.

Une fois celui-ci créé, il va falloir le modifier pour y ajouter les lignes permettant de faire reconnaître votre appareil.

- Pour ce faire, utilisez le lien de référence suivant :
<https://developer.android.com/studio/run/device.html#VendorIds>

The screenshot shows the Android Studio User Guide with the 'User Guide' section selected. A table lists various vendor names and their corresponding USB Vendor IDs:

Nvidia	0955
OTGV	2257
Pantech	10a9
Pegatron	1d4d
Philips	0471
PMC-Sierra	04da
Qualcomm	05c6
SK Telesys	1f53
Samsung	04e8
Sharp	04dd
Sony	054c
Sony Ericsson	0fce

- Repérez ensuite l'USB Vendor ID correspondant à la marque de votre téléphone. Dans le cas ci-dessus pour la marque Samsung, l'identifiant est **04e8**.

Pour notre exemple, nous allons donc ajouter la ligne suivante à notre fichier et le sauvegarder :

```
SUBSYSTEM=="usb", ATTR{idVendor}=="04e8", MODE="0666",
GROUP="plugdev"
```

La partie `MODE="0666"` correspond aux droits accordés à l'utilisateur à savoir lire et écrire, la partie `GROUP="plugdev"` indique le groupe auquel ces droits vont être attribués, si vous êtes le seul à utiliser votre ordinateur, le groupe n'est composé que de vous.

- Une fois cela fait, vous devez exécuter la ligne de commande suivante :

```
chmod a+r /etc/udev/rules.d/51-android.rules
```

`chmod` étant une ligne de commande permettant de changer les permissions d'accès au fichier.

- Relancez ensuite l'application avec la commande suivante :

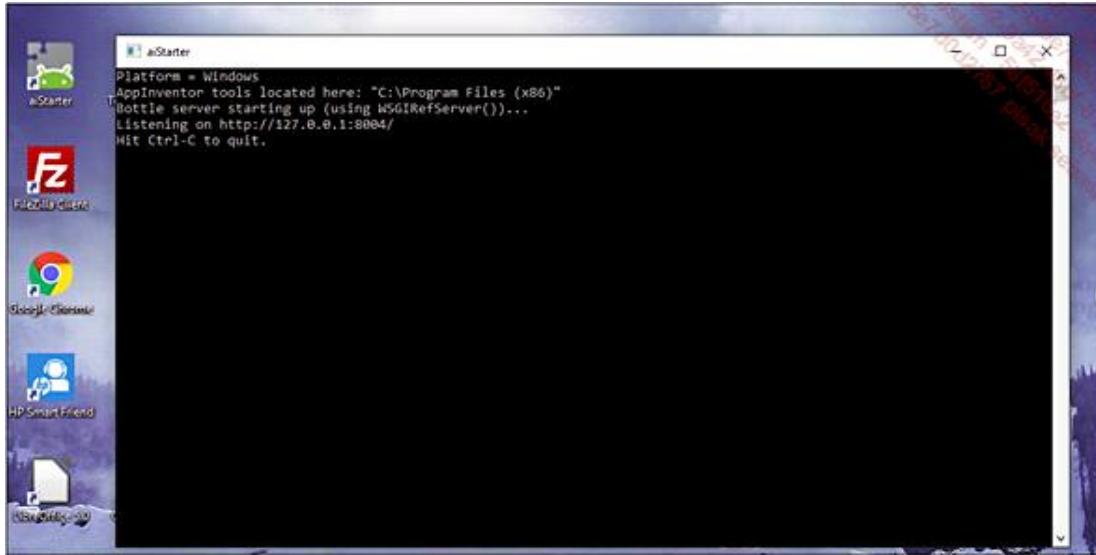
```
/usr/google/appinventor/commands-for-Appinventor/aiStarter &
```

- Rendez-vous sur <http://ai2.appinventor.mit.edu/>, sélectionnez le projet de votre choix, cliquez sur **Connect** et sélectionnez **USB**, vous devriez désormais pouvoir déboguer votre application en direct.

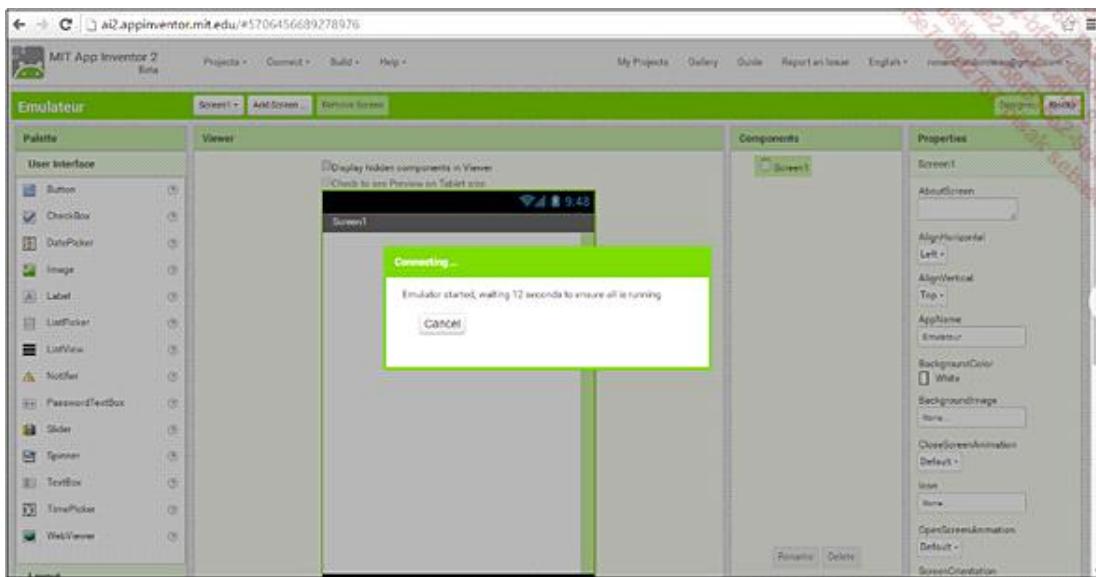
Installation de l'émulateur App Inventor 2

Si vous ne possédez pas de téléphone ou de tablette Android, vous pouvez utiliser l'émulateur. Il s'agit d'un logiciel que vous installez sur votre ordinateur que celui-ci soit sur Mac, Windows ou Linux. Naturellement, toutes les fonctions de votre téléphone émulé ne seront pas disponibles, mais cela vous permettra déjà d'avoir un aperçu.

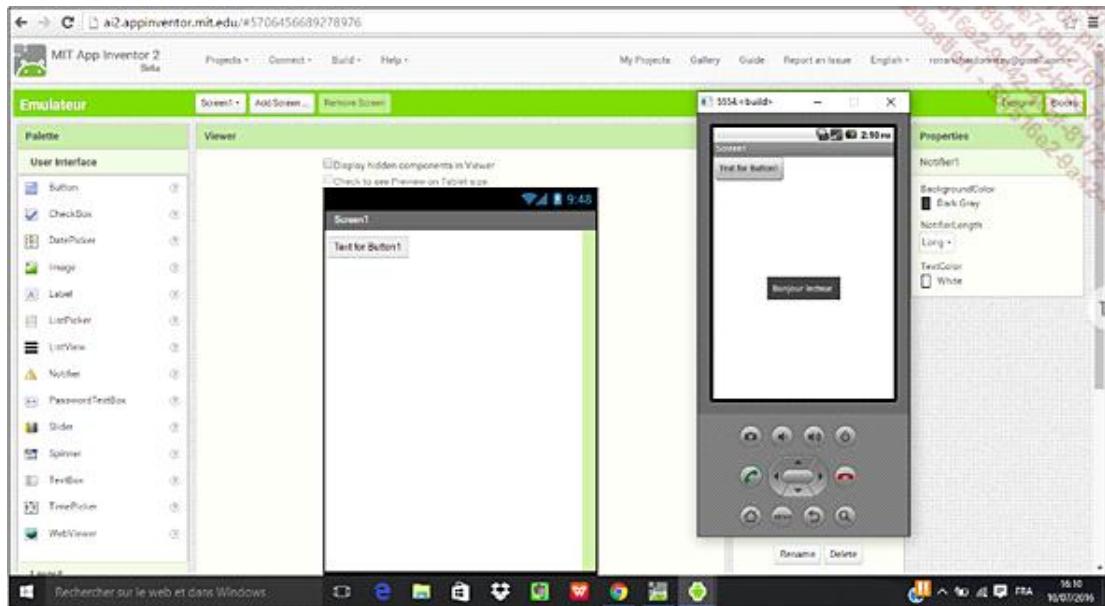
- Afin d'installer l'émulateur, vous devez installer App Inventor 2 sur votre machine comme nous l'avons vu lors de l'étape précédente.
- Double cliquez simplement sur le logo aiStarter afin de lancer l'application :



- Une fois cela fait, allez sur le site <http://appinventor.mit.edu/> afin de connecter l'émulateur.
- Dans <http://appinventor.mit.edu/>, sélectionnez l'un de vos projets ou créez-en un nouveau.
- Cliquez sur l'onglet **Connect** puis sur l'option **Emulator**, l'émulateur va alors se lancer. Soyez patient, le processus prend quelques secondes. Il est d'ailleurs possible que, durant cette étape, App Inventor vous demande de mettre à jour la version d'App Inventor 2.



Une fois l'émulateur mis en place, vous verrez apparaître une petite fenêtre vous permettant de voir votre application développée en temps réel et de pouvoir la tester. Vous allez ainsi gagner un temps non négligeable.



Installation d'App Inventor 2 en local sur votre ordinateur

La plupart des utilisateurs font fonctionner App Inventor 2 directement sur le site <http://ai2.appinventor.mit.edu/>. Cependant, cela peut vous poser quelques contraintes, par exemple : si vous avez une mauvaise connexion à Internet, si le serveur du MIT rencontre des problèmes (ce qui est déjà arrivé et vous arrivera probablement), le poids des applications est limité.

De par son aspect open source, App Inventor 2 peut s'installer directement sur votre ordinateur. La méthode présentée ci-dessous n'est pas une méthode officielle, ainsi nous vous préconisons d'utiliser la version en ligne d'App Inventor 2.

Les manipulations qui suivent peuvent vous paraître très techniques, si c'est le cas, ne vous découragez pas, il existe bien d'autres manières de pouvoir exploiter App Inventor 2.

Pour ce faire, vous devrez installer un programme du nom d'AiLiveComplete, que vous trouverez à l'adresse suivante : <https://sourceforge.net/p/ailivecomplete/wiki/Summary/>

Ce programme est le fruit du travail de monsieur Hossein Amerkashi, qui a d'ailleurs créé sa propre version payante d'App Inventor 2, AppyBuilder, disponible à l'adresse <http://appybuilder.com/>, et qui montre toutes les possibilités offertes par AI 2 au-delà même de la plateforme en ligne offerte par le MIT.

Le programme AiLiveComplete! fonctionne sur les principaux systèmes d'exploitation : Windows, Mac et Linux, il peut aussi bien être exécuté depuis votre machine que depuis une clé USB. Pour faire simple, ce programme va permettre l'installation d'un serveur sur lequel App Inventor 2 va pouvoir être exécuté et ainsi faire le rendu de l'application sur votre machine.

Le programme inclut également quelques avantages par rapport à la version en ligne, notamment la possibilité de créer des applications faisant plus de 5 MB et de personnaliser les polices de caractères des boutons et des champs de saisie de texte.

 Important : l'auteur de ce livre ne travaillant pas sur Mac, ce chapitre ne présentera les instructions d'installation que pour Windows et Linux. Mac et Linux comportant des similarités, les utilisateurs d'appareils Mac devraient parvenir à effectuer l'installation.

Pour les utilisateurs Windows

Afin d'installer le programme, voici les étapes à suivre pour les utilisateurs Windows :

- Visionnez la vidéo YouTube suivante faite par le créateur d'AiLivecomplete à l'adresse suivante : http://www.youtube.com/watch?v=n9r2bN_fZxw (vous pouvez vous en passer mais cela peut vous être utile notamment pour identifier de potentiels problèmes).
- Téléchargez le fichier à l'adresse suivante : <https://sourceforge.net/projects/ailivecomplete/files/AiLiveCompleteV1.34.zip/download> (taille du fichier à la date de l'écriture de l'ouvrage : 164 MB) ou rendez-vous sur la page <https://sourceforge.net/projects/ailivecomplete/files/> pour télécharger la dernière version (que ce soit sous Windows, Mac ou Linux, il s'agit du même fichier).
- Décomprimez ce fichier dans le répertoire de votre choix.
- La prochaine étape (qui peut être une étape délicate en fonction de la machine que vous utilisez) consiste à installer JDK, que vous trouverez à l'adresse suivante : <http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html> à un endroit spécifique de votre ordinateur, à savoir un chemin du type C:\Program Files\Java\jdk1.6.0_06 (en cas de problème, référez-vous à la documentation du développeur : <http://www.wikihow.com/Set-Java-Home>). Vous verrez que,

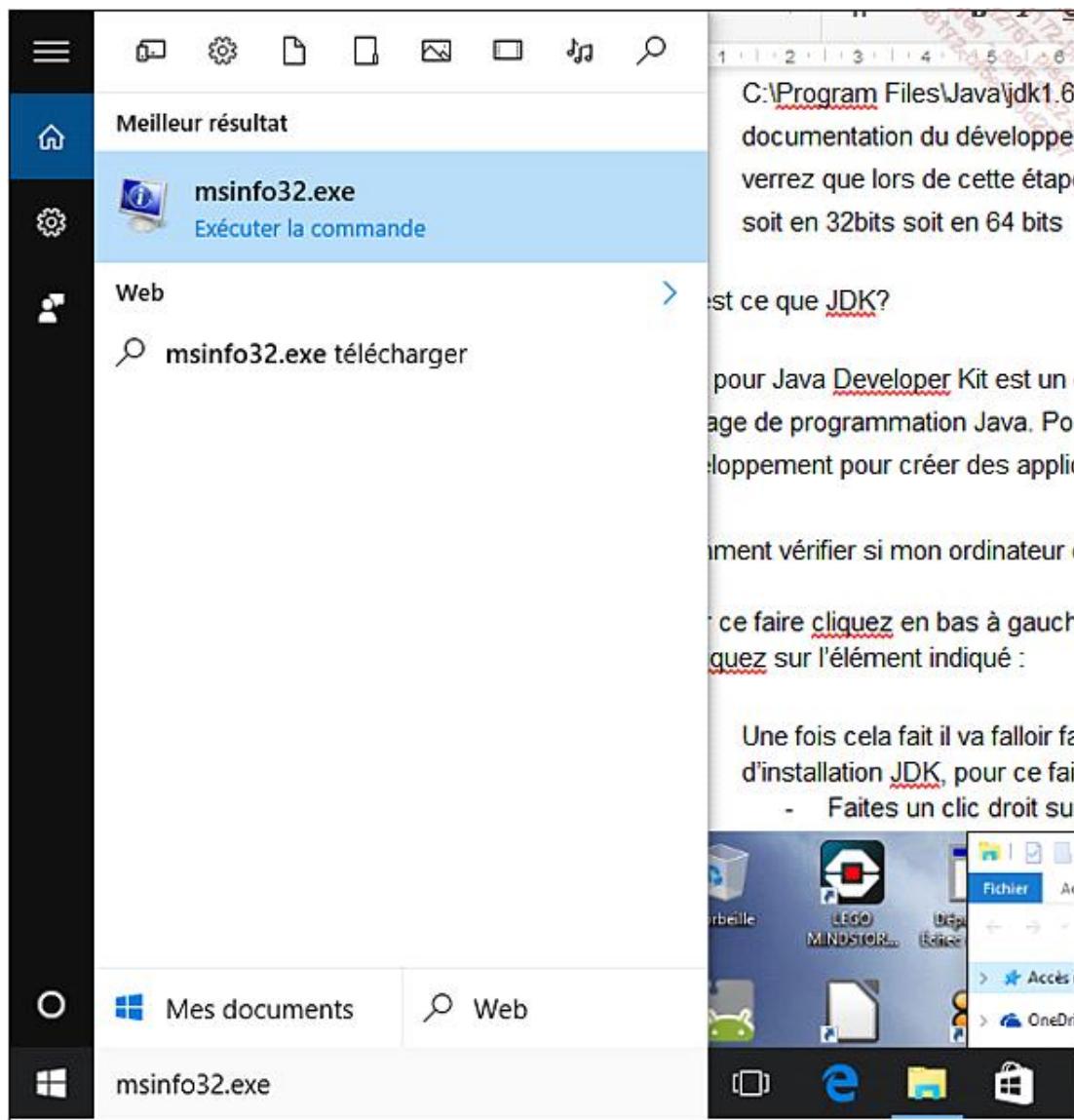
lors de cette étape, on vous proposera différents types de téléchargement, soit en 32 bits, soit en 64 bits.

Qu'est-ce que JDK ?

JDK pour Java Developer Kit est un ensemble de bibliothèques logicielles de base du langage de programmation Java. Pour faire simple, il s'agit d'un environnement de développement pour créer des applications pour mobile en utilisant le langage Java.

Comment vérifier si mon ordinateur est en 32 bits ou 64 bits ?

- Pour ce faire, cliquez en bas à gauche de votre écran et entrez le mot suivant : **msinfo32.exe**. Cliquez ensuite sur l'élément indiqué :



Dans la rubrique **Type d'Informations système**, vous trouverez l'information recherchée, à savoir ci-dessous une version 64 bits :

Résumé système	Élément	Valeur
(i) Ressources matérielles	Nom du système d'exploitation	Microsoft Windows 10 Famille
(i) Composants	Version	10.0.10586 Numéro 10586
(i) Environnement logiciel	Autre description du système d... Non disponible	
	Fabricant du système d'exploit...	Microsoft Corporation
	Ordinateur	DESKTOP-QEDTQQM
	Fabricant	HP
	Modèle	HP Notebook
	Type	PC à base de x64
	Référence (SKU) du système	PSP9TEA#ABF
	Processeur	Intel(R) Core(TM) i5-4210U CPU @ 1.70GHz, 2401 MHz, 2 cœur(s), 4 processeur(s)
	Version du BIOS/Date	Insyde F.1F, 18/01/2016
	Version SMBIOS	2.8
	Version du contrôleur embarqué	96.45
	Mode BIOS	UEFI
	Fabricant de la carte de base	HP

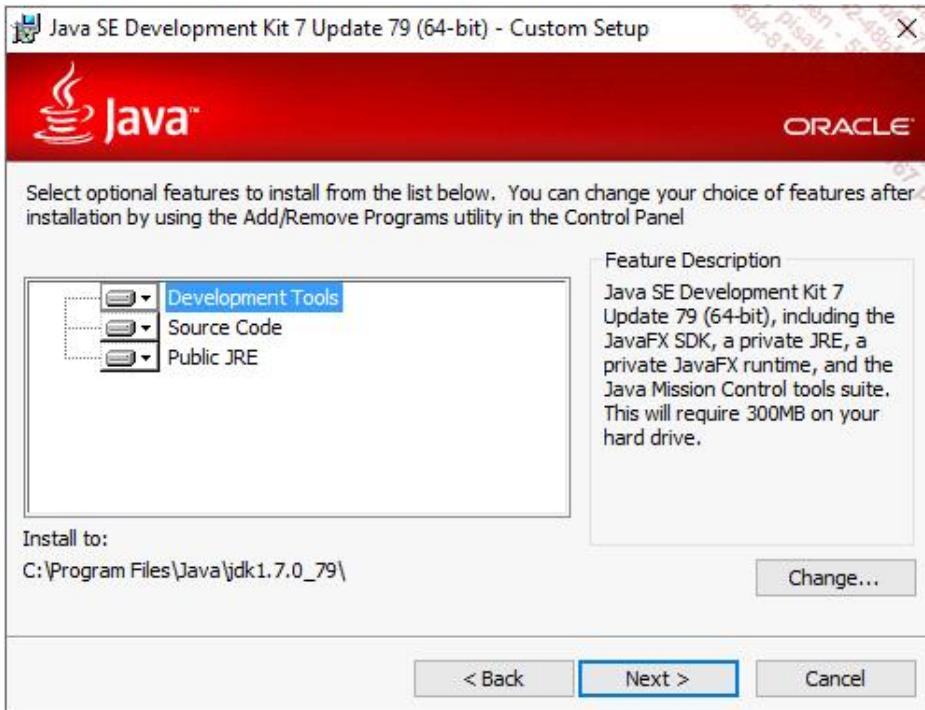
→ Dans notre cas de figure, il faudra donc télécharger la version 64 bits pour Windows :

Looking for JDK on ARM?
JDK 7 for ARM downloads have moved to the [JDK 7 for ARM download page](#).

Java SE Development Kit 7u79
You must accept the [Oracle Binary Code License Agreement for Java SE](#) to download this software.
Thank you for accepting the Oracle Binary Code License Agreement for Java SE; you may now download this software.

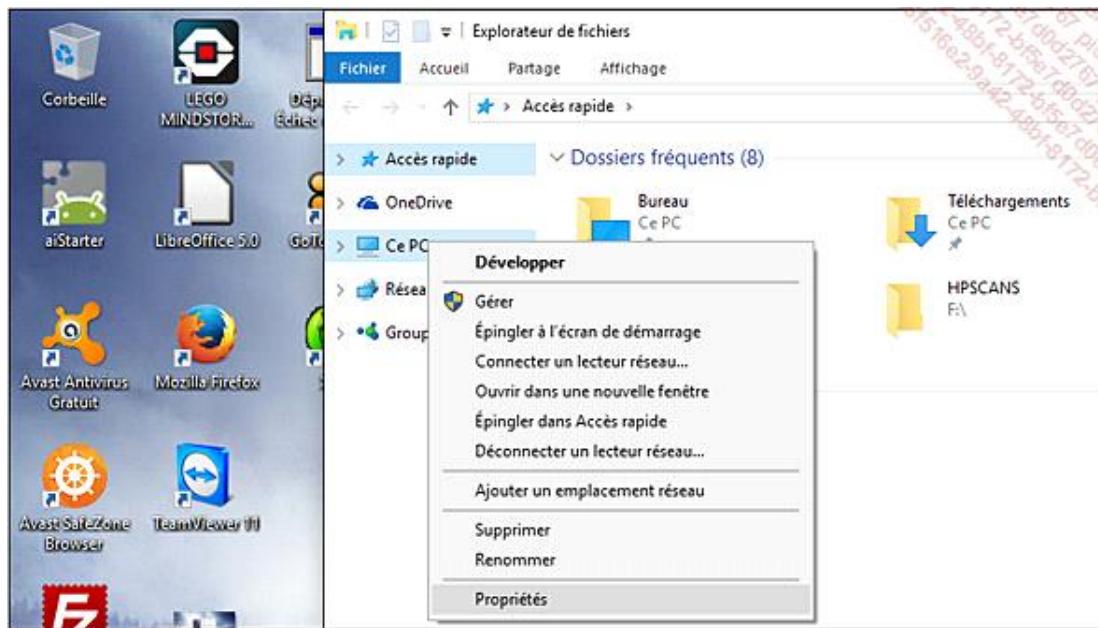
Product / File Description	File Size	Download
Linux x86	130.4 MB	jdk-7u79-linux-i586.rpm
Linux x86	147.6 MB	jdk-7u79-linux-i586.tar.gz
Linux x64	131.69 MB	jdk-7u79-linux-x64.rpm
Linux x64	146.4 MB	jdk-7u79-linux-x64.tar.gz
Mac OS X x64	196.89 MB	jdk-7u79-macosx-x64.dmg
Solaris x86 (SVR4 package)	140.79 MB	jdk-7u79-solaris-i586.tar.Z
Solaris x86	96.66 MB	jdk-7u79-solaris-i586.tar.gz
Solaris x64 (SVR4 package)	24.67 MB	jdk-7u79-solaris-x64.tar.Z
Solaris x64	16.38 MB	jdk-7u79-solaris-x64.tar.gz
Solaris SPARC (SVR4 package)	140 MB	jdk-7u79-solaris-sparc.tar.Z
Solaris SPARC	99.4 MB	jdk-7u79-solaris-sparc.tar.gz
Solaris SPARC 64-bit (SVR4 package)	24 MB	jdk-7u79-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	18.4 MB	jdk-7u79-solaris-sparcv9.tar.gz
Windows x86	138.31 MB	jdk-7u79-windows-i586.exe
Windows x64	140.06 MB	jdk-7u79-windows-x64.exe

Et, lors de l'installation, bien indiquer le chemin souhaité :

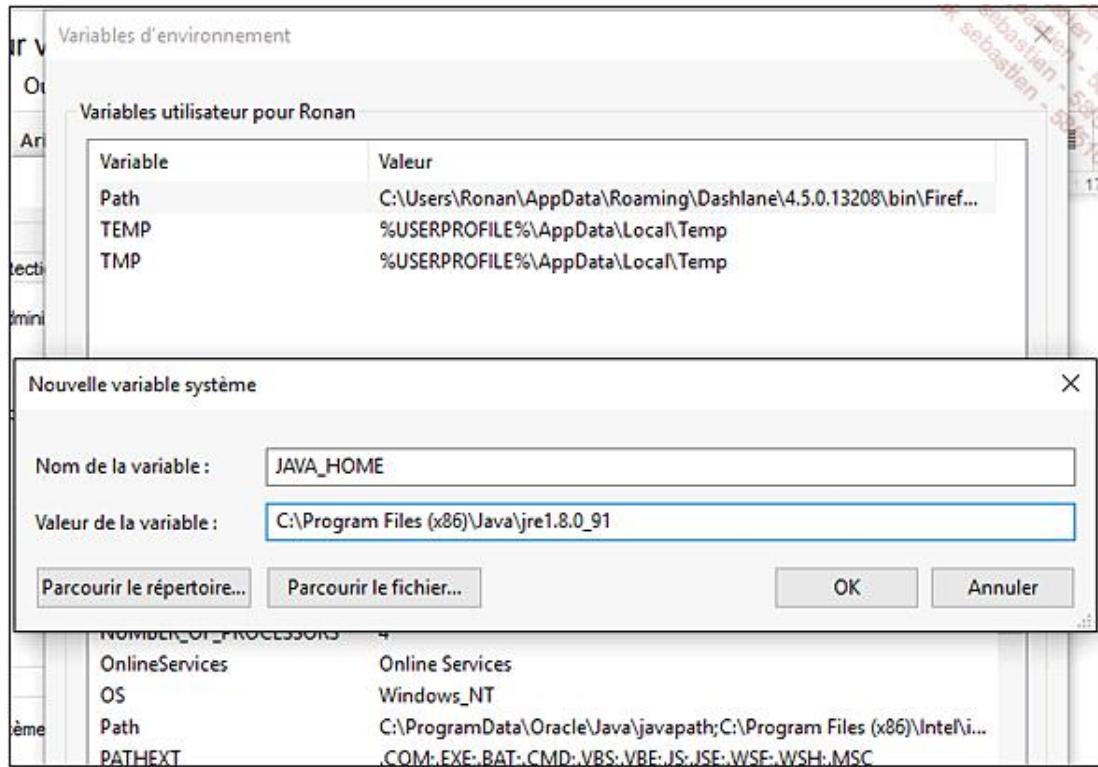


Une fois le jdk installé, il faut faire pointer la variable JAVA_HOME vers le répertoire d'installation JDK. Pour ce faire :

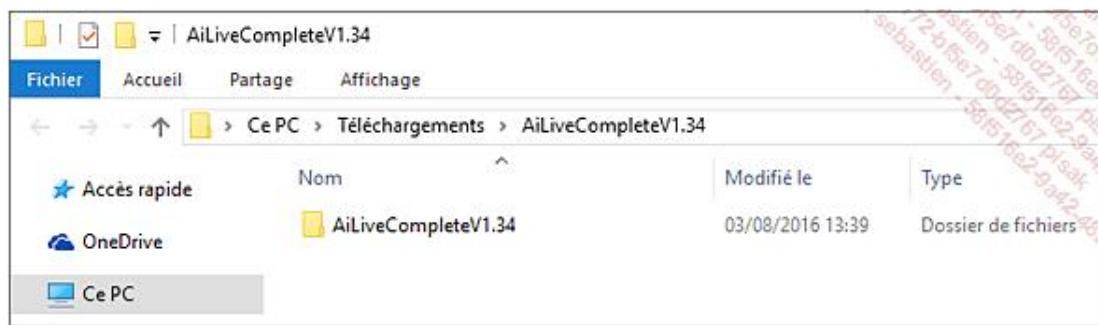
- Faites un clic droit sur **Mon PC** et sélectionnez **Propriétés** :



- Dans **Propriétés**, cliquez sur **Modifier les paramètres** puis **Paramètres système avancés** puis **Variables d'environnement**.
- En dessous du menu **Variables système**, cliquez sur **Nouvelle**.
- Entrez le nom de la variable **JAVA_HOME**.
- Entrez pour valeur le chemin d'installation du fichier JDK, soit un chemin similaire à **C:\Program Files (x86)\Java\jre1.8.0_91**.



- Cliquez sur **OK** puis sur **Appliquer les changements**.
- Redémarrez l'ordinateur afin d'être sûr que les changements ont bien été pris en considération en consultant à nouveau les variables d'environnement.
- Une fois cette action effectuée, accédez à l'endroit où vous avez décompressé les fichiers d'AiLiveComplete :



- Double cliquez sur **AiLiveCompleteV1.34** puis sur **WinStartAIServer** ainsi que sur **WinStartBuildServer** afin de lancer les serveurs locaux (ne vous étonnez pas si deux fenêtres noires se chargent avec des instructions, il s'agit de terminaux Windows, ne les fermez pas).
- Démarrez un navigateur tel que Google Chrome ou Firefox et rendez-vous à l'adresse : http://localhost:8888



À cette étape, vous devriez avoir une version d'App Inventor installée sur votre machine :



À noter qu'il est possible qu'un message vous demande de mettre à jour Java pour pouvoir exploiter intégralement le potentiel d'App Inventor hors ligne ; dans ce cas, suivez simplement les instructions suggérées par l'ordinateur.

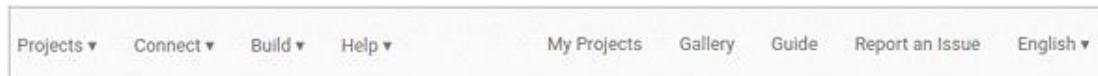
Pour les utilisateurs Linux

- Installez JDK : <http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html>. Pour ce faire, téléchargez le fichier relatif à votre système d'exploitation. Une fois cela fait, dézippez-le avec une requête en ligne de commande de type : tar zxvf jdk-7u<version>-linux-x64.tar.gz suivez le tutoriel de Korben : <http://korben.info/installer-java-7-ubuntu.html>
- Téléchargez et dézipez le fichier : <https://sourceforge.net/projects/ailivecomplete/files/latest/download>
- Ouvrez un terminal et rendez-vous à l'aide de la commande cd dans le répertoire dans lequel a été dézippé le fichier.
- Modifiez les fichiers .sh afin qu'ils soient exécutables en utilisant la commande chmod +x *.sh.
- Démarrez le serveur App Inventor en entrant la commande suivante : ./startAIServer.sh
- Ouvrez un autre terminal, et rendez-vous de la même manière via la commande dans le répertoire dans lequel vous avez dézippé les fichiers d'AILiveComplete!.
- Utilisez la commande ./startBuildServer.sh afin de lancer le serveur d'App Inventor.
- Ouvrez un navigateur et entrez l'adresse <http://localhost:8888>.

Vous savez désormais comment installer App Inventor 2 en local sur votre ordinateur.

Introduction

Lorsque vous vous connectez à App Inventor 2, vous remarquez que celui-ci est composé d'un menu supérieur comportant les éléments suivants :



Nous allons voir en détail dans ce chapitre, les différentes fonctionnalités que comporte chacun de ces menus.

Les menus

1. Projects

Le menu **Projects** est composé des éléments suivants :

- **My Projects**
- **Start new project**
- **Import project (.aia) from my computer**
- **Import project (.aia) from a repository**
- **Delete Project**
- **Save project**
- **Save project as ...**
- **Checkpoint**
- **Export selected project (.aia) to my computer**
- **Export all projects**
- **Import keystore**
- **Export keystore**
- **Delete keystore**

My Projects

Liste l'ensemble des applications que vous avez créées ou êtes en train de créer :

Name	Date Created	Date Modified	Published
Porte	Aug 6, 2016, 8:03:52 PM	Aug 7, 2016, 3:57:06 PM	No
FileComponent	Aug 5, 2016, 5:25:35 PM	Aug 4, 2016, 8:03:49 PM	No
LegоМindstorms	Jul 30, 2016, 7:59:54 PM	Aug 5, 2016, 5:13:22 PM	No
BrickAppEV3	Aug 2, 2016, 12:44:47 AM	Aug 3, 2016, 9:49:19 PM	No
BaseEV3ProjectKV	Aug 2, 2016, 10:40:39 AM	Aug 3, 2016, 9:46:15 PM	No
ListSample	Aug 3, 2016, 9:15:49 PM	Aug 3, 2016, 9:16:54 PM	No
MindstormControl	Aug 2, 2016, 10:23:54 AM	Aug 3, 2016, 8:28:40 PM	No

Pour chaque projet, vous verrez le nom s'afficher, la date à laquelle il a été créé, la date de dernière modification et l'état de publication (c'est-à-dire si votre application a été publiée dans la galerie communautaire d'App Inventor 2).

Start new project

Comme son nom l'indique, cette fonctionnalité permet de créer une nouvelle application.

Import project (.aia) from my computer

Les fichiers .aia sont propres à App Inventor 2, il s'agit ni plus ni moins que de fichiers sources pour AI 2, c'est-à-dire qu'ils vous permettent d'accéder à l'intégralité du code. La fonction **Import project (.aia) from my computer** vous permet d'importer un projet .aia et donc une application App Inventor 2 que vous avez téléchargée et qui se situe sur votre ordinateur. Il peut s'agir aussi bien d'une de vos applications que de personnes tierces dont vous avez téléchargé le fichier sur Internet.

Import project (.aia) from a repository

Vous avez la possibilité d'importer une application AI 2 depuis votre ordinateur, c'est-à-dire avec un téléchargement préalable, ou vous pouvez directement utiliser Internet en allant automatiquement chercher le fichier dans un dépôt, c'est justement ce que permet de faire cette fonctionnalité.

Delete Project

Comme son nom l'indique, cette fonctionnalité permet de supprimer un projet créé dans App Inventor 2. À noter que si vous utilisez la version en ligne d'App Inventor 2, vous utilisez les serveurs du MIT, qui vous sont prêtés gracieusement, c'est donc une bonne pratique de nettoyer de temps en temps votre interface d'AI 2.

Save project

Cette fonctionnalité vous permet de forcer la sauvegarde de votre projet en cours. Bien qu'App Inventor 2 sauvegarde automatiquement votre projet au cours de sa réalisation, cette fonction sert surtout pour les personnes très conscientes.

Save project as

Contrairement à la fonctionnalité précédente, sauvegarder votre projet "en tant que" va vous permettre de créer un point de sauvegarde sur votre ordinateur en vous créant une nouvelle application. App Inventor 2 écrasant systématiquement votre projet au cours de son développement, il s'agit d'une fonctionnalité très utile notamment pour de gros projets.

Checkpoint

Cette fonctionnalité est similaire à la précédente. App Inventor 2 ne vous permettant pas d'annuler les modifications que vous allez effectuer sur votre application, les checkpoints vont vous permettre de créer des versions de votre application à des moments que vous définissez. Ainsi, si vous commettez une erreur vous pourrez revenir à un état précédent.

Export selected project (.aia) to my computer

Cette fonctionnalité vous permet de récupérer le fichier .aia composant votre application afin de pouvoir le transférer vers un autre compte, à l'un de vos proches, voir même pour l'ouvrir depuis des éditeurs de texte et en voir le contenu.

Export all projects

Cette fonctionnalité vous permet d'exporter tous vos projets d'applications en un seul fichier, cela peut être utile si vous souhaitez par exemple clôturer votre compte AI 2 tout en gardant une trace de votre travail.

Import keystore

Le mot "keystore" fait référence à une clé privée qui est associée à votre compte App Inventor 2. Lorsque vous créez une application via AI 2 cette fameuse clé est utilisée et marque votre application. Si jamais une nouvelle version de cette application voit le jour, alors la même clé doit être utilisée. Les serveurs du MIT gèrent pour vous ces fameuses clés et ne devraient en principe pas les perdre. Cependant, si jamais cela arrive et que vous avez au préalable exporté cette fameuse clé, vous pourriez à nouveau l'importer grâce à la fonctionnalité **Import keystore**.

Export keystore

Cette clé privée est enregistrée dans ce que l'on appelle un fichier keystore qu'il vous est recommandé d'exporter, de garder à l'abri et de ne pas partager.

Delete keystore

Permet de supprimer la clé privée, chose qui n'est pas recommandée. À savoir qu'en cas de suppression, les serveurs du MIT ne seront pas en mesure de retrouver ce fichier. Pour en savoir plus sur les keystore : <http://appinventor.mit.edu/explore/ai2/google-play.html>.

2. Connect

Le menu **Connect** comporte les fonctionnalités suivantes :

- **AI Companion**
- **Emulator**
- **USB**
- **Reset Connection**
- **Hard Reset**

AI Companion

Cette fonctionnalité vous permet de vous connecter directement à l'application que vous êtes en train de créer sur le serveur d'AI 2 sur votre mobile.

Emulator

Cette fonctionnalité va activer un émulateur au sein de votre ordinateur afin de pouvoir directement tester l'application sur votre ordinateur. C'est notamment cette option que vous pouvez utiliser si vous n'avez pas de smartphone à portée de main. À noter que les fonctionnalités de l'émulateur sont limitées.

USB

Cette fonctionnalité va relier AI 2 depuis votre ordinateur à votre appareil mobile via un câble USB et vous permettre de voir en temps réel le développement de votre application.

Reset Connection

Il est possible que votre émulateur, votre connexion USB rencontrent des problèmes lors du développement de votre application. Comme l'indique directement le site d'AI 2, il s'agit d'une version bêta. **Reset Connection** va vous permettre de réactiver la connexion entre le serveur d'AI 2 et votre machine.

Hard Reset

Comme son nom l'indique, il s'agit de réinitialiser la connexion effectuée entre le serveur AI 2 et votre mobile.

3. Build

Ce menu comporte deux fonctionnalités :

- **App (provide QR code for .apk)**
- **App (save .apk to my computer)**

App (provide QR code for .apk)

Ce lien va générer un QR code que vous pourrez flasher à l'aide de l'application MIT AI 2 Companion afin d'installer directement l'application sur votre mobile et ainsi de pouvoir la tester.

App (save .apk to my computer)

Vous permet de télécharger le fichier .apk (.apk permet l'exécution de l'application en tant que telle, c'est ce type de fichier que vous téléchargez lorsque vous installez une application depuis le Play Store de Google), pour un transfert via un autre appareil afin d'installer l'application. Vous pouvez par exemple utiliser cette solution si vous n'avez pas installé l'application MIT AI 2 Companion sur votre mobile.

4. Help

Le menu **Help** comporte quant à lui les éléments suivants :

- **About**
- **Library**
- **Extensions**
- **Tutorials**
- **Troubleshooting**
- **Forums**
- **Report an Issue**
- **Companion Information**
- **Update the Companion**
- **Show Splash Screen**

About

Vous indique les informations de base concernant la version d'App Inventor 2 que vous utilisez. À noter que vous avez la possibilité de consulter les différentes évolutions qu'AI 2 a connues au cours des différentes versions à l'adresse suivante : <http://appinventor.mit.edu/ai2/ReleaseNotes.html>



Library

Ce lien vous redirige vers la documentation officielle d'App Inventor 2 : <http://appinventor.mit.edu/explore/library> en anglais.

Extensions

Ce lien vous dirigera vers la page officielle des extensions supportées par App Inventor 2 : <http://appinventor.mit.edu/extensions/>. Les extensions permettent de rajouter des fonctionnalités non comprises dans la version en ligne d'AI 2 (par exemple un composant détecteur de son, un composant Bluetooth moins gourmand en termes de ressources). Malheureusement, à la date de rédaction de cet ouvrage, peu d'extensions ont été développées par la communauté.

Tutorials

Ce lien redirige directement vers les tutoriels originaux d'App Inventor 2 : <http://appinventor.mit.edu/explore/ai2/tutorials>. Nous vous recommandons fortement de suivre les premiers tutoriels de niveau "Basic", ces derniers vous permettent de prendre très rapidement confiance en vous et de faire avancer vos projets pas à pas.

Troubleshooting

Ce lien vous renverra vers toutes les questions fréquentes en termes de problèmes que vous pourriez rencontrer lors de votre utilisation d'App Inventor 2 : <http://appinventor.mit.edu/explore/ai2/support/troubleshooting>. Vous serez probablement obligés de vous y référer lors du développement de vos applications, de nombreuses problématiques peuvent être rencontrées, notamment sur la partie connexion entre le serveur du MIT et votre mobile mais également dans l'utilisation des composants propres à votre application. En général ce sont surtout les composants de connexion qui vont vous donner du fil à retordre. Nous vous conseillons également le très bon site Internet : <http://puravidaapps.com/> qui se trouve être un excellent complément de la documentation officielle. À noter que les problèmes que vous rencontrerez se comptent en général sur les doigts d'une main.

Forums

Celui-ci redirige vers le groupe de discussion Google <https://groups.google.com/forum/#!forum/mitappinventortest> propre à App Inventor.

Report an Issue

À utiliser si vous souhaitez faire remonter un problème à la communauté App Inventor ; à noter qu'à la date de rédaction de l'ouvrage, celui-ci redirige vers le même lien que celui du groupe de discussion.

Companion Information

Un simple lien vous permettant de télécharger directement l'application App Inventor 2 pour votre mobile.

Update the Companion

Ce lien vous permet de mettre à jour directement sur votre mobile l'application App Inventor 2.

Show Splash Screen

Le Splash Screen fait référence à la pop-up d'affichage lorsque vous vous connectez à App Inventor 2. Bien que semblant inutile au premier abord, cette dernière a surtout pour objectif de vous informer de la dernière mise à jour du logiciel et surtout des fonctionnalités qui viennent d'être ajoutées ainsi que de potentiels bugs corrigés :

Welcome to App Inventor!

Welcome to MIT App Inventor

July 27, 2016: Release (nb150b) is out.

[More Information](#)

We have fixed the problem with event blocks in non-English languages.
You may have to reload your browser to see the effect. Help->About
should say you are running version nb150b

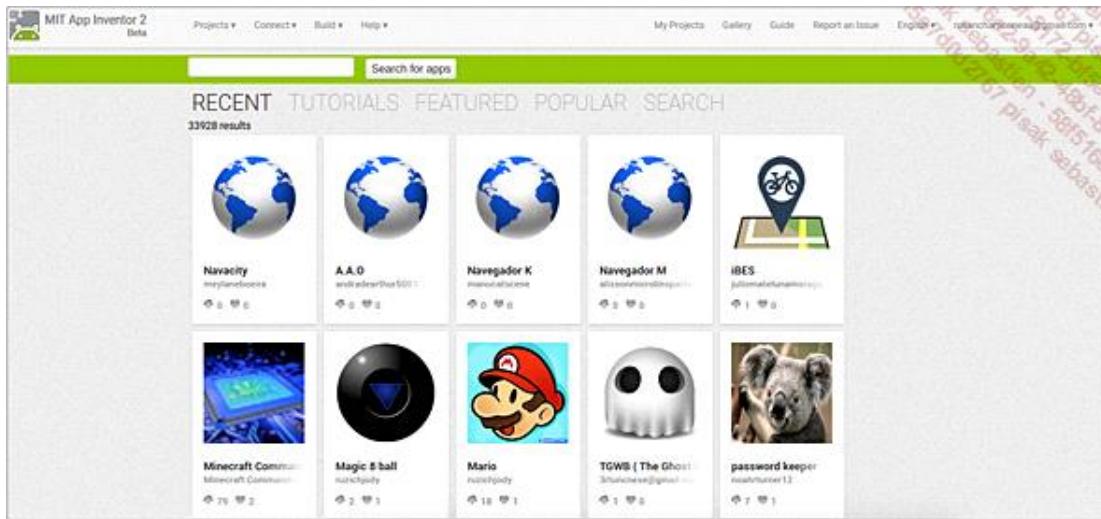
Got an Android phone or tablet? Find out how to
[Set up and connect an Android device.](#)

Don't have an Android device? Find out how to
[Set up and run the Android emulator.](#)

[Continue](#) Do Not Show Again

5. Gallery

Celle-ci regroupe l'ensemble des applications créées et publiées par d'autres utilisateurs :



Ces dernières vous sont notamment très utiles pour gagner un temps considérable lors du développement de vos applications. En effet, de par son aspect open source, AI 2 vous permet de télécharger ces applications, de pouvoir les ouvrir dans App Inventor 2, d'en récupérer la structure et ainsi de compléter ou d'améliorer vos applications existantes :



Il vous suffit de cliquer sur le bouton **OPEN THE APP** pour que celle-ci s'installe au sein de votre interface d'App Inventor 2, vous n'aurez plus qu'à copier-coller les blocs logiques vous intéressant pour votre propre application.

Identifiez la galerie comme un lieu test afin de voir si vos applications pourraient avoir un succès en cas de publication sur le Play Store. Après tout, si votre application plaît à la communauté App Inventor 2, elle devrait plaire à un plus large public.

6. Guide

Ce lien redirige vers les guides d'utilisation officiels d'App Inventor 2 précédemment cités.

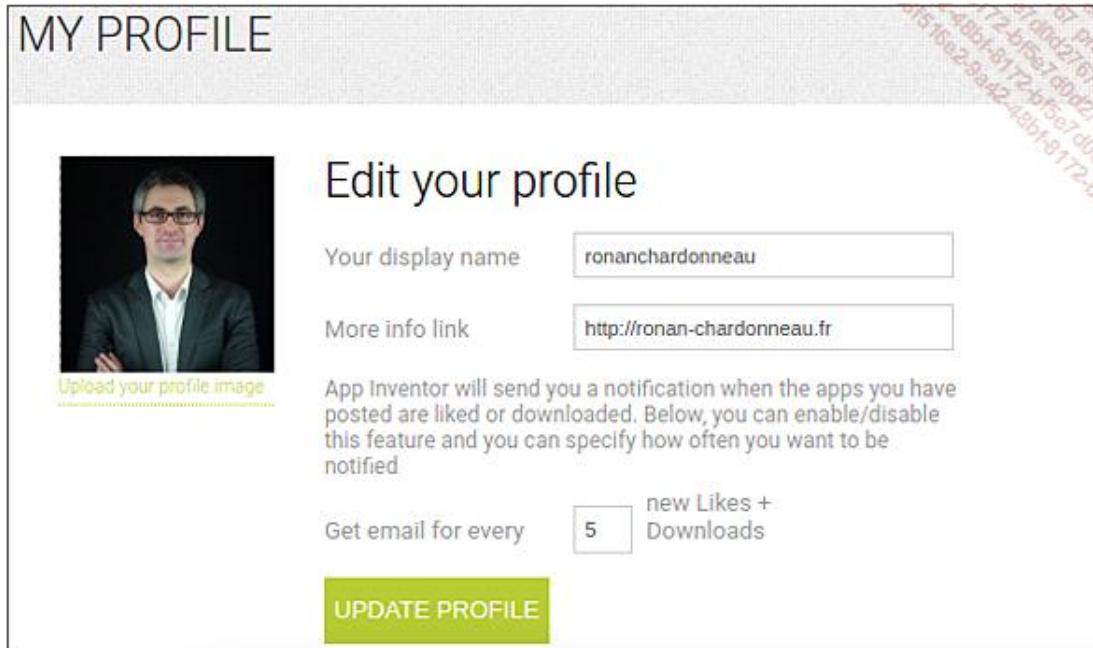
7. Languages

À la date d'écriture de cet ouvrage, App Inventor 2 propose sa plateforme dans près de 12 langues. Cependant certaines traductions étant très approximatives, nous vous recommandons fortement de rester sur la version anglaise de ce dernier. C'est par ailleurs une bonne pratique car la plupart des tutoriels avancés que vous trouverez sur des plateformes telles que YouTube sont en anglais.

Profil

Lorsque vous vous connectez à App Inventor 2, votre adresse e-mail apparaît en haut à droite de l'écran. Lorsque vous cliquez sur celle-ci, un menu déroulant s'affiche vous offrant la possibilité de vous connecter ou d'accéder à votre profil.

Le profil vous permet de vous présenter brièvement auprès de la communauté lorsque votre application est publiée dans la galerie d'App Inventor 2 :



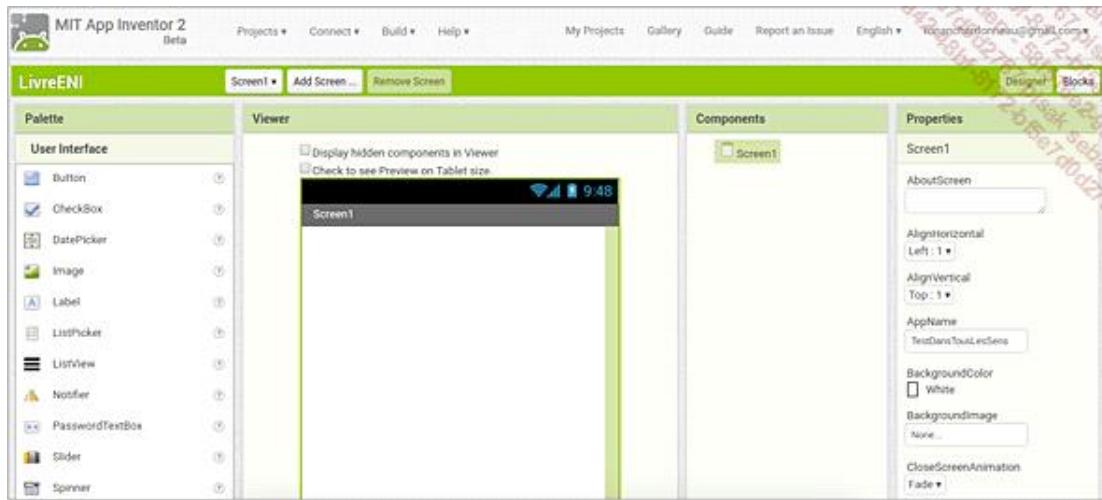
Vous pouvez notamment y indiquer le nom que vous voulez voir afficher, votre site internet, votre photo, et paramétriser le nombre de notifications que vous souhaitez recevoir lorsque des utilisateurs téléchargent ou aiment votre application.

C'est la fin de chapitre, vous connaissez désormais les différentes opportunités offertes par l'espace d'administration d'App Inventor 2, découvrons maintenant les différents composants de la plateforme.

Introduction

Maintenant que vous connaissez les principales fonctions de l'administration d'App Inventor 2 nous allons voir les différents composants mis à notre disposition par la plateforme.

App Inventor 2 se présente de la manière suivante :



Dans la colonne de gauche intitulée **Palette** se trouve l'ensemble des composants que vous pourrez inclure à votre application. C'est-à-dire les différentes fonctionnalités : un formulaire, une image, la fonctionnalité Bluetooth...

Afin d'inclure ces composants, il vous suffit simplement de les faire glisser depuis la palette jusqu'à l'écran du mobile symbolisé ci-dessus par le rectangle visible dans la zone **Viewer**.

La troisième colonne intitulée **Components** pour "Composants" en français, sert à sélectionner les composants que vous avez cliqués-déposés depuis la palette.

Chacun de ces composants que vous ajoutez à votre application possède ce que l'on appelle des propriétés, c'est-à-dire des caractéristiques qui lui sont propres et qui sont regroupées dans la quatrième colonne nommée **Properties**.

Voici quelques exemples de propriétés pour un composant bouton : la taille du bouton, le texte écrit sur le bouton, la forme du bouton, sa couleur...

Toutes ces colonnes, **Palette**, **Viewer**, **Components**, **Properties** se situent dans l'espace **Designer** dont le nom apparaît en haut à droite de l'écran. Il existe deux espaces quand vous créez une application, l'espace **Designer**, où vous indiquez graphiquement ce à quoi votre application va ressembler, et l'espace **Blocks**, que nous verrons plus tard dans cet ouvrage et qui représente le comportement que va avoir votre application en fonction du comportement de l'utilisateur.

Voyons maintenant plus en détail les différents composants que vous pouvez ajouter dans App Inventor 2.

Dans ce livre, nous avons fait le choix de vous présenter l'interface en anglais d'App Inventor en vous indiquant entre parenthèses la traduction française des principaux éléments.

Découvrez ci-après la liste des composants que vous pouvez utiliser pour créer vos applications Android. Ce chapitre n'aborde pas comment les utiliser, ce sera en revanche le cas des prochains chapitres.

Lorsque vous créez votre application sous App Inventor 2, vous devez sélectionner les composants qui vont être inclus dans cette dernière. Cela peut aller d'un simple bouton à des composants complexes tels qu'un podomètre, une

caméra vidéo. Ces fonctionnalités dépendront naturellement des capacités de votre smartphone ou tablette.

Il va de soi que si votre téléphone ne comporte pas de composant Bluetooth, votre application ne pourra pas utiliser cette fonctionnalité sur votre mobile.

Comme indiqué précédemment, la palette regroupe l'ensemble des composants supportés par App Inventor 2.

Palette
User Interface
Layout
Media
Drawing and Animation
Sensors
Social
Storage
Connectivity
LEGO® MINDSTORMS®
Experimental
Extension

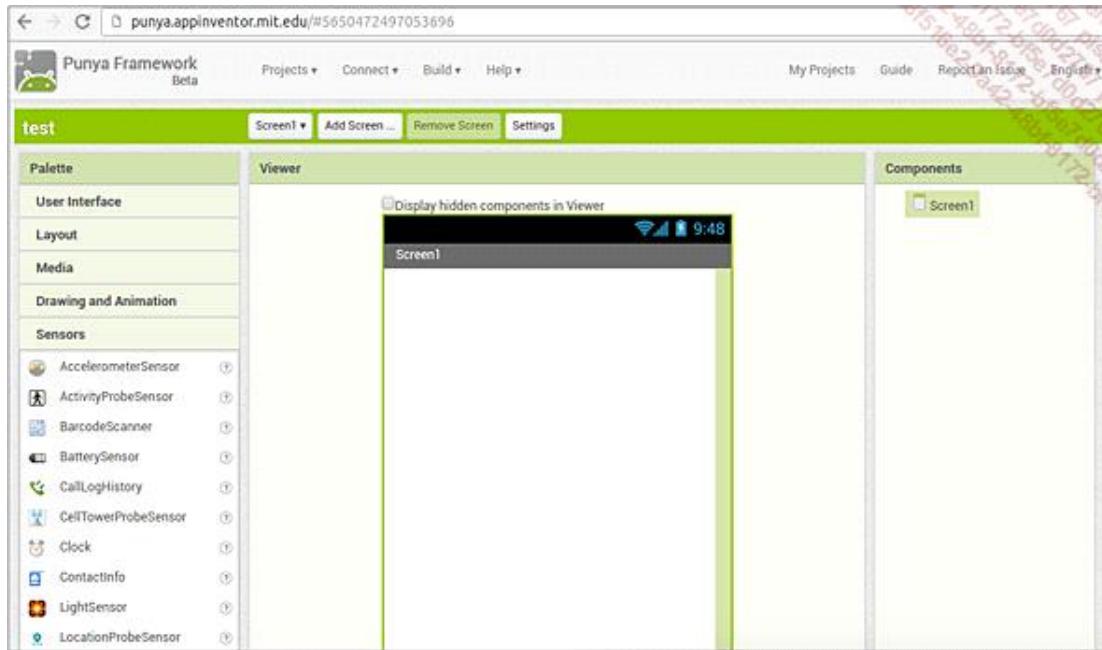
Ils se divisent en plusieurs catégories regroupant les composants relatifs :

- à l'interface utilisateur (bouton, case à cocher, image),
- à la disposition des blocs (c'est-à-dire l'affichage, le design, de votre application),
- aux médias (tout ce qui est relatif aux médias traditionnels : sons, vidéos, appareil photo),
- au dessin et à l'animation (tout ce qui va afficher des animations à l'écran tel que des GIF animés, des mouvements d'images),
- aux capteurs (les composants techniques propres à votre smartphone tels que l'orientation de votre téléphone, le temps qui passe...),
- aux interactions sociales (tout ce qui concerne la communication, comme les réseaux sociaux, les appels téléphoniques...),
- au stockage (tout ce qui est lié à la base de données),
- à la connectivité (communication via les réseaux tels que le Bluetooth, le Web),
- à LEGO MINDSTORMS (robotique liée à des produits spécifiques de la marque LEGO),
- aux fonctionnalités expérimentales (nouvelles fonctionnalités testées par AI 2, à la date de l'écriture de cet ouvrage, il s'agit de Firebase),
- aux extensions (cela concerne tous les développements spécifiques à AI 2 réalisés par des tiers).

Il est possible que votre téléphone possède des composants autres que ceux présents dans AI 2. Dans ce cas de

figure, vous ne pourrez malheureusement pas les utiliser tant que vous n'avez pas créé d'extensions. Par exemple, sur certains téléphones, il existe des composants permettant d'émettre de l'infrarouge et ainsi de contrôler des appareils tels que des télévisions.

À noter également qu'il existe des "forks" d'App Inventor 2, c'est-à-dire d'autres plateformes basées sur la technologie initiale d'AI 2 qui peuvent vous proposer nativement d'autres fonctionnalités, plus élaborées par exemple. C'est le cas de Punya, qui vous propose la possibilité d'utiliser la lumière "flash" de votre téléphone, le journal d'appels de votre téléphone, le niveau de batterie... Pour en savoir plus sur le projet Punya : <http://punya.mit.edu/project>

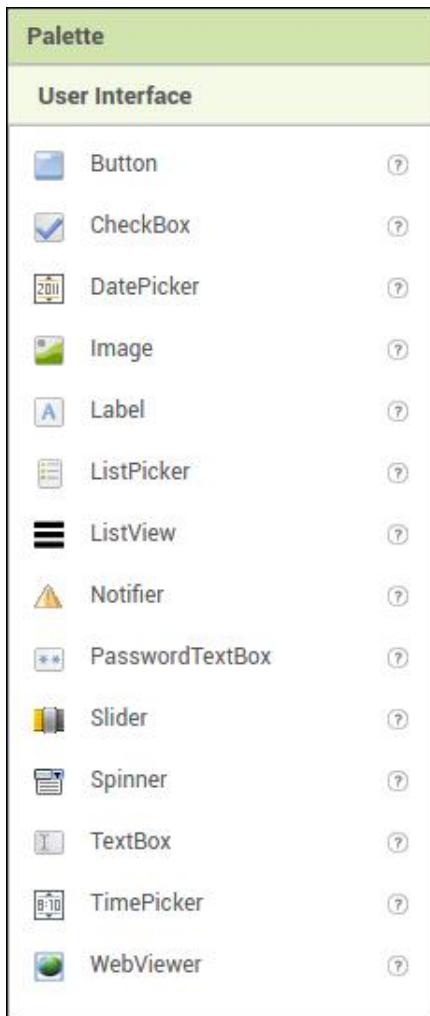


App Inventor 2 reste cependant à l'heure de l'écriture de cet ouvrage la version la plus utilisée.

Il existe également des distributions payantes, c'est le cas de <http://appybuilder.com/> avec là aussi des composants en plus tels que le capteur de luminosité, des fonctionnalités de monétisation, e-commerce...

User Interface (Interface utilisateur)

L'interface utilisateur est le premier ensemble de composants visible dans la colonne de gauche lorsque vous êtes dans l'espace **Designer** de votre application ; il est composé des éléments suivants :



Pour découvrir ce qui se cache derrière chacun de ces composants, vous pouvez toujours cliquer sur le point d'interrogation situé à droite de celui-ci. Cependant, nous vous conseillons fortement d'utiliser la version anglaise afin d'avoir une explication de qualité.

Nous détaillons ci-après chacun de ces éléments et présentons leur utilité.

Button (Bouton)

Le bouton est l'un des éléments les plus utilisés dans la réalisation des applications. Celui-ci réagit aux clics effectués sur ce dernier et peut ainsi déclencher des actions. Par exemple, lorsque l'utilisateur clique sur un bouton, l'application affiche un autre écran, c'est typiquement le cas lorsque vous êtes sur un écran d'accueil pour qu'il vous renvoie vers un menu d'options par exemple. Vous pouvez également l'utiliser pour mettre à jour une donnée telle qu'un score sur un jeu vidéo (reset), incrémenter un programme...

CheckBox (Case à cocher)

Similaire au bouton, la case à cocher va notamment réagir lorsque son état (coché ou décoché) est actionné lorsque vous cliquez dessus. Elle possède plus ou moins la même finalité que les boutons à cliquer vus précédemment.

DatePicker (Sélectionneur de date)

Comme son nom l'indique, cet élément vous permet de choisir une date en particulier lorsque vous appuyez sur le bouton correspondant. Il va alors faire apparaître un calendrier en format pop-up :



En fonction de la date choisie, l'administrateur de l'application pourra déterminer toute une série de comportements telle que des informations de type : afficher un message tel que "nous n'avons pas de disponibilité sur ce créneau", afficher la date en question, afficher un stock...

Image

Permet d'insérer une image de votre choix notamment à des fins esthétiques. À noter que les images peuvent être insérées de différentes manières, aussi bien sur les boutons que sur les écrans. Il existe également d'autres images propres à l'animation de l'application que nous verrons plus tard dans ce chapitre.

Label

App Inventor 2 n'a pas donné de traduction en français pour cet élément mais on pourrait le traduire par "Libellé". Il s'agit ni plus ni moins de blocs de texte que vous pouvez insérer au sein de votre application notamment pour afficher un message à l'utilisateur, écrire des titres. Vous pouvez également jouer sur l'aspect visibilité du libellé notamment pour valider les actions de l'utilisateur. Par exemple afficher un libellé de type "votre adresse e-mail a bien été enregistrée" à la suite d'un clic sur un bouton d'envoi d'un formulaire.

ListPicker (Sélectionneur de liste)

Le sélectionneur de liste va afficher un bouton qui, une fois cliqué, vous fera apparaître une liste de choix que vous avez définis. Vous avez également la possibilité d'effectuer une recherche à l'intérieur de cette liste.

ListView (Vue liste)

Ce composant est quasiment identique au précédent à la différence que celui-ci ne se déclenche pas au clic sur un bouton mais se présente directement à vous.

Notifier (Notificateur)

Ce composant affiche des fenêtres d'alerte telles que celles que vous observez sur un ordinateur classique :



Ce composant vous sera très utile si vous souhaitez faire confirmer des actions à l'utilisateur.

PasswordTextBox (Zone texte mot de passe)

Il s'agit d'une zone de texte classique dans laquelle les caractères entrés par les utilisateurs sont remplacés par des points. Naturellement, il s'agira d'une zone de texte que vous utiliserez si l'utilisateur doit entrer des mots de passe.

Slider (Ascenseur)

La version anglaise d'App Inventor 2 utilise le mot "slider" pour définir ce composant, ce qui est peut-être un peu plus juste. Il s'agit d'une barre de jauge vous permettant d'indiquer un niveau d'intensité que l'utilisateur définit en bougeant la barre.



Spinner (Curseur animé)

Le curseur animé est un composant possédant quasiment les mêmes fonctionnalités que les sélectionneurs de liste et les vues liste.

TextBox (Zone de texte)

Les zones de texte permettent à l'utilisateur de saisir du texte, par exemple son nom, son prénom, le contenu d'un sms... Il s'agit d'un des éléments les plus utilisés dans les applications. À noter que l'utilisateur devra cliquer sur le bouton **OK** pour fermer l'affichage du clavier et continuer à utiliser l'application.

TimePicker (Sélectionneur de temps)

À l'image du sélectionneur de date, celui-ci vous permettra de sélectionner une heure précise :

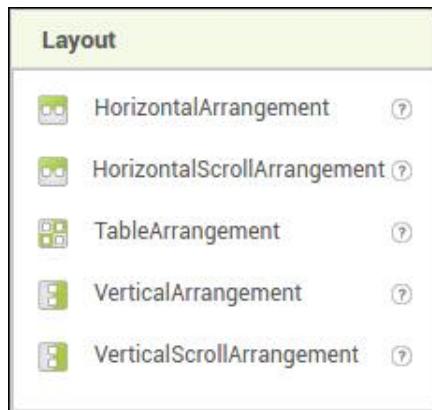


WebViewer (Afficheur Web)

Pour ceux et celles qui connaissent l'univers des sites Internet, l'afficheur web serait l'équivalent de ce que l'on appelle des iframe. L'afficheur web va donc vous permettre d'inclure des pages web au sein de votre application. Vous pouvez également décider de rendre ces pages interactives, les liens seront des cliquables et vous pourrez naviguer à l'intérieur du site au sein de votre application.

Layout (Disposition)

Chaque composant que vous ajoutez à votre application ne peut être placé par défaut à l'endroit de votre choix. Pour ce faire, vous devez utiliser l'un des cinq blocs de disposition ci-dessous :



HorizontalArrangement (Arrangement horizontal)

L'arrangement horizontal va vous permettre de placer vos composants de gauche à droite sur l'écran.

HorizontalScrollViewArrangement

À la différence du précédent, celui-ci va vous permettre de scroller parmi les composants.

TableArrangement (Arrangement tableau)

Ce composant vous permet d'organiser ces derniers sous la forme d'un tableau.

VerticalArrangement (Arrangement vertical)

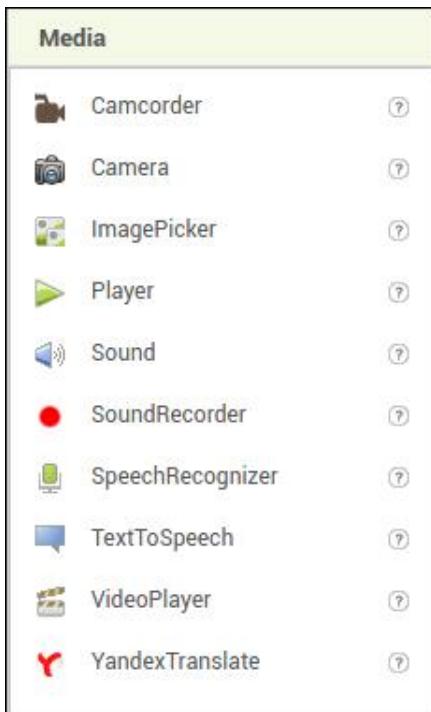
L'arrangement vertical va vous permettre de placer vos composants de haut en bas sur l'écran.

VerticalScrollViewArrangement

À la différence du précédent, celui-ci va vous permettre de scroller parmi les composants.

Media (Média)

Les composants médias permettent d'ajouter de l'interactivité à vos applications :



Camcorder (Caméscope)

Ce composant permet d'utiliser l'appareil photo du téléphone pour enregistrer une vidéo. Celui-ci va alors générer un fichier vidéo que vous pourrez réutiliser par la suite pour la visionner grâce au composant lecteur vidéo.

Camera (Caméra)

Il s'agit ni plus ni moins de l'appareil photo de votre téléphone qui vous permettra donc de photographier ce que vous souhaitez. Une fois la photo prise, un nom de fichier sera généré que vous pourrez utiliser pour l'afficher dans votre application. À noter que le mode frontal n'est pas encore supporté.

ImagePicker (Sélectionneur d'image)

Le sélectionneur d'image permet de parcourir les photos qui se situent à l'intérieur de votre téléphone. Lorsque l'une d'entre elles est sélectionnée, elle est enregistrée dans un endroit dédié de la carte SD de votre téléphone pour vous permettre de l'utiliser comme bon vous semble dans votre application. Vous pouvez ainsi sélectionner un maximum de dix images.

Player (Lecteur)

Il s'agit de l'un des composants incontournables d'App Inventor 2. Celui-ci vous permet de jouer des mélodies ainsi que de faire fonctionner le vibreur de votre téléphone. À noter que tous les types de formats son ne sont pas supportés, veuillez vous référer à la documentation officielle d'Android pour voir si le fichier que vous souhaitez inclure sera supporté : <https://developer.android.com/guide/appendix/media-formats.html>

Sound (Son)

Le composant **Son** a la même finalité que le composant **Lecteur** à la différence que ce premier est principalement

utilisé pour les fichiers de son court.

SoundRecorder (Enregistreur son)

Ce composant permet simplement d'enregistrer les sons qu'il perçoit.

SpeechRecognizer (Reconnaissance vocale)

Ce composant permet de transformer les paroles entendues par l'intermédiaire du microphone de votre smartphone en texte.

TextToSpeech (Texte à parole)

Ce composant permet de lire un texte et d'utiliser la synthèse vocale de votre smartphone pour le prononcer.

VideoPlayer (Lecteur vidéo)

Comme son nom l'indique, il s'agit d'un lecteur vidéo à part entière qui vous permettra de lire les vidéos que vous possédez sur votre smartphone.

À noter que les serveurs d'App Inventor limitent la lecture des fichiers vidéo à 1 MB, ce qui signifie que vous ne pourrez aller très loin en termes de lecture et qu'il va vous falloir l'optimiser en termes de dimensions. Vous pouvez en revanche l'utiliser pour lire une vidéo embarquée sur une URL dédiée.

YandexTranslate (Traduction Yandex)

Ce composant utilise le service de traduction de Yandex, le concurrent russe de Google afin de vous permettre de traduire des phrases à la volée au sein de votre application. Une connexion à Internet est nécessaire pour faire fonctionner ce composant.

Drawing and Animation (Dessin et animation)

Les dessins et animations sont les composants utilisés pour animer votre application, cela peut par exemple être le cas lorsque vous avez envie de créer une bannière animée, un jeu vidéo de type jeu de plateau, faire défiler du texte...



Ball (Balle)

Il s'agit d'un type spécifique d'image. Elle est notamment utilisée dans App Inventor 2 pour faire des jeux de balle tels qu'un pong, des jeux de football... À noter que contrairement aux images lutin que nous verrons très prochainement, vous ne pouvez pas modifier l'image de la balle.

Canvas (Cadre)

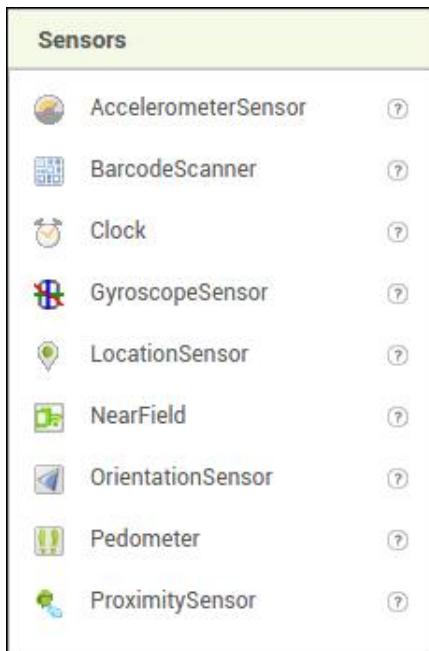
Le cadre est le plateau sur lequel se déplacent les balles et les images lutin. Sans ce dernier, vous ne pourrez faire interagir ces composants, c'est-à-dire que vous ne pourrez pas permettre leur déplacement ni définir des comportements en cas de collision.

ImageSprite (Image lutin)

Une image lutin n'est ni plus ni moins qu'une image que vous insérez et qui pourra avoir une interaction sur l'élément cadre.

Sensors (Capteurs)

Il s'agit d'éléments qui permettent de détecter des comportements tels que du temps qui passe, la position GPS du téléphone, l'orientation de celui-ci.



AccelerometerSensor (Accéléromètre)

L'accéléromètre détecte l'orientation dans l'espace du téléphone en fonction des axes x, y et z du téléphone en fonction de la force de gravité.

BarcodeScanner (Scanneur code à barres)

Grâce à ce composant, votre téléphone sera en mesure de détecter si l'objet que vous mettez devant son champ de vision possède un code à barres. Si tel est le cas, vous pourrez alors définir un comportement.

Clock (Horloge)

Ce composant vous permet d'utiliser des fonctionnalités de temps. Cela vous sera très utile pour définir des comptes à rebours, afficher l'heure, déclencher des comportements au bout d'un temps précis...

GyroscopeSensor (Gyroscope)

L'accéléromètre permet d'identifier les inclinaisons du mobile, le composant Gyroscope possède les mêmes fonctionnalités à la différence que les inclinaisons sont plus sensibles et que celui-ci inclut également la rotation du téléphone. L'idéal pour comprendre la différence est de visionner cette vidéo de Steve Jobs présentant le Gyroscope : <https://www.youtube.com/watch?v=sQQuQL5nOEU> pour l'iPhone

À noter que tous les smartphones n'en sont pas équipés.

LocationSensor (Capteur position)

Ce composant permet de géolocaliser votre appareil et ainsi d'indiquer les coordonnées latitude et longitude de ce dernier. Celui-ci va naturellement utiliser les coordonnées GPS de votre téléphone. Concrètement, avec un tel composant, vous pourriez créer une application vous indiquant où vous avez garé votre voiture. C'est d'ailleurs l'un

des tutoriels de base présentés par AI 2 : <http://appinventor.mit.edu/explore/ai2/android-wheres-my-car.html>. Vous pouvez aussi déclencher des comportements d'objets connectés dès que vous arrivez à proximité de tel ou tel lieu ou de votre domicile.

NearField (Champ proche)

Ce composant est un capteur NFC (*Near Field Communication*) ; le capteur NFC permet l'échange de données entre votre mobile et une cible NFC, plus communément appelée "sans contact". On trouve notamment ce type de fonctionnement dans des drives de supermarché, des abribus. Le NFC est un peu le cousin du QR code à la différence qu'au lieu de flasher une image avec l'appareil photo de votre téléphone, vous devez mettre celui-ci en contact direct avec la cible.

Le composant NFC d'App Inventor 2 va donc vous permettre de lire des bornes NFC.

OrientationSensor (Capteur orientation)

Il s'agit d'un composant qui permet de connaître l'orientation physique du téléphone.

Pedometer (Podomètre)

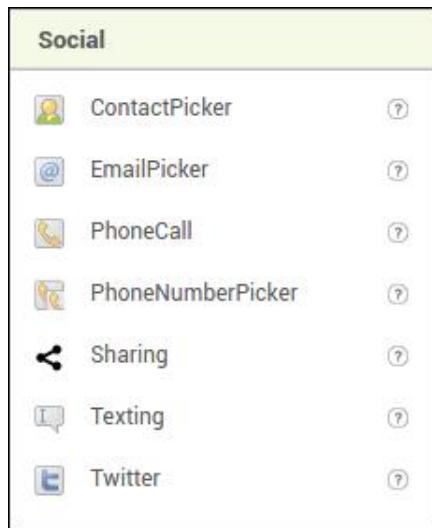
Ce composant calcule le nombre de pas, il utilise pour cela l'accéléromètre du téléphone pour identifier lorsqu'un pas est effectué.

ProximitySensor

Ce composant permet d'identifier la proximité d'un objet en centimètres vis-à-vis de l'écran de votre téléphone. À noter que la distance dépend du terminal utilisé, certains en effet vous indiqueront seulement si l'obstacle est proche ou non sans vous indiquer de distance réelle.

Social

Découvrons maintenant les composants permettant d'effectuer des partages d'information :



ContactPicker (Sélectionneur de contact)

Le sélectionneur de contact est un composant de type bouton qui, quand il est cliqué, va afficher la liste des contacts de votre mobile pour vous permettre d'en choisir un.

Une fois celui-ci sélectionné, vous pourrez avoir accès aux propriétés suivantes :

- Nom du contact,
- Adresse e-mail,
- Photo du contact.

Et ainsi vous permettre de les réutiliser avec un autre composant.

EmailPicker (Sélectionneur e-mail)

Ce composant est une zone de texte que l'utilisateur peut commencer à remplir. Une fois la démarche commencée, un menu déroulant va apparaître affichant les adresses e-mail déjà identifiées dans le carnet de contact de votre téléphone.

PhoneCall (Appel téléphonique)

Ce composant va vous permettre d'utiliser la fonctionnalité appel téléphonique de votre mobile. Il vous suffit pour cela de le relier à un numéro de téléphone, notamment le composant sélectionneur de numéro de téléphone. À noter qu'un format bien spécifique doit être supporté.

PhoneNumberPicker (Sélectionneur numéro téléphone)

À la manière du sélectionneur de contact, il s'agit d'un bouton qui, une fois cliqué, affiche la liste des contacts du téléphone afin de sélectionner celui de votre choix. Une fois sélectionné, les propriétés suivantes vont être proposées :

- Nom du contact,

- Numéro de téléphone,
- Adresse e-mail,
- Photo.

Sharing (Partage)

Ce composant va permettre le partage de fichiers et/ou de messages entre votre application et les autres applications installées sur votre mobile.

Texting (SMS)

Ce composant va permettre l'envoi de SMS depuis votre mobile.

Twitter

Ce composant permet d'utiliser l'API de Twitter pour publier sur le fameux réseau social.

- Pour utiliser ce composant, il faut créer un compte Twitter si vous n'en possédez pas déjà un et obtenir les utilisations de ce dernier en accédant à l'adresse suivante : http://twitter.com/oauth_clients/new

The screenshot shows the Twitter Application Management interface. At the top, there's a navigation bar with the Twitter logo, 'Application Management', 'Have an account? Sign in', and a search bar. Below the bar, a yellow banner displays a cookie consent message: 'By using Twitter's services you agree to our [Cookie Use](#) and [Data Transfer](#) outside the EU. We and our partners operate globally and use cookies, including for analytics, personalisation, and ads.' The main content area is titled 'Twitter Apps' and contains the text 'Please [sign in](#) with your Twitter Account to create and maintain Twitter Apps.' A 'Create New App' button is visible in the bottom right corner.

- Une fois identifié avec votre compte Twitter il faudra créer une nouvelle application :

This screenshot is similar to the previous one but focuses on the 'Create New App' step. It shows the same Twitter Application Management interface with the 'Twitter Apps' section. The 'Create New App' button is now clearly visible in the bottom right corner of the main content area.

- À la suite de cela, il vous suffit de remplir un court questionnaire présentant votre application et de cliquer sur **Create a new application** :

Application Management

By using Twitter's services you agree to our [Cookie Use](#) and [Data Transfer](#) outside the EU. We and our partners operate globally and use cookies, including for analytics, personalisation, and ads.

Create an application

Application Details

Name *
MeasureBowlingManager 

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description *
This App is designed in order to help MeasureBowling organizers.

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

- Une fois cela fait, vous accédez à l'interface des paramètres de Twitter. Cliquez sur **Keys and Access Tokens**, pour afficher l'écran suivant, sur lequel vous trouverez les deux informations dont vous avez besoin pour utiliser le composant, à savoir **Consumer Key** et **Consumer Secret** :

Application Management

By using Twitter's services you agree to our [Cookie Use](#) and [Data Transfer](#) outside the EU. We and our partners operate globally and use cookies, including for analytics, personalisation, and ads.

MeasureBowlingManager

[Test OAuth](#)

[Details](#) [Settings](#) [Keys and Access Tokens](#) [Permissions](#)

Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key) [REDACTED]

Consumer Secret (API Secret) [REDACTED]

Access Level Read and write ([modify app permissions](#))

Owner [REDACTED]

Owner ID [REDACTED]

Storage (Stockage)

Afin de pouvoir stocker des données et les restituer sous différentes formes, App Inventor 2 permet différents systèmes de stockage que nous vous présentons ci-dessous :



File (Fichier)

Le composant **File** sert à stocker des informations au sein de votre téléphone en tant que tel par l'intermédiaire d'un fichier et non d'une base de données ou d'une application que vous avez créée. Généralement, cela se fait par l'intermédiaire d'un fichier .txt.

FusiontablesControl

Ce composant permet d'envoyer des données dans une base qui appartient au service de Google : Google Data Fusion Table. Il s'agit d'un service un peu similaire à Google Spreadsheet, à la différence que Google Data Fusion Table est beaucoup plus orienté base de données.

Ce service permet donc de stocker de la donnée pour une réutilisation future.

Les applications qui utilisent Google Data Fusion Tables doivent s'authentifier auprès de Google. Nous reviendrons plus tard dans cet ouvrage sur la manière dont vous pouvez les utiliser.

Pour plus d'informations : https://developers.google.com/fusiontables/docs/v1/getting_started.

TinyDB

TinyDB est une mini base de données, propre à votre application. Elle permet donc de stocker de la donnée, comme le score d'un jeu pour pouvoir le restituer à chaque fois que l'application est utilisée.

TinyWebDB

TinyWebDB a la même finalité que TinyDB, sauf que TinyWebDB stocke la base de données sur le Web et non au sein de votre application. Il vous faudra pour cela trouver un prestataire vous permettant d'héberger votre base de données.

Connectivity (Connectivité)

Découvrons maintenant les composants de connexion possible avec App Inventor 2 :



ActivityStarter (Déclencheur d'activité)

Probablement l'un des composants les plus techniques d'App Inventor 2, celui-ci permet de lancer d'autres applications à partir de l'application que vous avez créée. Il permet également de lancer de nombreuses autres actions telles que l'envoi d'e-mails, le lancement de vidéos YouTube, de votre navigateur.

BluetoothClient (Client Bluetooth)

Ce composant va permettre d'utiliser votre mobile en tant que client Bluetooth.

BluetoothServer (Serveur Bluetooth)

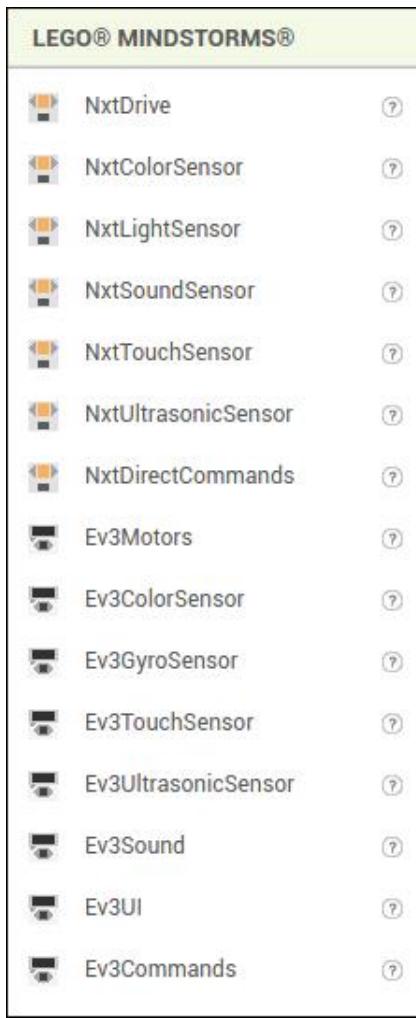
Ce composant va permettre d'utiliser votre mobile en tant que serveur Bluetooth.

Web

Ce composant permet d'envoyer des requêtes web. Typiquement, c'est le genre de composant que l'on utilise si on veut incorporer des outils tels que Google Analytics à son application.

LEGO® MINDSTORMS®

Vous avez bien lu, il est possible d'utiliser App Inventor 2 avec les produits de la célèbre marque danoise :



App Inventor 2 reconnaît pour le moment deux types de technologies NXT et EV3. NXT étant la version qui précédait EV3. Cette technologie étant difficile à trouver dans le commerce, nous ne détaillerons qu'EV3 dans cet ouvrage. L'une et l'autre représentent une technologie développée par l'entreprise danoise LEGO® et permettent d'embarquer de la programmation informatique à l'intérieur de vos projets.

LEGO et MINDSTORMS sont des marques déposées du groupe LEGO.

Probablement pour des questions d'accessibilité, des composants spécifiques à LEGO Mindstorms ont été développés pour App Inventor 2.

LEGO Mindstorms est un kit de LEGO permettant de créer votre propre robot comprenant de l'informatique embarquée. Cela signifie qu'en plus des classiques briques de LEGO, la motorisation intégrée par l'intermédiaire de piles (rotation des roues, jeux de lumière), LEGO embarque une technologie informatique permettant d'indiquer à ce que vous avez construit, les moments auxquels vous souhaitez déclencher les motorisations.

Le fait de pouvoir faire un pont entre la technologie d'App Inventor 2 et celle de LEGO va vous permettre de contrôler le robot avec les programmes que vous allez réaliser sur App Inventor 2.

La robotique a malheureusement un prix, et même s'il s'agit de LEGO, le coût du pack LEGO Mindstorms est de l'ordre de 350 € minimum neuf (prix Amazon) dont le kit EV3 Intelling Brick (le cœur du robot) vaut 199 €, source :

Le kit contient également d'autres composants électroniques dont :

- 1 capteur infrarouge,
- 3 moteurs.

LEGO Mindstorms vous permettra une introduction à l'univers de la robotique, nous en donnerons quelques exemples dans ce livre.



Pour ceux et celles qui souhaitent tenter l'aventure LEGO Mindstorms, voici quelques informations complémentaires.

Tout d'abord, ce qu'il faut savoir, c'est que votre appareil mobile pourra communiquer avec le robot LEGO Mindstorms uniquement par Bluetooth, d'où l'intérêt de déjà maîtriser ce composant. LEGO mindstorms est fourni avec une télécommande IR, cependant tous les smartphones n'en étant pas équipés, App Inventor 2 ne reconnaît pas encore ce matériel, le Bluetooth sera donc le seul réseau vous permettant d'envoyer des instructions à LEGO Mindstorms.

Les robots LEGO Mindstorms se montent assez rapidement, par exemple le robot principal se monte en 21 étapes (une demi-journée à une journée de travail).

Pour les besoins du contenu que nous allons présenter ci-après, nous avons fait le choix de construire le robot GRIPP3R qui a l'avantage d'être un robot dont les plans sont conçus par LEGO et qui sont suffisamment documentés.

À noter que LEGO Mindstorms possède ses propres logiciels vous permettant déjà de pouvoir prendre le contrôle du robot, cependant nous allons voir comment App Inventor 2 peut apporter une véritable valeur ajoutée au programme.

Pour en savoir plus sur le composant LEGO Mindstorms :
<http://ai2.appinventor.mit.edu/reference/components/legomindstorms.html>

Ev3Motors

Cette fonctionnalité va vous permettre de contrôler les moteurs de votre robot. Vous allez notamment l'utiliser pour le diriger, par exemple pour le faire avancer ou le faire reculer.

Ev3ColorSensor

Ce composant permet d'indiquer à votre robot LEGO les couleurs que la brique LEGO est en train de rencontrer. Par exemple en faisant pointer ce capteur vers le sol vous pouvez indiquer à votre robot d'arrêter de se déplacer s'il rencontre une couleur que vous lui avez indiquée, par exemple si jamais il rencontre une couleur marron alors que le carrelage de la cuisine est de couleur blanche, alors il doit s'arrêter car cela signifie qu'il vient d'entrer dans une chambre où le parquet est marron.

Ev3GyroSensor

Il s'agit d'un composant LEGO qui inclut un gyroscope et permet donc d'indiquer la position du robot dans l'espace.

Ev3TouchSensor

Il s'agit d'un composant de pression permettant de savoir si le robot a rencontré un obstacle.

Ev3UltrasonicSensor

Très populaires dans l'univers des objets connectés, les capteurs ultrasoniques permettent de remonter la distance entre le robot et un obstacle rencontré, notamment utile pour comptabiliser des passages devant le robot ou bien pour l'empêcher d'entrer en collision avec un futur obstacle.

Ev3Sound

Il s'agit d'un composant permettant d'émettre des sons.

Ev3UI

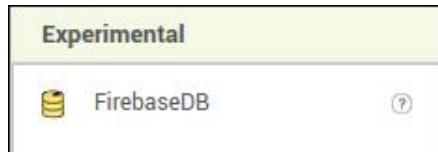
Il s'agit d'un composant permettant d'afficher des informations dans la console de LEGO Mindstorms.

Ev3Commands

Ce composant permet de lui envoyer des instructions diverses.

Experimental (Expérimental)

La rubrique **Experimental** contient, comme son nom l'indique, des composants à titre de test, c'est le cas, à la date de rédaction de l'ouvrage, de Firebase :



Rachetée en octobre 2014 par Google, Firebase est un prestataire de services dans le domaine du Cloud Computing. Il fournit de nombreux services pour les applications pour mobile. Dans le cas d'App Inventor 2, le service fourni est le plus populaire à savoir FirebaseDatabase, une base de données NoSQL, c'est-à-dire une base de données réputée pour sa flexibilité et pour être moins compliquée que des bases de données SQL traditionnelles.

Cette base de données a également pour particularité de stocker les données sous la forme de documents JSON (format de données très utilisé à la date de rédaction de cet ouvrage) et de traiter la donnée en temps réel, de la mettre à disposition sur tous les postes clients sans avoir besoin de recharger l'application.

À noter qu'il s'agit ici d'une fonctionnalité expérimentale qui peut potentiellement ne plus être fonctionnelle à la date de sortie de cet ouvrage et qu'App Inventor 2 utilise un compte Firebase spécial (c'est-à-dire indépendamment de celui que vous pourriez créer en vous rendant sur le site officiel) qui ne peut être utilisé qu'à des fins personnelles et expérimentales et non pour un usage professionnel.

Pour faire simple, un composant tel que Firebase va trouver son utilité pour des projets où le partage d'informations en temps réel est important. On peut imaginer une application de jeu téléchargée par une dizaine d'utilisateurs, lorsqu'un score maximum est atteint celui-ci est mis à jour dans la base de données de Firebase et tous les utilisateurs de l'application voient en temps réel le nouveau score affiché.

Quelles sont les différences entre Firebase et TinyWebDB ?

Tout d'abord, Firebase est une fonction expérimentale. Bien que puissante à la date de la rédaction de cet ouvrage, il n'est pas dit qu'elle soit supportée dans le futur.

L'événement que nous verrons plus tard, DataChanged permet de mettre à jour automatiquement un écran d'application sans mise à jour manuelle.

Les variables utilisées dans TinyWebDB peuvent entrer en conflit entre différentes applications ce qui est problématique et nécessite une configuration spécifique.

Pour en savoir plus à ce sujet : <http://appinventor.mit.edu/explore/content/custom-tinywebdb-service.html>

Pour en savoir plus sur Firebase : <https://firebase.google.com/>

Extension

Comme nous avons pu le découvrir dans le chapitre précédent, il est possible d'ajouter des extensions à App Inventor 2 afin de rendre celui-ci plus fonctionnel.

C'est dans cette rubrique que vous pourrez importer des extensions téléchargées sur d'autres sites :



Conclusion

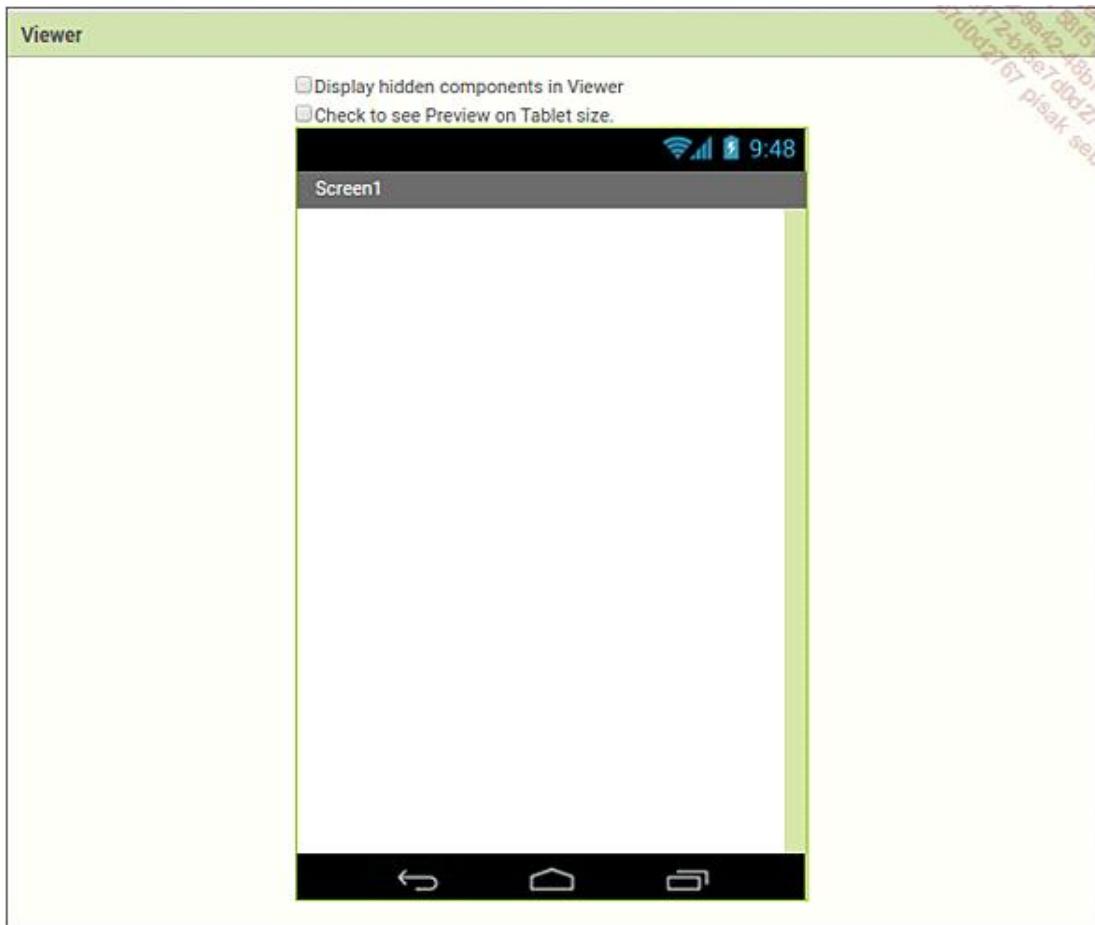
Ce chapitre se termine. Vous connaissez désormais les différents composants que vous pouvez ajouter à App Inventor 2. À ce stade nous ne savons pas encore les utiliser et créer des applications avec, mais c'est ce que nous découvrirons dans le chapitre Les propriétés. Mais avant tout, consacrons un peu de temps au design de notre application...

Introduction

Maintenant que nous connaissons les différents composants que nous pouvons inclure au sein de nos applications, voyons comment nous pouvons les positionner. Ce chapitre aura également pour objectif d'aborder la partie réalisation graphique des applications sous AI 2. Souvent, les personnes manipulant sur App Inventor 2 ont tendance à négliger cet aspect. Cela a pour conséquence de présenter des applications peu esthétiques et qui malheureusement ne donnent pas envie de tester App Inventor 2. Découvrons dans ce chapitre comment éviter cela.

Charte graphique

La partie graphique de votre application s'effectue dans la zone **Viewer**.

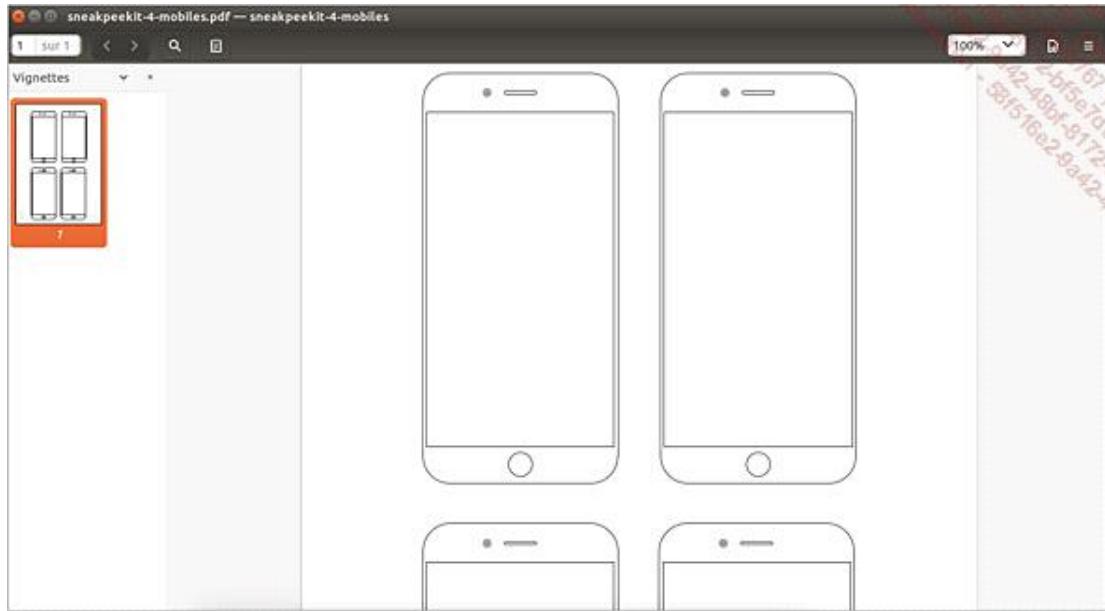


 Avant de vous lancer dans la réalisation graphique de votre application, nous vous conseillons très fortement de vous reporter au chapitre Le cahier des charges relatif à la rédaction du cahier des charges. En effet, celui-ci comprend la réalisation des différentes vues de votre application.

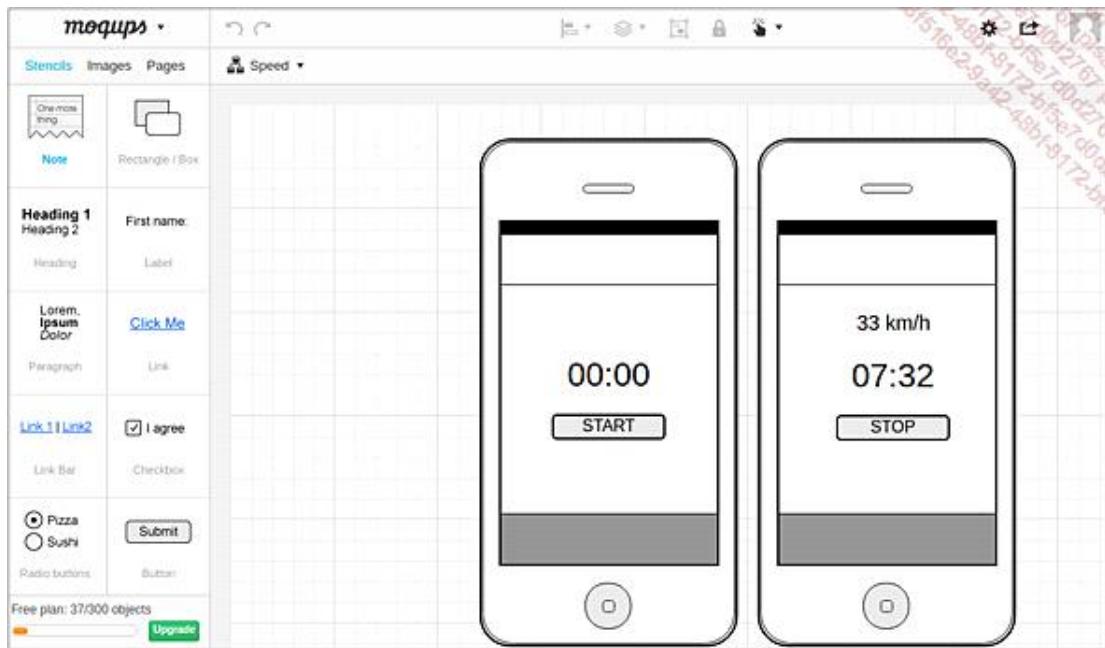
Nous vous conseillons en amont d'utiliser des logiciels tels que :

- GIMP : <https://www.gimp.org/>
- Photoshop : <http://www.adobe.com/fr/products/photoshop.html>
- Moqups : <https://moqups.com/>
- Balsamiq : <https://balsamiq.com/>
- Marvel : <https://marvelapp.com/>

Vous pouvez également travailler sur papier en imprimant des croquis déjà tout faits de skins pour mobile, par exemple sur : <http://sneakpeekit.com/>.



L'idée étant de réaliser des "croquis" de ce à quoi votre application va ressembler :

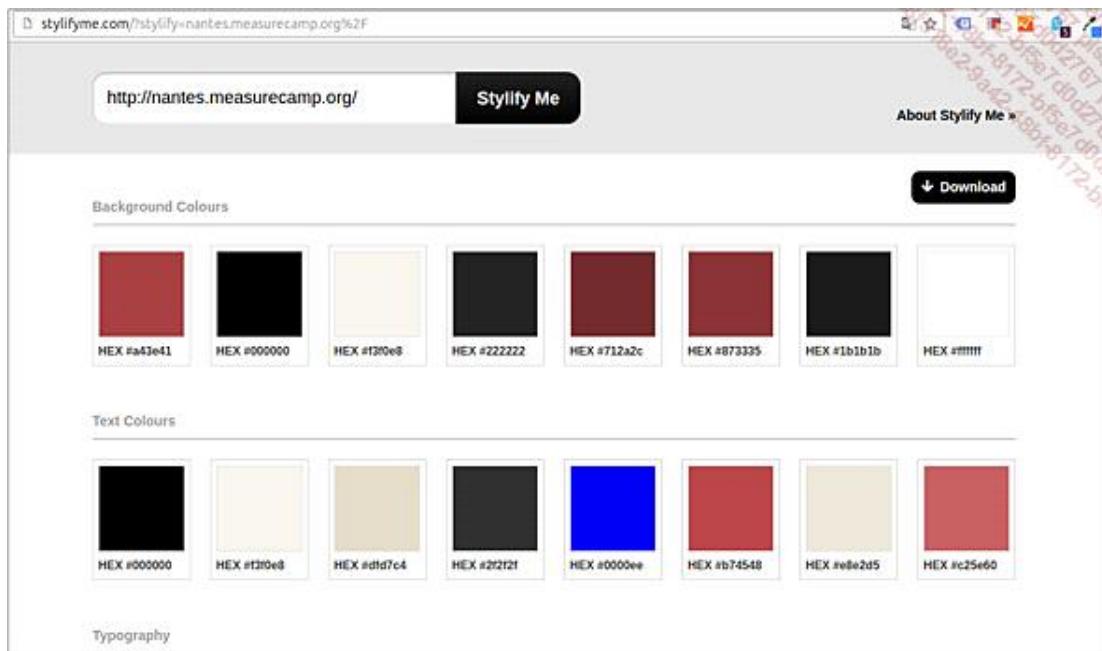


En réalisant ce croquis, vous allez rapidement vous rendre compte si l'interface utilisateur que vous aimerez obtenir sera fonctionnelle ou non.

Récupérer la charte graphique d'un site Internet

Lorsque vous devez réaliser une application pour un tiers, vous verrez qu'il est souvent délicat de savoir quel style lui donner. L'un des éléments qui peuvent vous aider est de connaître l'ensemble des palettes de couleurs que l'entreprise utilise sur son site Internet (la charte graphique) afin d'avoir une application mobile cohérente avec les designs déjà créés par le passé.

Pour vous aider, il existe des sites tels que <http://stylifyme.com/> qui vous permettent de récupérer en un clic l'ensemble des éléments qui composent la charte graphique d'un site Internet. Ainsi, vous connaîtrez facilement les couleurs que vous devez utiliser :



Une fois les palettes de couleur identifiées, il sera beaucoup plus simple de pouvoir les utiliser dans App Inventor 2.

Par défaut, AI 2 possède une série de couleurs limitée. Afin de pouvoir les personnaliser, vous devez connaître les codes RGB de chacune d'entre elles. Le site <http://www.rapidtables.com/convert/color/hex-to-rgb.htm> vous aidera à convertir les codes hexadécimaux des couleurs en RGB.

Vous pouvez également utiliser des applications telles qu'Eye Dropper pour trouver facilement la couleur d'une zone d'un site Internet : <https://chrome.google.com/webstore/detail/eye-dropper/hmdcmflkchdmnmnmheododdhjedfccka/related>

Comme vous pourrez le constater dans les prochains chapitres, vous pouvez ajouter n'importe quelle couleur dans App Inventor 2 grâce aux blocs de code.

Vous constaterez également que dans App Inventor 2, les boutons de sélection n'offrent pas beaucoup de fonctionnalités en termes de design, par exemple :



Afin d'y remédier, vous pouvez réaliser des boutons avec plus d'allure en utilisant des sites tels que <http://dabuttonfactory.com/>.



De la même manière, vous verrez qu'il n'est pas toujours ais  de mettre en forme le composant bloc de texte. En effet, si vous souhaitez que celui-ci soit format  d'une certaine mani re, il vous faudra utiliser des outils tels que <https://html-online.com/editor/>.

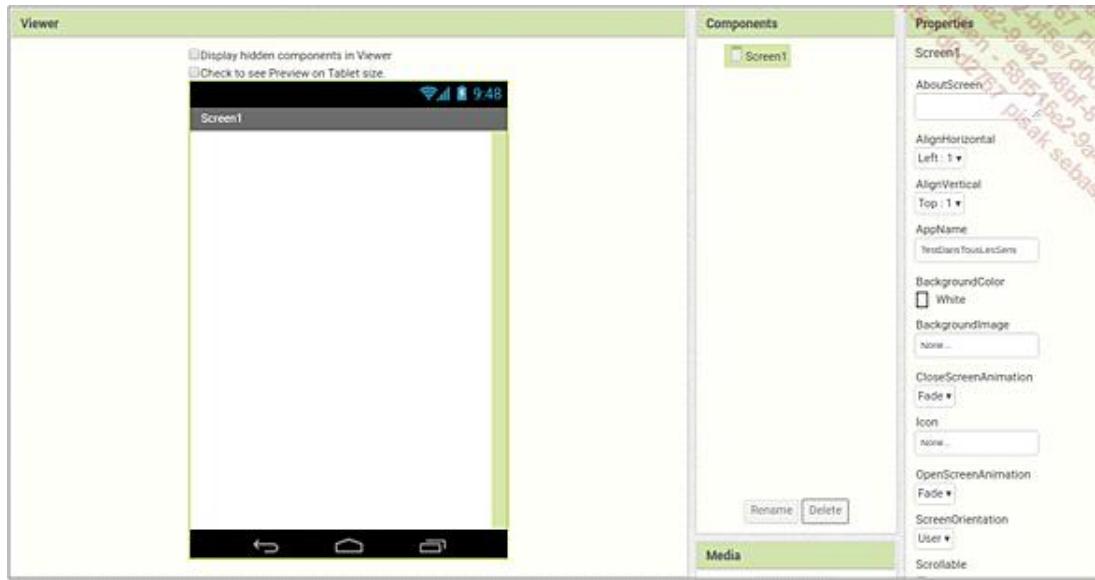
Conclusion

Vous savez désormais qu'avant de commencer votre projet, il faut avant tout imaginer ce à quoi il va ressembler. Vous avez également une idée des différents composants que vous pourrez utiliser afin de lui donner l'aspect esthétique voulu. Voyons désormais ce qui se cache derrière chacun de ces composants.

Introduction

Nous allons découvrir dans ce chapitre comment personnaliser chacun des composants d'App Inventor 2 grâce aux propriétés.

Chaque composant d'App Inventor 2 possède des propriétés qui lui sont propres et que vous pouvez configurer directement dans la partie **Designer** d'AI 2. Lorsque vous sélectionnez dans la partie **Components** l'élément de votre choix, ses propriétés apparaissent dans la colonne **Properties** visible sur la droite :



Ci-dessus, la partie **Properties** du composant **Screen** (écran) s'affiche lorsqu'on le sélectionne.

Nombre de ces propriétés pourront être redéfinies par la suite ; cependant, en les paramétrant au lancement de votre application, vous gagnez du temps et leur utilisation sera plus accessible.

Découvrons ci-dessous l'ensemble des propriétés pour chacun des composants disponibles dans App Inventor 2.

Catégorie User Interface

1. Les propriétés du composant Screen (écran)

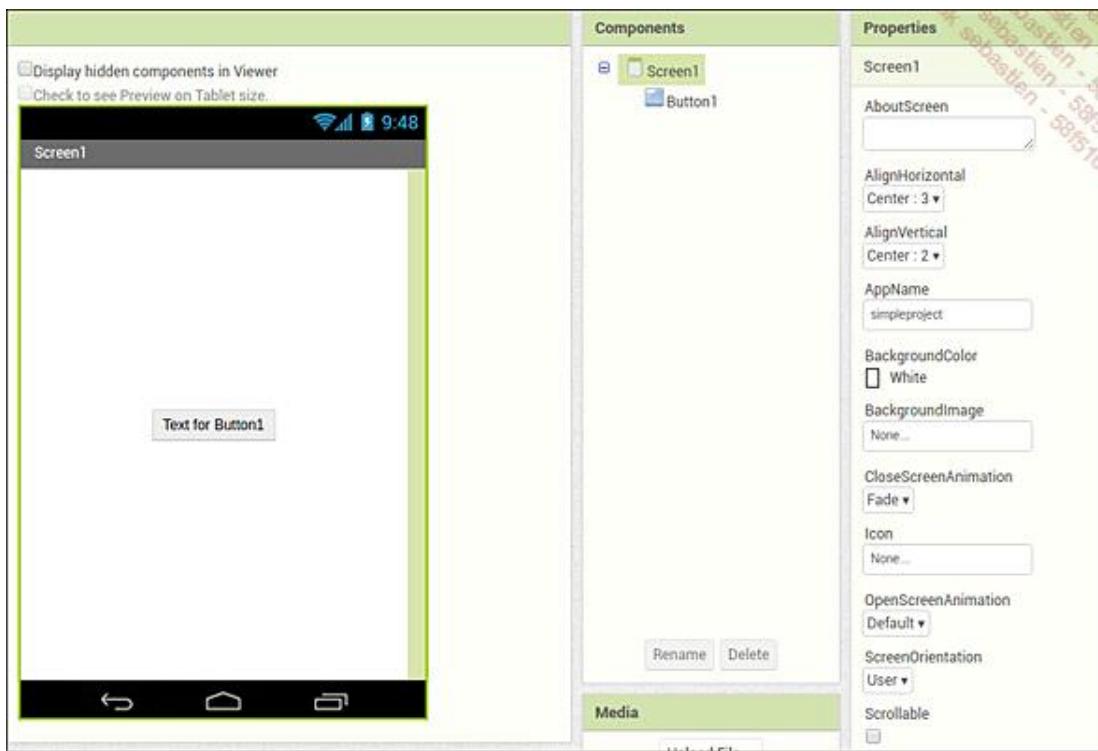
AboutScreen (À propos de l'écran)

Vous permet d'indiquer des informations additionnelles sur l'écran affiché.

AlignHorizontal (Alignment horizontal)

Il s'agit de l'alignement général de l'écran. Par défaut, les composants affichés sur l'écran sont alignés sur le coin en haut à gauche de l'écran. Vous êtes libre de le définir en trois états : **Left**, (gauche), **Center** (centre), **Right** (droite).

Par exemple, si vous sélectionnez **Center**, tous les composants que vous allez ajouter seront par défaut centrés sur l'écran comme le montre l'image ci-dessous :



AlignVertical (Alignment vertical)

Définit l'affichage des éléments à l'écran de façon verticale. Trois états sont disponibles : **Top** (haut), **Center** (centre) ou **Bottom** (bas).

AppName (Nom de l'application)

Il s'agit du nom que vous souhaitez donner à votre application. À noter que cela ne changera pas le nom donné initialement à l'application, mais vous permettra de le faire apparaître quand vous la téléchargerez ou la lancerez depuis votre téléphone.

BackgroundColor (Couleur de fond)

Permet de personnaliser la couleur de fond de l'écran.

BackgroundImage (Image de fond)

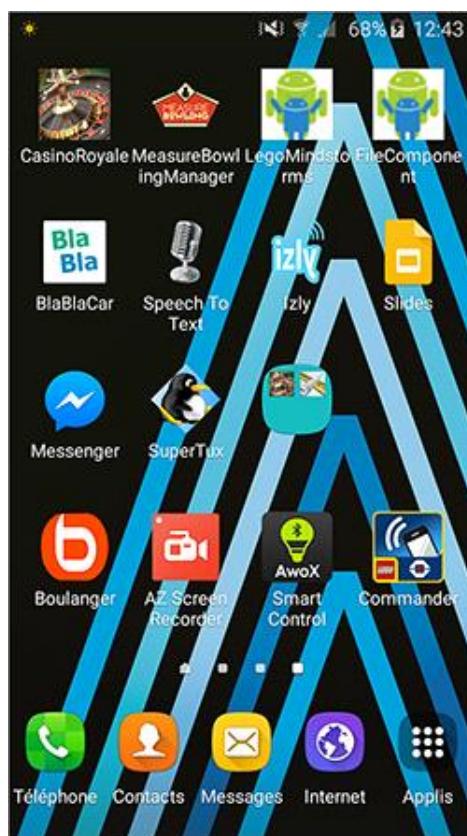
Permet de mettre en place une image de fond pour cet écran.

CloseScreenAnimation (Animation fermeture écran)

Il s'agit d'une option fort intéressante qui permet de donner de l'interaction à votre application. Cette animation apparaît lors du passage d'un écran vers un autre et peut permettre une meilleure expérience utilisateur (c'est surtout plus esthétique). Les valeurs que peut prendre cette animation sont les suivantes : **Fade**, **Zoom**, **SlideHorizontal (Glissement horizontal)**, **SlideVertical (Glissement vertical)**.

Icon (Icône)

Il s'agit de l'icône qui sera reprise une fois l'application installée sur le mobile de l'utilisateur. Privilégiez ici une icône carrée de bonne qualité.



Dans l'écran ci-dessus, la première ligne n'est composée que d'applications réalisées avec App Inventor 2. L'option **Icon** a donc permis de personnaliser les icônes affichées lors de l'installation de l'application.

OpenScreenAnimation (Animation ouverture écran)

Similaire à l'animation de fermeture d'écran, sauf qu'elle apparaîtra à l'ouverture de l'écran.

ScreenOrientation (Orientation écran)

Cette propriété va déterminer la manière dont l'écran va apparaître aux yeux de l'utilisateur. Il peut prendre les valeurs : **Unspecified** (indéterminé) (c'est-à-dire que le mobile réagira en fonction de la position que vous prenez), **Portrait**, **Landscape** (paysage), **Sensor** (capteur) et **User** (utilisateur). À noter que les valeurs que vous spécifiez ici sont propres à chaque écran et devront donc être définies en conséquence.

Scrollable (Défilant)

Cela signifie que vous autorisez la possibilité de dérouler l'écran : souhaitez-vous que l'on puisse "scroller" l'écran ?

ShowStatusBar

Il s'agit de la barre noire tout en haut de votre écran qui vous indique le pourcentage de charge de votre batterie, le fait que vous ayez accès à Internet... Vous pouvez décider dans votre application de ne pas permettre l'affichage de cette barre de notification en décochant cette case.

Sizing

Vous avez le choix ici entre la dimension fixe et le fait de rendre votre application responsive. En choisissant l'option **responsive**, vous aurez une application dont les composants s'afficheront et s'adapteront en fonction de la taille de l'écran. Les deux méthodes présentent des avantages et des inconvénients.

Title (Titre)

Il s'agit du titre de l'écran qui apparaîtra juste en dessous de la barre des statuts.

TitleVisible

Cette case à cocher vous permet de faire apparaître ou non le titre que vous avez défini pour cet écran.

VersionCode

Cette propriété est utilisée lorsque vous mettez votre application sur le Play Store. Elle permet d'indiquer que la version de votre application est différente de celles déjà uploadées.

VersionName (Nom version)

Il s'agit d'une valeur à renseigner afin de faire la distinction entre les différentes versions uploadées de votre application dans le Play Store.

2. Les propriétés du composant Button (bouton)

BackgroundColor (Couleur de fond)

Cette propriété vous permet de personnaliser la couleur du bouton en utilisant l'une des couleurs proposées : noir, bleu, cyan, gris foncé, gris, vert, gris clair, magenta, orange, rose, rouge, blanc, jaune. Si aucune d'entre elles ne correspond à vos besoins, vous aurez la possibilité de la modifier par la suite grâce aux blocs logiques. Vous pouvez la modifier soit en cliquant directement sur la couleur choisie ou en la personnalisant avec des combinaisons de couleurs RGB.

Enabled (Activé)

Cette option permet d'indiquer si vous souhaitez que le bouton déclenche une activité si une interaction est faite dessus, telle qu'un clic. En général, vous laisserez souvent cette option cochée et utiliserez plutôt l'aspect visible ou non visible du bouton. Si cette case est décochée, alors le bouton aura un aspect grisé indiquant qu'il n'est pas activé.

FontBold (Gras)

Permet de mettre en gras le texte rédigé à l'intérieur du bouton.

FontItalic (Italicque)

Permet de mettre en italique le texte rédigé à l'intérieur du bouton.

FontSize (Taille de police)

Permet de définir la taille de la police insérée à l'intérieur du bouton.

FontTypeface (Type de police)

Permet de définir le type de police que vous souhaitez avoir pour ce bouton. À la date de rédaction de l'ouvrage, il semblerait qu'il n'existe pas de possibilité d'ajouter d'autres polices de caractères que sans serif, serif et monospace. Une solution est d'utiliser des logiciels type Photoshop et d'insérer des images en tant que boutons.

Height (Hauteur)

Il s'agit de la hauteur du bouton que vous pouvez définir en remplaçant toute la hauteur de l'écran disponible **fill parent** (en prenant en considération la taille des autres éléments), en pixel ou en pourcentage.

Width (Largeur)

Il s'agit de la largeur du bouton, que vous pouvez définir en remplaçant toute la largeur de l'écran disponible **fill parent** (en prenant en considération la taille des autres éléments), en pixel ou en pourcentage.

Image

Cette propriété va vous permettre de changer le bouton par une image de votre choix.

Shape (Forme)

Vous pourrez ici définir la forme de votre bouton : soit ovale, soit rectangulaire soit avec les bords légèrement arrondis.

ShowFeedback (Montrer réaction)

Derrière cet intitulé difficilement compréhensible, se cache le fait de montrer concrètement que le bouton a été appuyé. Cela se symbolise à l'écran par le fait que le bouton semble légèrement enfoncé lorsqu'on appuie dessus. Ne pas activer cette case vous donnerait l'impression que le bouton est complètement statique.

Text (Texte)

Le texte qui apparaîtra sur le bouton.

TextAlignment (Alignment texte)

Cette propriété est relative au positionnement du texte à l'intérieur du bouton. Si en revanche vous souhaitez déplacer le bouton, il faudra pour cela le mettre dans un bloc de disposition horizontal ou vertical.

TextColor (Couleur texte)

Il s'agit de la couleur du texte du bouton que vous pouvez choisir parmi noir, bleu, cyan, gris foncé, gris, vert, gris clair, magenta, orange, rose, rouge, blanc, jaune.

Visible

Cette propriété permet d'indiquer si vous souhaitez que le bouton soit visible ou pas. Un bloc logique permettra de le faire apparaître en fonction de comportement donné.

3. Les propriétés du composant CheckBox (case à cocher)

Les propriétés des cases à cocher sont quasi identiques à celles des boutons. La seule différence est l'option **Checked (Vérifié)**, qui vous permet une fois cochée de faire en sorte que la case soit cochée par défaut lors du lancement de l'écran.

4. Les propriétés du composant DatePicker (Sélectionneur de dates)

S'agissant à la base d'un simple bouton, celui-ci possède exactement les mêmes propriétés que le composant **Button**.

5. Les propriétés du composant Image

RotationAngle (Angle de rotation)

Permet d'indiquer l'inclinaison de l'image.

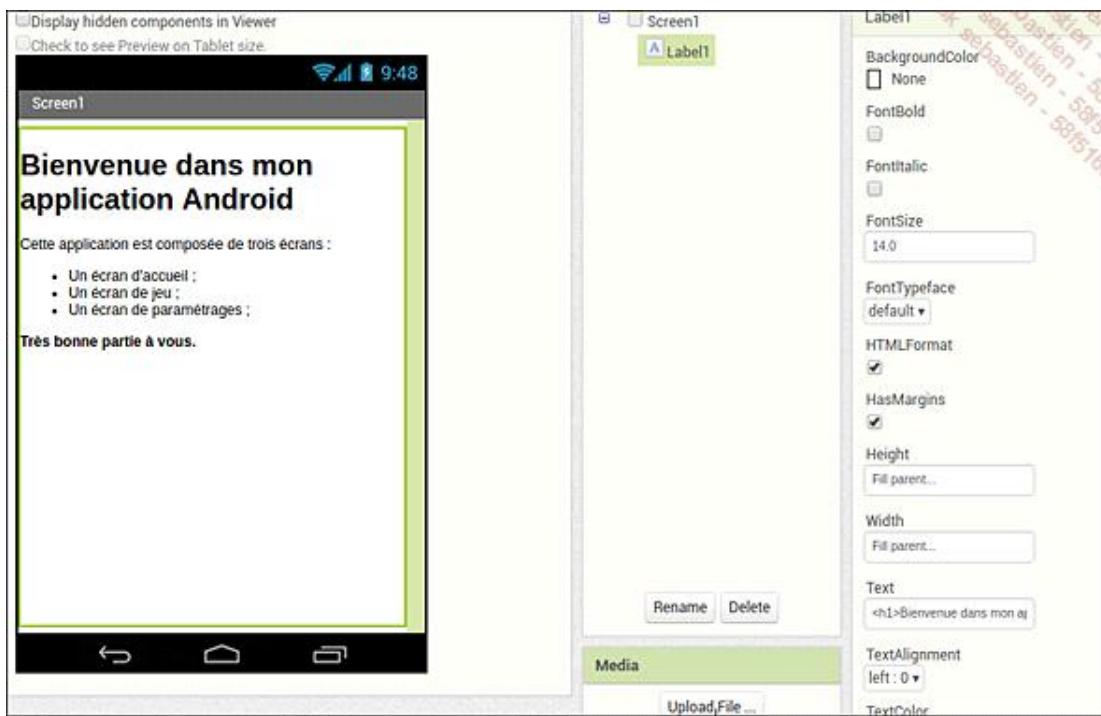
ScalePictureToFit (Redimensionnement image pour ajustement)

Si vous cochez cette case, l'image que vous importerez sera automatiquement adaptée au format de l'écran de l'internaute ; si vous importez une image trop grande et que vous ne cochez pas cette case, l'utilisateur ne la verra pas correctement.

6. Les propriétés du composant Label

HTMLFormat

Cette propriété va vous permettre de pouvoir écrire en HTML à l'intérieur du **Label** et ainsi de lui donner une mise en forme similaire à celle d'une page web. C'est notamment utile pour y copier-coller un bloc de texte :



À noter que certains éléments HTML peuvent ne pas être supportés une fois transférés sur le téléphone ; c'est par exemple le cas ici, avec les listes à puces.

HasMargins

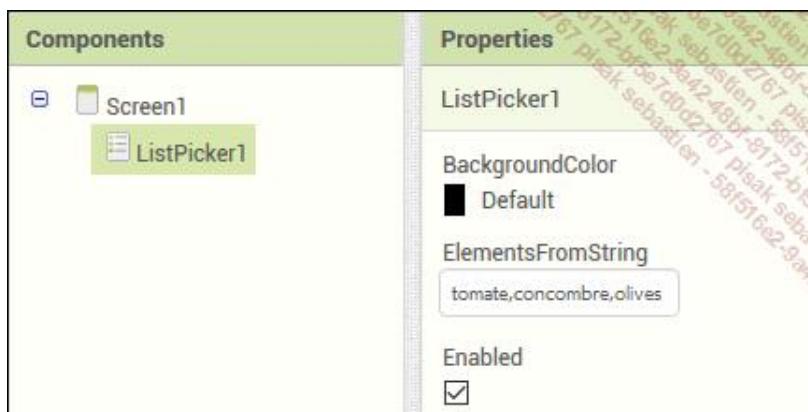
Permet d'ajouter une légère marge à gauche de l'écran.

7. Les propriétés du composant ListPicker (Sélectionneur de liste)

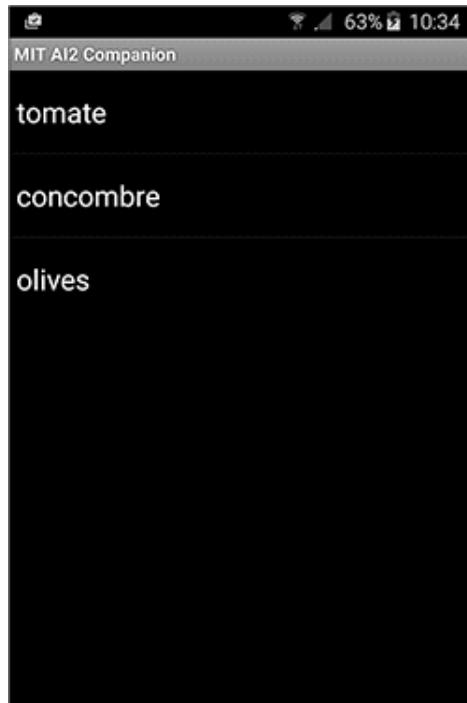
Le sélectionneur de liste possède de nombreuses propriétés en commun avec des composants précédemment cités, nous allons ici nous intéresser uniquement aux propriétés inédites.

ElementsFromString (Éléments de la chaîne)

Il s'agit des éléments qui seront affichés lorsque vous cliquerez sur la liste, chaque élément doit être séparé par une virgule, par exemple :



Ce qui affichera à l'écran après un clic sur le bouton :



ItemBackgroundColor

Cette propriété signifie que vous pouvez personnaliser la couleur de fond d'écran des éléments qui composent la liste.

ItemTextColor

Vous pouvez décider dans cette propriété de la couleur de texte que vont prendre ces éléments.

Sélection

Il s'agit des valeurs que va prendre chacun des éléments lorsque vous les sélectionnerez.

8. Les propriétés du composant ListView (Vue liste)

Ce composant possède de nombreuses propriétés en commun avec des composants précédemment cités, nous allons ici nous intéresser uniquement à la propriété **SelectionColor**.

SelectionColor

Il s'agit de la couleur que prend l'élément de la liste que vous sélectionnez.

9. Les propriétés du composant Notifier (Notificateur)

NotifierLength (TailleNotificateur)

Vous avez ici le choix entre deux options : court (**Short**) et long (**Long**).

10. Les propriétés du composant PasswordTextBox (Zone de texte mot de passe)

Hint (Nuance)

Derrière ce titre énigmatique, cette option permet d'indiquer à l'utilisateur le texte qu'il doit entrer dans la rubrique correspondante. La version anglaise est plus précise en utilisant le mot "hint" (indice).

Text

Valeur texte entrée par défaut dans le champ.

11. Les propriétés du composant Slider (Ascenseur)

ColorLeft (Couleur gauche)

Il s'agit de la couleur située à gauche du curseur.

ColorRight (Couleur droite)

Il s'agit de la couleur située à droite du curseur.

MaxValue (Valeur maximale)

Il s'agit de la valeur maximale que pourra prendre l'ascenseur.

MinValue (Valeur Min)

Il s'agit de la valeur minimale que pourra prendre l'ascenseur.

ThumbEnabled

Si cette case est décochée, cela signifie que le curseur de l'ascenseur n'apparaîtra pas et ne pourra pas être utilisé.

ThumbPosition (Position puce)

Indique à quel endroit le curseur de l'ascenseur sera situé sur l'échelle.

12. Les propriétés du composant Spinner (Curseur animé)

Prompt (Invite)

Cette propriété fait référence au titre qui sera affiché lorsque l'on va cliquer sur le sélectionneur de liste.

13. Les propriétés du composant TextBox (Zone de texte)

Multiline (Multi-lignes)

Une fois cochée, cette case permet de gérer les retours à la ligne si votre texte est trop long.

NumbersOnly (Nombres uniquement)

Lors d'une saisie, le clavier affiché ne montrera que des nombres :



14. Les propriétés du composant WebViewer (Afficheur web)

FollowLinks (Suivre liens)

Si cette case est décochée, cela signifie que tout lien cliqué à l'intérieur de l'afficheur web ne vous renverra pas sur une page web. Cela permet de garder le contrôle sur la page en question.

HomeURL (Url Accueil)

Correspond à l'URL que vous souhaitez voir charger dans l'écran de l'application :



IgnoreSslErrors

Cela signifie que si la page affichée n'est pas sécurisée, votre application vous empêchera d'y accéder.

PromptforPermission (Invite pour permission)

Si la case est cochée, cela signifie que l'application doit vous demander votre permission pour vous géolocaliser. Si la case est décochée, cela signifie que vous avez donné votre permission.

UsesLocation (Utilisation position)

Parfois demandée par certains sites pour afficher des informations personnalisées.

Catégorie Layout

1. Les propriétés du composant HorizontalArrangement (Arrangement horizontal)

AlignHorizontal (Alignement horizontal)

Cette propriété vous permet de définir l'endroit où vous souhaitez que soient alignés les composants inclus à l'intérieur du bloc alignement horizontal : **gauche, centrer, droite**.

AlignVertical (Alignement vertical)

Cette propriété vous permet de définir l'endroit où vous souhaitez que soient alignés les composants inclus à l'intérieur du bloc alignement vertical : **haut, centrer, bas**.

Toutes les autres propriétés ont déjà été présentées précédemment, nous ne reviendrons pas dessus.

2. Les propriétés du composant TableArrangement (Arrangement_tableau)

Columns (Colonnes)

Cette propriété indique le nombre de colonnes que vous souhaitez avoir dans le bloc arrangement tableau.

Rows (Lignes)

Cette propriété indique le nombre de lignes que vous souhaitez avoir dans le bloc arrangement tableau.

Catégorie Média

1. Les propriétés du composant Player (Lecteur)

Loop (Boucle)

Cette propriété permet d'indiquer si vous souhaitez qu'à la fin de la musique celle-ci soit rejouée de manière indéfinie.

PlayOnlyInForeground (Joué seulement au premier plan)

Si la case est cochée, cela signifie que le lecteur sera mis en pause si l'écran est quitté, si la case est décochée le lecteur continuera à jouer.

Source

Permet d'indiquer quelle musique vous souhaitez exécuter.

Volume

Indique le niveau sonore de la musique.

2. Les propriétés du composant Sound (Son)

MinimumInterval (ms) (Intervalle Minimum)

Il s'agit d'un écart en millisecondes que vous souhaitez avoir entre deux sons sur le même lecteur, par exemple si vous déclenchez deux fois la même fonction Play.

3. Les propriétés du composant SoundRecorder (Enregistreur_son)

SavedRecording

Vous permet d'indiquer le chemin d'enregistrement du fichier. Si vous ne mettez rien, vous laissez la main à App Inventor 2 pour l'enregistrer sur le chemin de son choix.

4. Les propriétés du composant TextToSpeech (Texte à parole)

Country (Pays)

Indique la prononciation que vous souhaitez avoir pour ce texte.

Language (Langue)

Indique l'accent que vous souhaitez obtenir quand ce texte sera lu.

Pitch (Vitesse parole)

Indique le débit de parole.

Catégorie Drawing and Animation

1. Les propriétés du composant Ball (Balle)

Orientation

La direction dans laquelle va aller la balle.

Intervalle

Temps d'interaction de la balle.

Rayon

Relatif au rayon de la balle.

Vitesse

Vitesse de la balle.

X

Les coordonnées abscisses de la balle.

Y

Les coordonnées ordonnées de la balle.

Z

La situation de la balle par rapport aux autres sur le cadre, est-elle au-dessus ? En dessous ?

2. Les propriétés du composant Canvas (cadre)

Linewidth (Largeur ligne)

La largeur des lignes du cadre.

Catégorie Sensors (Capteurs)

1. Les propriétés du composant AccelerometerSensor (Accéléromètre)

MinimumInterval (ms) Intervalle Minimum

L'intervalle minimum en millisecondes entre deux mouvements de l'appareil.

Sensitivity (Sensibilité)

Il s'agit de la sensibilité du capteur, il possède trois niveaux : faible, modéré, fort. Même à un faible niveau vous verrez que votre téléphone réagira déjà beaucoup.

2. Les propriétés du composant BarcodeScanner (Scanneur_code_à_barre)

UseExternalScanner

Si cette case est cochée, App Inventor 2 utilisera un programme de lecture de QR code différent de celui fourni avec la plateforme.

3. Les propriétés du composant Clock (Horloge)

TimerAlwaysFires

Cette option signifie que le composant agira toujours tant qu'il n'est pas désactivé.

TimeInterval (IntervalleChronomètre)

Il s'agit de la fréquence à laquelle vous souhaitez que le chronomètre réagisse.

4. Les propriétés du composant LocationSensor (Capteur_position)

DistanceInterval (Intervalle de distance)

Permet d'indiquer la fréquence en termes de distance à laquelle vous souhaitez mettre à jour ce capteur.

TimeInterval (Intervalle temps)

Permet d'indiquer en termes de temps la fréquence à laquelle vous souhaitez mettre à jour ce capteur.

5. Les propriétés du composant NearField (Champ_proche)

ReadMode (Mode lecture)

Permet d'indiquer si vous souhaitez l'activer ou non.

6. Les propriétés du composant Pedometer

Stop/Arrêt détection Timeout

Il s'agit de la durée à indiquer en millisecondes avant que le podomètre ne s'arrête de comptabiliser les pas.

StrideLength (Longueur de pas)

Permet de définir la longueur attendue en mètres pour chaque pas.

7. Les propriétés du composant ProximitySensor

KeepRunningWhenOnPause

Cette fonction signifie que le capteur sera actif même si le mobile se met en veille.

Catégorie Social

1. Les propriétés du composant PhoneCall (Appel_téléphonique)

PhoneNumber (Numéro téléphone)

Il s'agit du numéro de téléphone qui sera appelé.

2. Les propriétés du composant Texting (SMS)

GoogleVoiceEnabled (Google Voice activé)

Lorsque cette case est cochée, l'utilisateur a directement la possibilité d'envoyer le message vocalement.

Message

Le contenu du SMS.

Receiving Enabled (Réception activée)

Permet d'activer si oui ou non vous souhaitez recevoir des messages. Les valeurs attendues sont des chiffres allant de 1 à 3, où 1 signifie non, 2 signifie uniquement en tâche de fond et 3 signifie que vous souhaitez y accéder de suite.

3. Les propriétés du composant Twitter

ConsumerKey (Clé consommateur)

Il s'agit de la première clé d'identification pour utiliser l'API de Twitter.

ConsumerSecret (Secret consommateur)

Il s'agit de la seconde clé d'identification pour utiliser l'API de Twitter.

Catégorie Storage (Stockage)

1. Les propriétés du composant FusionTablesCentre (ContrôleFusionTables)

ApiKey (Clé Api)

Il s'agit de la clé d'identification qui permet à Google de vous authentifier.

KeyFile (Fichier clé)

Il s'agit d'un fichier fourni par Google qui permet de vous autoriser à dialoguer avec son service API.

Query (Requête)

La requête SQL que vous souhaitez envoyer. Pour plus d'informations sur la manière de les structurer : https://developers.google.com/fusiontables/docs/v2/getting_started.

ServiceAccount Email

Il s'agit de l'e-mail qui a été créé pour vous au moment où vous avez ouvert votre service API chez Google sur console.developers.google.com, celui-ci est différent de votre e-mail de connexion à App Inventor 2.

UseServiceAuthentication (Utiliser service authentification)

Permet d'activer le service d'authentification.

2. Les propriétés du composant TinyWebDB

ServiceURL

L'URL du service associée à l'envoi des données.

Catégorie Connectivity

1. Les propriétés du composant Web

AllowCookies (Accepter cookies)

Permet d'accepter ou non les cookies.

ResponseFileName (Nom fichier réponse)

Le nom du fichier ou la réponse doit être enregistré.

SaveResponse (Enregistrer réponse)

Si la case est cochée, la réponse de la requête web sera enregistrée dans le fichier indiqué ci-dessus.

Url

L'URL pour la requête web.

Catégorie Experimental

1. Les propriétés du composant FirebaseDatabase

FirebaseDB étant un composant expérimental, c'est-à-dire qu'il ne peut être utilisé afin de créer une application professionnelle, vous n'aurez pas besoin de manipuler les propriétés de ce dernier. Les paragraphes ci-dessous ont donc été rédigés à titre d'information uniquement.

FirestoreToken

Vous n'aurez pas besoin de vous en préoccuper, en effet FirebaseDatabase étant une fonction expérimentale, AI 2 crée et gère automatiquement ce jeton pour vous, vous n'avez donc pas besoin de vous créer un compte spécifique :



FirebaseURL

De la même manière, vous pouvez laisser ici l'information **DEFAULT** préremplie.

UseDefault

À décocher si vous souhaitez y indiquer une URL dédiée d'une base créée sur Firebase.

Persist

Lorsque cette propriété est cochée, les valeurs sont stockées même hors ligne et sont envoyées dès qu'une connexion est établie.

ProjectBucket

Il s'agit du nom de votre projet Firebase.

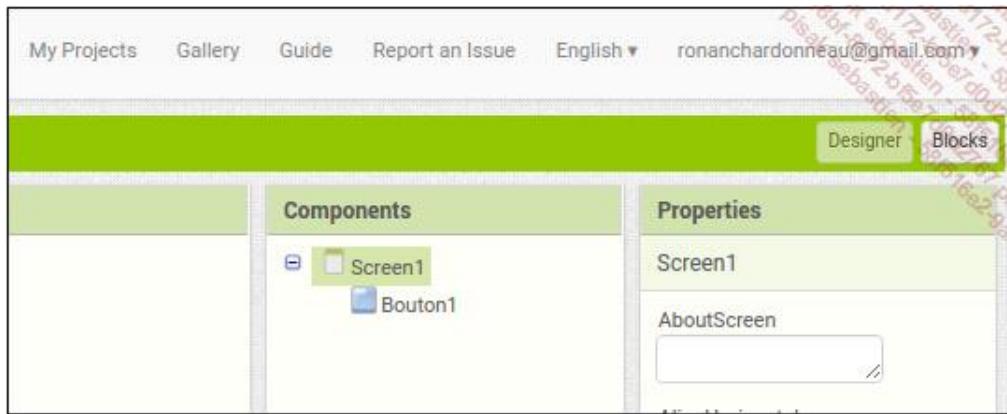
Introduction

Dans ce chapitre, nous entrons dans le cœur d'App Inventor 2. C'est ici que votre application "prendra vie".

Les blocs logiques sont des morceaux qui représentent du code et qui vont vous permettre de déterminer les comportements attendus de la part de votre application. Découvrons sans plus attendre les possibilités qui s'offrent à vous. Nous avons fait en sorte de présenter le plus de fonctionnalités possible avec des applications concrètes, n'hésitez pas à les refaire afin de vous approprier App Inventor 2.

Présentation de l'écran Blocks

- Vous accédez aux blocs logiques en cliquant en haut à droite sur le bouton **Blocks**, juste à côté du bouton **Designer** dans l'interface d'App Inventor 2 :



L'écran des blocs logiques se présente de la manière suivante :

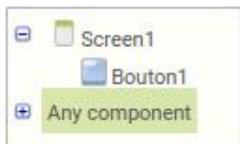


Dans la colonne de gauche se trouve l'ensemble des instructions que vous pouvez utiliser afin de réaliser votre application :

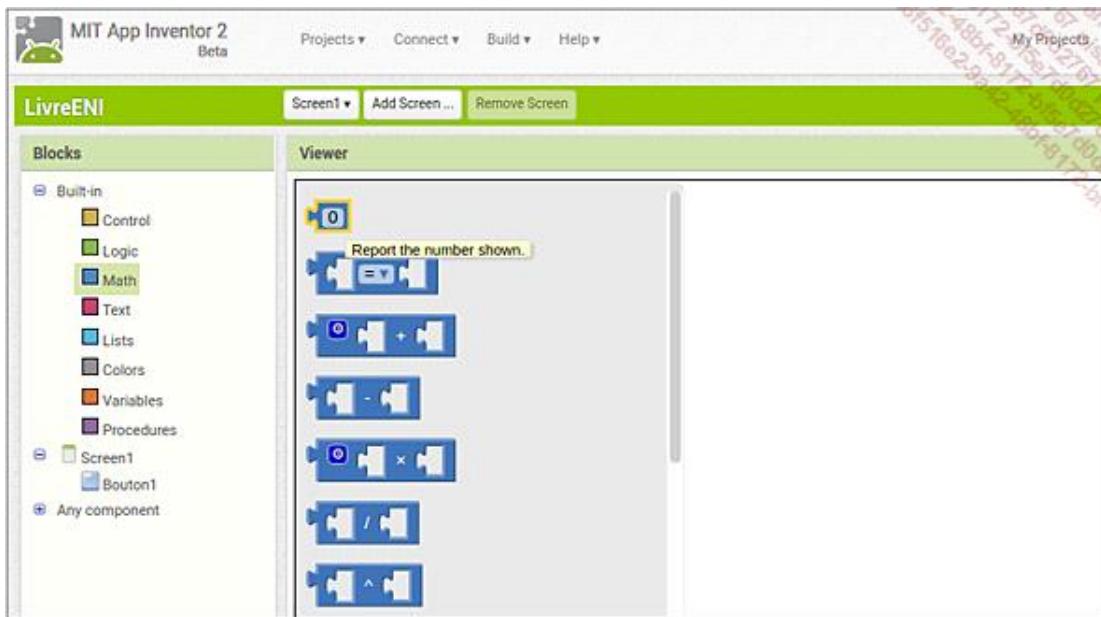


Ces instructions vous permettent de lier les comportements des composants que vous allez ajouter avec l'ensemble de l'application.

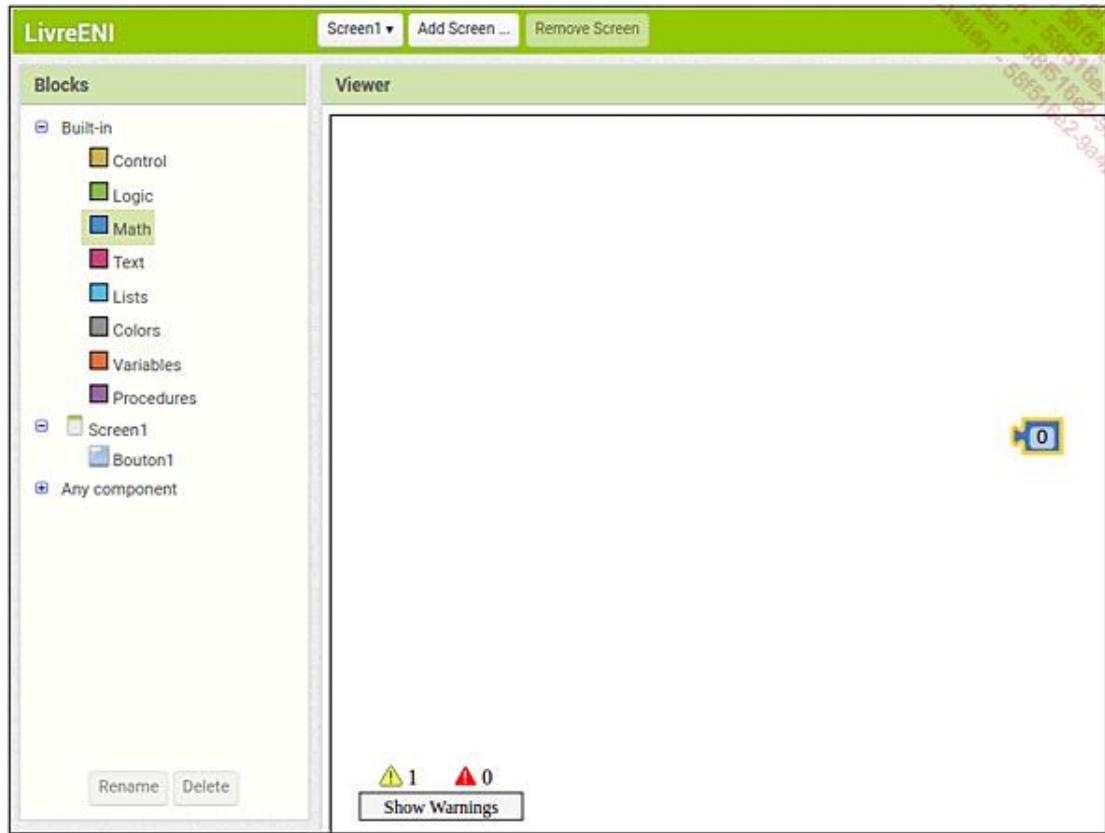
Tous les composants que vous ajoutez dans la partie **Designer** vont apparaître en dessous de ces blocs d'instruction ; dans l'exemple ci-dessous les composants ajoutés sont un écran (toujours présent dans chaque application) et un bouton :



- Lorsque vous souhaitez utiliser un bloc logique, il vous suffit de le sélectionner en cliquant dessus, un nouveau menu va alors apparaître :



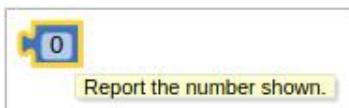
- Choisissez alors le bloc logique de votre choix en fonction du comportement que vous souhaitez que votre application provoque et faites glisser le bloc logique choisi dans l'espace vide à droite du menu :



En cas d'anomalie, c'est-à-dire si votre code n'a pas de logique ou si certains blocs de code entrent en conflit, vous verrez que les panneaux de danger de couleur jaune et rouge seront incrémentés en bas à gauche de votre écran.

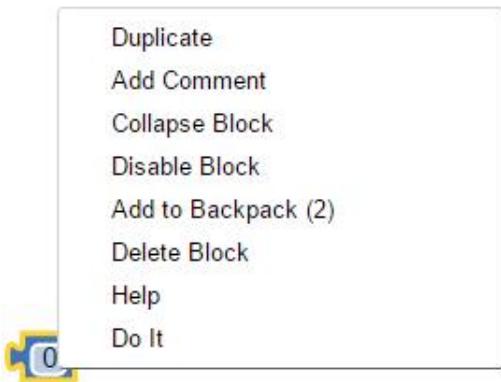
Un avertissement de couleur jaune signifie que votre application sera fonctionnelle mais comportera des bugs, alors que l'avertissement de couleur rouge vous empêchera de pouvoir créer votre application.

- À noter qu'à tout moment vous pouvez connaître le comportement du bloc logique que vous avez ajouté en passant votre souris dessus (le curseur de votre souris prendra alors la forme d'une main), les instructions apparaîtront alors sous la forme d'une info-bulle :



Gérer les blocs

De nombreuses options sont disponibles pour chacun de ces blocs lorsque vous effectuez un clic droit dessus. Découvrons-les maintenant.



Duplicate (Dupliquer)

Cette option effectue un copier-coller de votre bloc. Vous pouvez également effectuer cette manipulation à l'aide de votre clavier. Cette fonction vous sert à aller beaucoup plus vite dans le développement de votre application. Vous découvrirez également plus loin l'option Add to Backpack qui permet également de gagner un temps considérable.

Add Comment (Ajouter un commentaire)

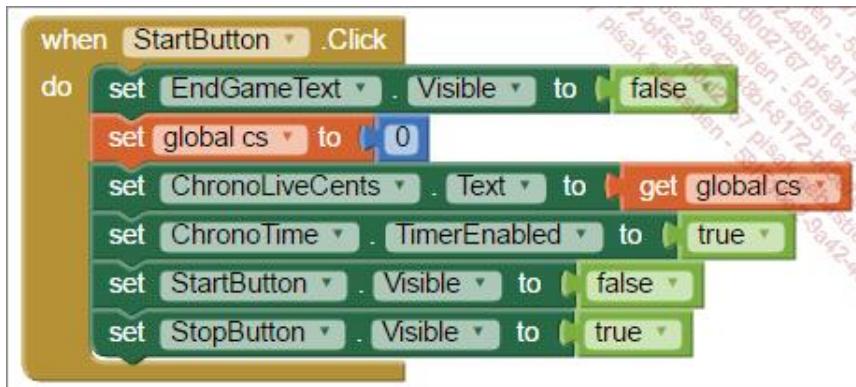
Cette option permet d'ajouter des informations relatives au bloc logique sur lequel vous êtes en train de travailler, cela vous permet de commenter votre code, par exemple :



Il s'agit d'une fonctionnalité très utile quand vous travaillez à plusieurs sur le même projet ou que vous développez une portion de code très complexe. Il vous suffit de cliquer sur le point d'interrogation pour le faire apparaître et disparaître.

Collapse Block (Réduire le bloc)

Cette option permet de minimiser l'affichage d'un groupe de code, cela vous permet de gagner en lisibilité, notamment lorsqu'un comportement est bien connu et ne nécessite plus de modification ; voici un exemple avant réduction :



Et voici un exemple une fois ce même bloc de code réduit (la diminution se fait uniquement au niveau graphique, il ne s'agit pas d'une compression de code).

when StartButton Click do...

À noter que cette opération peut s'effectuer sur n'importe quel bloc de la chaîne.

Disable Block (Désactiver le bloc)

Cette option va rendre le bloc sélectionné non fonctionnel, c'est notamment ce type d'option qui vous permet de voir si un bloc entre en conflit avec un autre. C'est une option de débogage.

Add to Backpack (0)

Le sac à dos, de l'anglais "backpack" :



Il s'agit de l'une des fonctionnalités les plus utiles d'App Inventor 2, elle vous permet de copier-coller des blocs de code entre applications. Par exemple, imaginons que vous découvrez une application qui vous plaît dans la galerie d'AI 2 et que vous décidez de l'installer sur votre compte, vous pourrez reprendre en quelques clics, les bouts de code qui vous intéressent pour ensuite les utiliser directement dans votre application.

Pour ce faire, il vous suffit de cliquer-glisser les bouts de code vous intéressant et de les insérer dans le sac à dos :



Vous n'aurez plus qu'à cliquer dessus à nouveau dans l'application de votre choix pour réutiliser le code désiré.

Delete Block (Supprimer un bloc)

Même finalité que lorsque vous appuyez sur la touche **[Suppr]** après avoir cliqué sur le ou les blocs de votre choix. À noter qu'il n'est pas possible d'annuler une suppression.

Help (Aide)

Cette option vous renverra directement à l'aide en ligne.

Do It (Faire)

Cette option va directement provoquer l'enchaînement des blocs logiques sélectionnés. Pour que cela soit effectif, il faut que l'émulateur ou votre mobile soit branché à votre programme App Inventor 2. Il s'agit donc également d'une fonction de débogage.

→ De la même manière lorsque vous effectuez un clic droit non pas sur un bloc mais sur le fond blanc de l'écran **Blocks**, les options suivantes vous seront proposées :

- **Download Blocks as Image**
- **Collapse Blocks (Réduire les blocs)**
- **Expand Blocks (Développer les blocs)**
- **Arrange Blocks Horizontally (Arranger les blocs horizontalement)**
- **Arrange Blocks Vertically (Arranger les blocs verticalement)**
- **Sort Blocks by Category (Trier les blocs par catégories)**
- **Paste All Blocks from Backpack**
- **Copy All Blocks to Backpack**
- **Empty the Backpack**
- **Help (Aide)**

Download Blocks as Image

Cette option vous permet de télécharger l'ensemble des blocs que vous avez dans votre application pour la transformer en image au format .png. Cette option est notamment utile lorsque vous souhaitez libérer votre code et

l'inclure pour des tierces personnes dans des tutoriels de blogs.

Collapse Blocks (Réduire les blocs)

Cette option va réduire tous les enchaînements de blocs logiques que vous avez créés et donc optimiser l'affichage.

Expand Blocks (Développer les blocs)

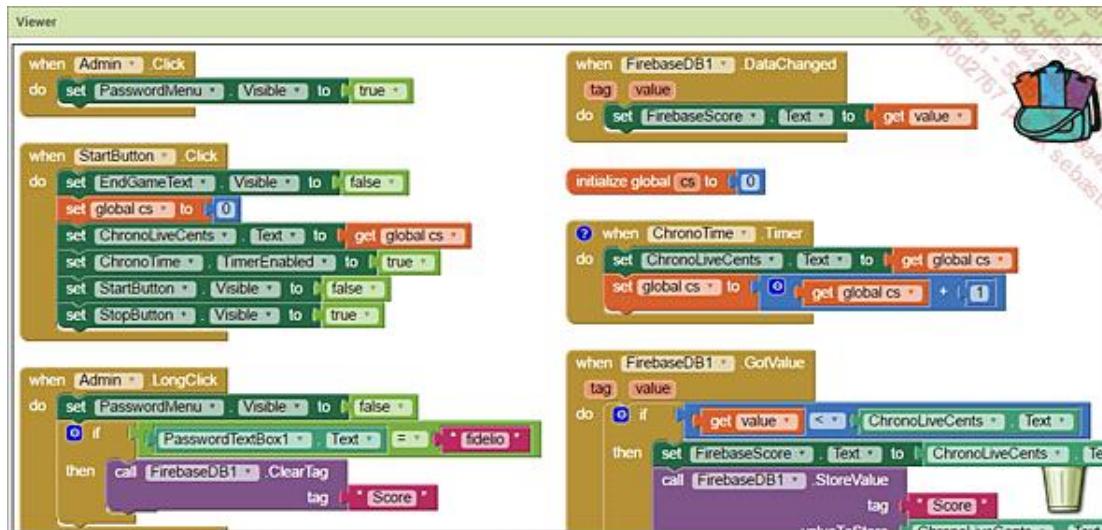
Vous permet d'annuler l'option précédente.

Arrange Blocks Horizontally (Arranger les blocs horizontalement)

Cette option va vous permettre de ranger l'ensemble des blocs logiques que vous avez affiché à l'écran et les classer dans un ordre horizontal. Vous utiliserez cette option lorsque vous souhaiterez identifier tous les blocs présents.

Arrange Blocks Vertically (Arranger les blocs verticalement)

Même option que précédemment mais dans l'autre sens :



Sort Blocks by Category (Trier les blocs par catégories)

Cette option va permettre de regrouper les blocs par catégories, par exemple, les variables avec les variables.

Paste All Blocks from Backpack

Vous permet de coller tous les éléments inclus dans le sac à dos dans votre application courante.

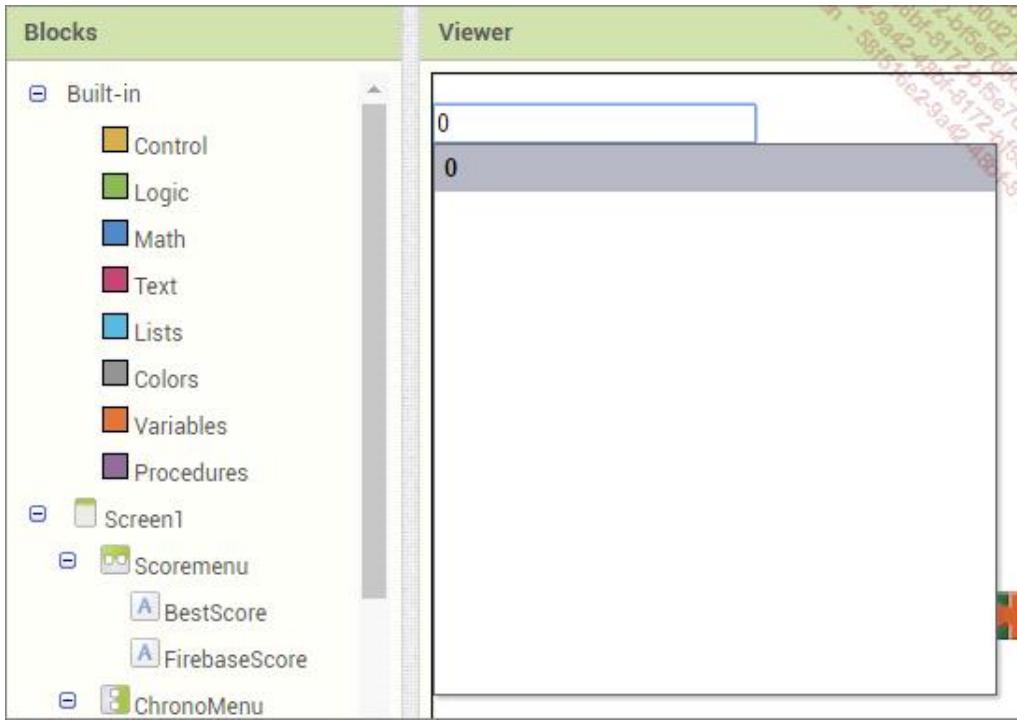
Copy All Blocks to Backpack

Vous permet de copier tous les blocs de votre application dans le sac à dos.

Empty the Backpack

Vous permet de vider le contenu du sac à dos.

Autre astuce à connaître : certains blocs sont souvent amenés à être plus utilisés que d'autres ; afin de vous éviter de toujours cliquer dans la colonne de gauche, vous pouvez directement entrer le nom du bloc recherché :



Dans l'exemple ci-dessus, un simple appui sur la touche **0** du clavier de l'ordinateur permet d'obtenir le bloc logique **0**.

Nous connaissons désormais les fonctionnalités de contrôle offertes par la partie **Blocks**, voyons plus en détail les différents blocs que nous pouvons utiliser pour construire notre application.

Les différentes catégories de blocs logiques

Avant de démarrer, il vous faudra appréhender quelques notions de vocabulaire. Tout d'abord il y a dans App Inventor 2, ce que l'on appelle des événements, "event" en anglais.

Les événements sont symbolisés par des blocs de couleur jaune. Ils vont exécuter ce que l'on appelle des commandes qui sont symbolisées par d'autres couleurs : violette (les méthodes également appelées procédures), verte (les propriétés), orange (les variables).

Les commandes peuvent être accompagnées de valeurs ou de paramètres, par exemple la valeur 1, 2, 3... et pour les paramètres : vrai, faux.

À noter que les événements exécutent les commandes de haut en bas.

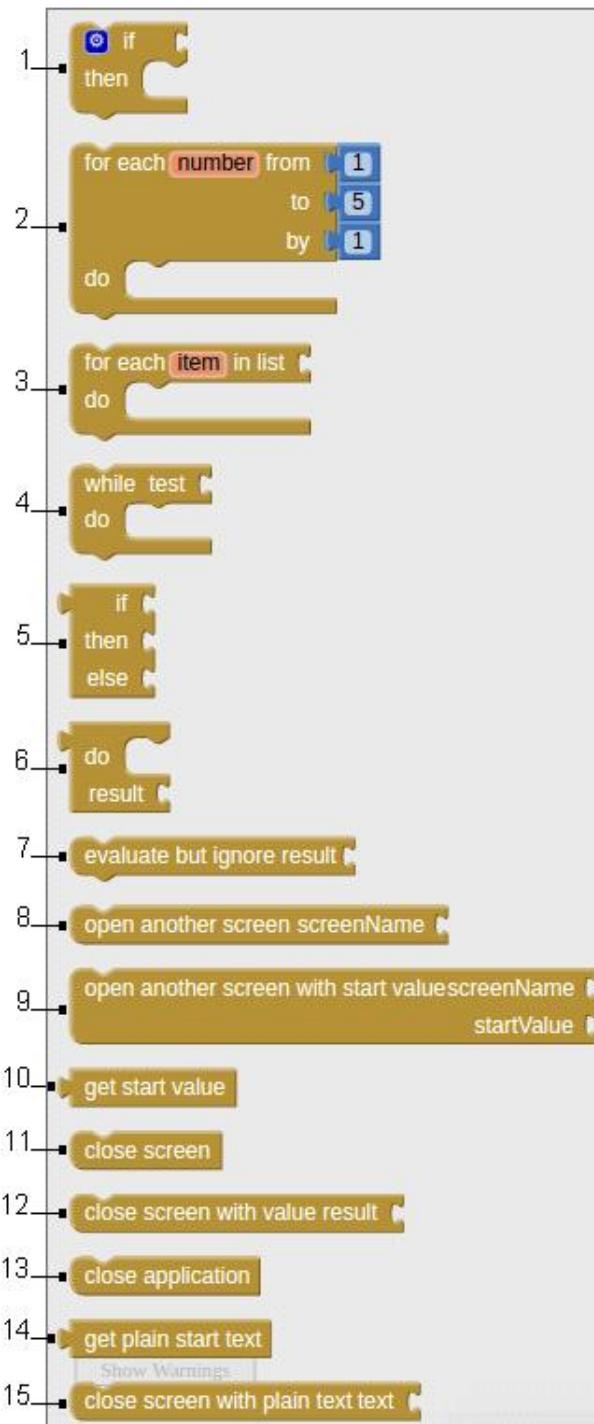
 Les exemples étant légion pour ces blocs logiques, les explications qui suivent sont directement tirées de la documentation officielle d'App Inventor 2.

Les premiers blocs logiques que vous rencontrerez n'appartiennent pas à des composants mais peuvent être utilisés pour effectuer des liaisons entre ces derniers. Vous les trouverez directement dans la colonne de gauche sur l'écran **Blocks** :



1. Control (Contrôle)

→ Pour accéder à cet ensemble de blocs, il vous suffit de cliquer sur **Control**.



Dans les descriptions ci-après, nous avons choisi de traduire les instructions en français.

1. si alors

Si la valeur est vraie, alors faire quelque chose.
 2. pour chaque nombre de 1 à 5 par pas de 1 faire

Exécute les blocs dans la section "do" pour chaque valeur dans l'intervalle du début à la fin, en incrémentant la valeur à chaque fois. Utilise le nom de la variable donnée pour se référer à la valeur actuelle.
 3. pour chaque élément dans la liste, faire

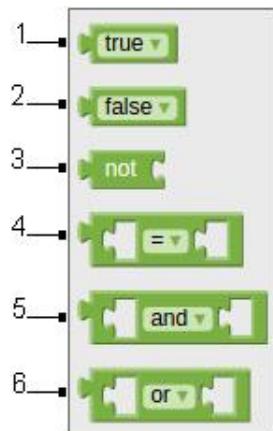
Exécute les blocs dans la section "do" pour chaque élément de la liste. Le bloc utilise alors le nom de la variable donnée pour faire référence à l'élément de la liste courante.
 4. tant que test faire

Exécute le bloc dans la partie "do" tant que le test est vrai.

5. si alors sinon
Si la condition testée est vraie, elle retourne le résultat de l'évaluation de l'expression attachée à la section "then" ; sinon elle retourne le résultat de l'évaluation de l'expression attachée à la section "else" ; au maximum, le résultat de l'une des deux sections va être évalué.
6. faire résultat
Exécute les blocs dans la section "do" et retourne un résultat. Ce bloc est pratique si vous avez besoin d'une procédure avant de retourner une valeur à une variable.
7. évaluer mais ignorer résultat
Exécute les blocs connectés et ignore la valeur renvoyée (s'il y en a). Ce bloc est pratique si vous avez besoin d'appeler une procédure qui retourne une valeur sans en avoir besoin.
8. ouvrir un autre écran Nom écran
Ouvre un nouvel écran dans une application à écrans multiples.
9. ouvrir un autre écran avec une valeur de départ
Ouvre un nouvel écran dans une application multiécran et assigne la valeur de départ à cet écran.
10. obtenir valeur de départ
Retourne la valeur qui a été assignée à cet écran quand il a été ouvert, typiquement par un autre écran dans une application à plusieurs écrans. Si aucune valeur n'a été assignée, retourne un texte vide.
11. fermer écran
Ferme l'écran actuel.
12. fermer écran avec valeur résultat
Ferme l'écran actuel et retourne un résultat à l'écran qui a ouvert celui-ci.
13. fermer application
Ferme tous les écrans dans cette application puis l'arrête.
14. obtenir texte brut de début
Retourne le texte brut qui a été assigné à cet écran quand il a été lancé par une autre application. Si aucune valeur n'a été assignée, retourne un texte vide. Pour les applications multiécrans, utilisez le bloc Obtenir valeur de départ au lieu d'Obtenir texte brut de départ.
15. fermer écran avec texte brut
Ferme l'écran actuel et retourne un texte à l'application qui l'a ouvert. Cette commande permet de retourner du texte aux activités non réalisées par App Inventor et non aux écrans App Inventor. Pour les écrans App Inventor, comme dans les applications multiécrans, utilisez Fermer écran avec valeur et non Fermer écran avec texte brut.

2. Logic (Logique)

Les blocs logiques permettent de définir des conditions :



1. vrai

Retourne la valeur booléenne vrai.

2. faux

Retourne la valeur booléenne faux.

3. pas

Retourne vrai si l'entrée est fausse. Retourne faux si l'entrée est vraie.

4. =

Teste si deux objets sont égaux. Les objets à comparer peuvent être n'importe quoi, pas seulement des nombres. Les nombres sont considérés égaux à leur format de chaîne de caractères, par exemple, le numéro 0 est égal au texte **0**.

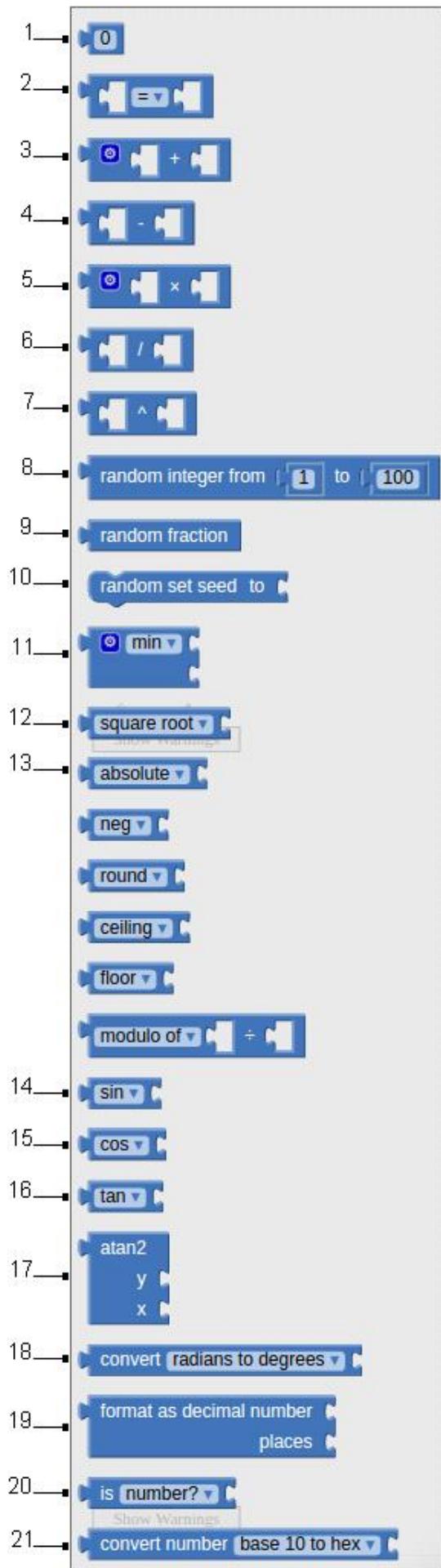
5. et

Retourne vrai si toutes les entrées sont égales.

6. ou

3. Math

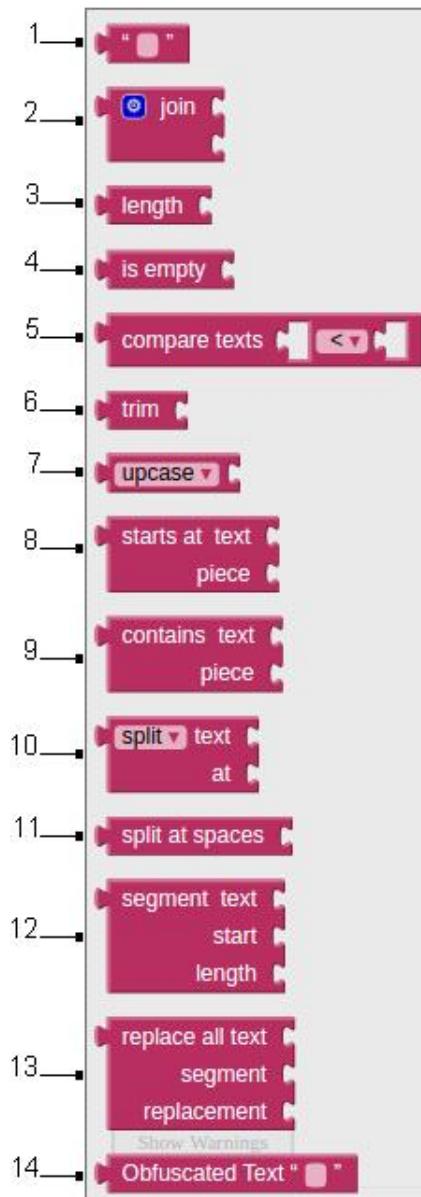
Ces blocs permettent de faire du calcul :



1. 0
Signale le numéro indiqué.
2. =
Retourne vrai si les deux nombres sont égaux.
3. +
Retourne la somme des deux nombres.
4. -
Retourne la différence entre les deux nombres.
5. x
Renvoie le produit des deux nombres.
6. /
Renvoie le quotient des deux nombres.
7. ^
Retourne le premier nombre à la puissance du second nombre.
8. entier aléatoire en 1 et 100
Retourne un nombre aléatoire compris entre la limite supérieure et la limite intérieure. Les limites seront coupées pour être plus petites que $2^{**}30$.
9. fraction aléatoire
Renvoie un nombre aléatoire entre 0 et 1.
10. génère des ensembles de nombres aléatoires répétables à
Spécifie la valeur de référence pour le générateur de nombres aléatoires répétables.
11. min
Retourne le plus petit des arguments proposés.
12. racine carrée
Renvoie la racine carrée d'un nombre.
13. absolue
Renvoie la valeur absolue du nombre spécifié.
14. sin
Fournit le sinus de l'angle donné en degrés.
15. cos
Fournit le cosinus de l'angle donné en degrés.
16. tan
Renvoie la tangente de l'angle en degrés.
17. atan2 y x
Fournit l'angle dans l'intervalle (-180, 180) degrés avec les coordonnées rectangulaires données.
18. convertir radians à degrés
Renvoie la valeur de degré dans l'intervalle n[0,360] correspondant à son argument en radians.
19. format décimal nombre places
Retourne le nombre au format décimal avec un nombre spécifique de places.
20. est un nombre?
Teste si quelque chose est un nombre.
21. Convertir le nombre base 10 en hexadécimal.
Prend le nombre en base 10 et retourne une valeur hexadécimale.

4. Text (Texte)

Ces blocs permettent de gérer du contenu textuel :



1. " "

Une chaîne de caractères.
2. joint

Joint toutes les entrées pour former une seule chaîne de caractères. S'il n'y a aucune entrée, crée un texte vide.
3. longueur

Retourne le nombre de caractères (espaces inclus) dans le texte donné.
4. est vide

Retourne vrai si la taille du texte est 0, sinon retourne faux.
5. comparer textes <

Teste si text1 est inférieur à text2 (d'une façon lexicographique). Si un texte est le préfixe de l'autre, le texte le plus court est considéré comme plus petit. Les caractères en majuscules précèdent les caractères en minuscules.
6. couper

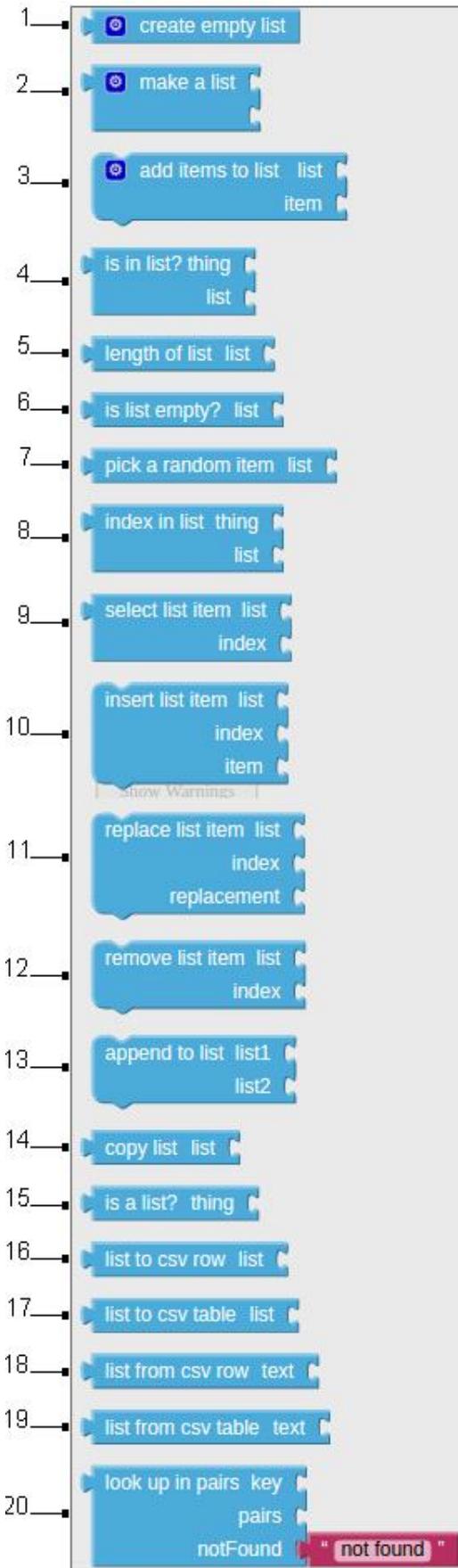
Retourne une copie de ses arguments textes avec la suppression des espaces de début et de fin.
7. majuscule

Retourne une copie de sa chaîne de caractères en argument convertie en majuscule.

8. commencer a texte sous-chaîne
Retourne l'index de début de la sous-chaîne dans le texte où index 1 indique le début du texte. Retourne 0 si la sous-chaîne n'est pas dans le texte.
9. contient texte sous-chaîne
Teste si la sous-chaîne se trouve dans le texte.
10. diviser texte à
Divise le texte à l'endroit indiqué.
11. diviser aux espaces
Divise le texte en sous-chaînes séparées par des espaces.
12. segment texte début taille
Extrait le segment de la taille donnée à partir du texte donné en commençant du texte donné à partir de la position donnée. Position 1 donne le début du texte.
13. remplacer tout texte segment remplaçant
Retourne un nouveau texte obtenu par le remplacement de toutes les occurrences du segment avec le remplaçant.
14. texte masqué ""
Produit un texte.

5. Lists (Listes)

Ces blocs permettent de gérer des listes :



1. créer une liste vide

Crée une liste vide.

2. créer une liste

Crée une liste avec n'importe quel nombre d'éléments que vous ajoutez.

3. ajouter éléments à la liste liste élément

Ajoute des éléments à la fin d'une liste.

4. est dans la liste? objet liste ?

Retourne vrai si l'objet est un élément dans la liste et faux dans le cas contraire.

5. taille de la liste liste

Compte le nombre d'éléments dans la liste.

6. est une liste vide ? liste

Retourne vrai si la liste est vide.

7. choisir un élément aléatoire liste

Choisit un élément aléatoirement dans la liste.

8. index dans la liste objet liste

Trouve la position de l'objet dans la liste. S'il n'est pas dans la liste, retourne 0.

9. choisir liste élément liste index

Retourne l'élément à la position index dans la liste.

10. insérer liste élément liste index élément

Insère un élément dans une liste dans la position spécifiée.

11. remplacer élément de liste liste index remplacement

Remplace un élément dans la liste à l'emplacement spécifié.

12. supprimer liste élément liste index

Supprime l'élément à la position spécifiée dans la liste.

13. ajouter à la liste liste 1 liste 2

Ajoute tous les éléments dans liste2 à la fin de liste1. Après l'ajout, liste1 va contenir ces éléments additionnels, par contre liste2 reste inchangée.

14. copier liste liste

Réalise une copie de la liste, y compris toutes les sous-listes.

15. est une liste? objet

Teste si quelque chose est une liste.

16. liste vers ligne csv liste

Interprète la liste comme une ligne d'une table et retourne un texte CSV (valeur séparée par des virgules) représentant la ligne. Chaque élément dans la liste de la ligne est considéré comme un champ, et est coté avec guillemets dans le texte CSV résultant. Les éléments sont séparés par des virgules. Le texte de la ligne retournée n'a pas de séparateur de ligne à la fin.

17. liste vers table csv liste

Interprète la liste comme un tableau en format ligne-major (row-major) et renvoie un texte CSV (valeur séparée par des virgules) représentant la table. Chaque élément de la liste elle-même doit être une liste représentant une ligne de la table CSV. Chaque élément dans la ligne de la liste est considéré comme un champ et est coté avec guillemets dans le texte CSV résultant. Dans le texte renvoyé, les éléments en ligne sont séparés par des virgules et les lignes sont séparées par CRLF().

18. liste de ligne csv texte

Analyse un texte comme une ligne au format CSV (valeurs séparées par des virgules) pour produire une liste de champs. Si la ligne de texte contient des sauts de ligne à l'intérieur de champs, ceci est considéré comme erreur (plusieurs lignes). Il est accepté pour la ligne de texte de finir en une seule nouvelle ligne ou CRLF.

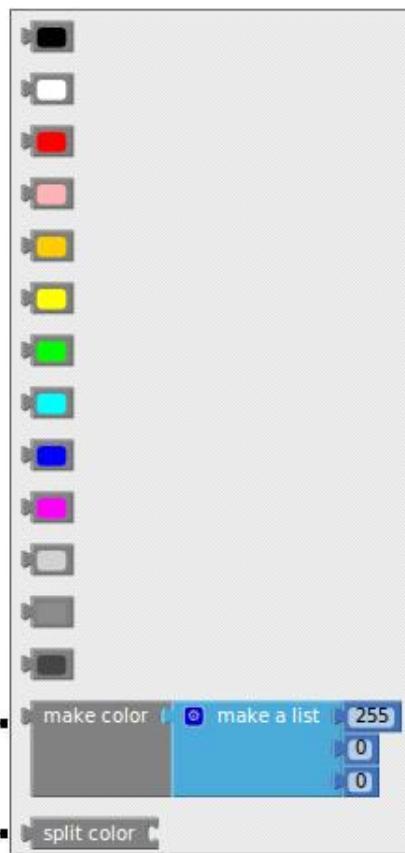
19. liste de table csv texte

Analyse du texte comme un tableau formaté au format CSV pour produire une liste de lignes, chacune d'elles est une liste de champs.

20. recherche dans les paires clé paires nonTrové

Retourne la valeur associée à la clé dans la liste des paires. Ce bloc permet de faire une recherche dans une liste où la clé représente l'élément recherché. Chaque clé possède une valeur, c'est pourquoi on parle de paires.

6. Colors (Couleurs)



1. créer couleur

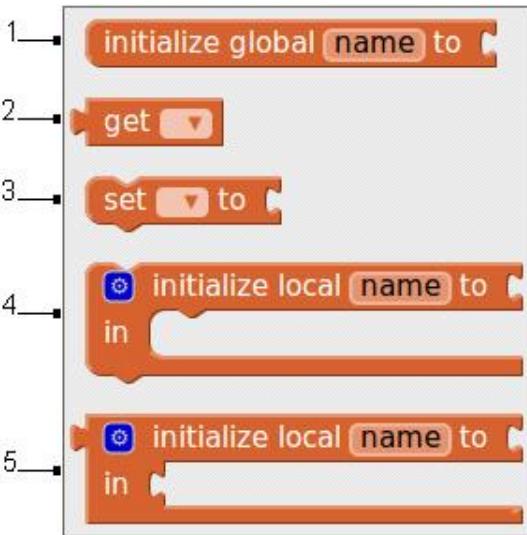
Ce bloc permet de créer la couleur de votre choix. Les trois éléments par défaut représentent les trois couleurs : rouge, vert et bleu. Chacun peut prendre une valeur entre 0 et 255

2. diviser couleur

Ce bloc de code permet de renvoyer les quatre valeurs de couleurs que représentent l'élément indiqué. Par exemple, en utilisant ce bloc vous pourrez connaître les couleurs du composant que vous avez sélectionné dans votre application.

7. Variables

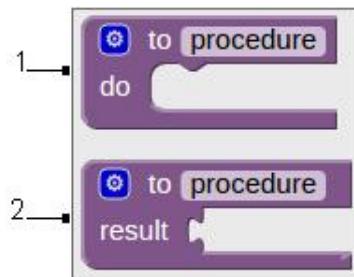
Comme leur nom l'indique, les variables sont des valeurs que vous définissez et qui vont évoluer en fonction d'un comportement donné.



1. initialiser global nom à
Crée une variable et lui donne la valeur du bloc attaché.
2. obtenir
Retourne la valeur de cette variable.
3. mettre à
Assigne à cette variable la valeur saisie.
4. initialiser local nom à dans
Permet de créer des variables qui sont uniquement accessibles dans la section do de ce bloc.
5. initialiser local nom à dans
Permet de créer des variables qui sont accessibles uniquement dans la section retour de ce bloc. Ce bloc et le précédent ont le même nom mais leur structure est différente. Dans un cas, il y aura une exécution, dans l'autre une inclusion.

8. Procédures

Ces blocs logiques permettent de définir des fonctions que vous allez pouvoir réutiliser par la suite. On les utilise pour exploiter un bloc de code à de nombreuses reprises au sein d'une application.



1. à procédure faire
Fait appel à une procédure qui ne retourne pas de valeur.
2. à résultat
Fait appel à une procédure qui retourne une valeur résultat.

Les fonctions des composants User interface

Nous venons de voir la présentation des blocs logiques classiques, voyons désormais l'utilisation des blocs pour chaque composant de l'interface utilisateur. Il est possible qu'à la date à laquelle vous lirez cet ouvrage de nouveaux composants aient été ajoutés ; à titre indicatif, voici le lien vers la documentation officielle : <http://ai2.appinventor.mit.edu/reference/components/userinterface.html#buttonproperties>

Vous trouverez ci-dessous la liste des différentes fonctions que vous pouvez utiliser pour chacun des composants de l'interface utilisateur ; lorsque cela nous semble nécessaire, nous avons ajouté un exemple concret.

1. Composant Screen (écran)

Événements associés à l'écran

BackPressed()

Cet événement permet de déclencher une suite de commandes lorsque le bouton retour de votre mobile est appuyé. En général, on recourt à ce bloc de commande pour retourner à un écran précédent ou fermer l'application. Un exemple de bloc à construire pourrait être :



Ce code peut par exemple être inséré sur le premier écran d'une application. Ici, lorsque nous appuyons sur le bouton retour de notre mobile, un message sous forme de pop-up apparaît pour nous demander si nous souhaitons quitter l'application. Si nous répondons **Yes** alors l'application se ferme.

ErrorOccurred()

Cet événement est à conseiller pour les utilisateurs avancés, lorsque votre application rencontre une erreur, par exemple un souci de connexion avec une base de données ou avec un autre appareil, vous pouvez créer une suite de commandes qui vous permettra de connaître les caractéristiques de l'erreur rencontrée et ainsi de pouvoir trouver des pistes de résolution.

Initialize()

Probablement l'une des fonctions les plus utilisées pour l'écran. Elle permet d'indiquer les commandes que vous voulez voir exécuter lorsque l'application se lance. Par exemple dans l'application ci-dessous :



Lorsque l'écran est chargé, alors le fond d'écran prend la valeur personnalisée indiquée. En effet, par défaut, l'écran ne peut prendre une couleur personnalisée et est limité à celles proposées de base par App Inventor 2.

`OtherScreenClosed()`

Permet de déclencher un comportement lorsqu'un autre écran se ferme.

`ScreenOrientationChanged()`

Cet événement va lancer une série de commandes lorsque l'orientation de l'écran va changer, c'est-à-dire par exemple passer de l'orientation portrait à paysage.

Méthodes

`HideKeyboard()`

Cette commande va cacher le clavier de votre mobile.

`Button Click()`

Probablement l'événement le plus populaire d'App Inventor, lorsqu'un clic est enregistré sur le bouton alors l'application exécute les commandes indiquées :



Dans l'exemple ci-dessus, lorsque l'utilisateur appuie sur le bouton **About** alors l'application va ouvrir un nouvel écran (d'une certaine façon va rediriger l'utilisateur vers un autre écran) qui, ici, a pour nom **About**.

`GotFocus()`

Cet événement se déclenche lorsqu'une zone est sélectionnée au détriment d'une autre.

`LongClick()`

Cet événement permet de déclencher une série de commandes lorsqu'un clic long est effectué sur le bouton.

Par exemple avec le code suivant :

```

when Manage .Click
do set DisplayMenu .Visible to true

when Manage .LongClick
do set DisplayMenu .Visible to false

```

Lorsque nous cliquons sur un bouton du nom de **Manage** alors un menu apparaît.

Si nous effectuons un clic long sur ce bouton, alors le menu ne s'affiche plus.



Ci-dessus, un exemple lorsqu'on effectue un simple clic, ci-dessous lorsqu'on effectue un clic long :



`LostFocus()`

Cet événement se déclenche lorsque la zone n'est plus sélectionnée.

`TouchDown()`

Cet événement permet d'indiquer ce que vous souhaitez faire lorsque le bouton est appuyé.

`TouchUp()`

Cet événement permet d'indiquer ce que vous souhaitez faire lorsque le bouton est relâché.

2. CheckBox

`Click()`

Cela signifie que l'utilisateur a cliqué sur la case à cocher. Cet événement va donc permettre de cocher et décocher la case.

`GotFocus()`

Cet événement se déclenche lorsque l'élément est sélectionné.

`LostFocus()`

Cet événement se déclenche lorsque l'élément n'est plus sélectionné.

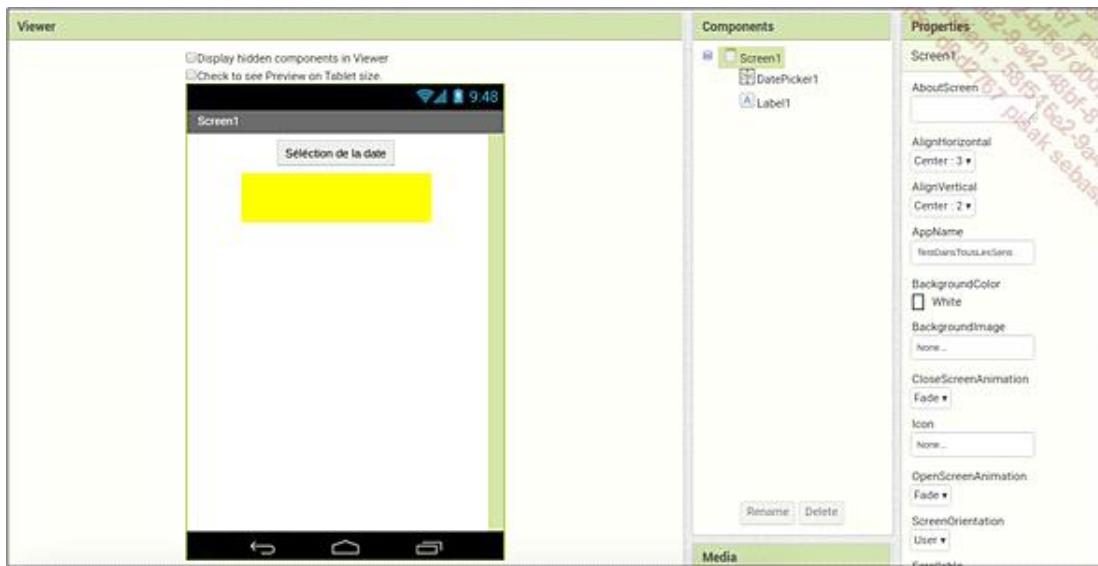
3. DatePicker

Événements

AfterDataSet()

Cet événement va se réaliser lorsque l'utilisateur aura sélectionné la date de son choix dans la boîte de dialogue.

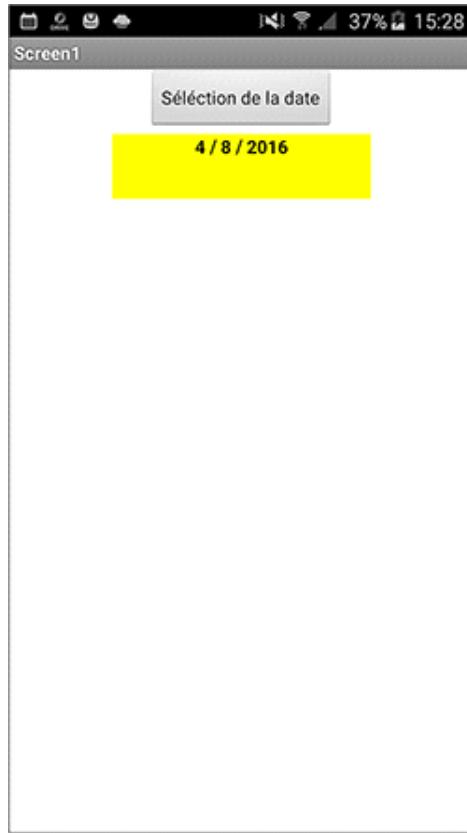
Il est possible, par exemple, d'imaginer une application pour laquelle nous souhaiterions faire afficher la date sélectionnée dans un libellé, par exemple :



Avec un code tel que celui-ci :



Le rectangle jaune prendra alors la valeur de la date sélectionnée :



`GotFocus()`

Cet événement se déclenche lorsque l'élément est sélectionné.

`LostFocus()`

Cet événement se déclenche lorsque l'élément n'est plus sélectionné.

`TouchDown()`

Même finalité que pour les boutons.

`TouchUp()`

Même finalité que pour les boutons.

Méthodes

`LaunchPicker()`

Cette fonction permet de déclencher l'affichage de la pop-up de sélection de date. Par défaut, cette dernière s'effectue lorsqu'on clique sur le bouton. Cependant, vous êtes libre de la déclencher lorsque vous le souhaitez. Par exemple, si vous souhaitez que cette dernière se déclenche dès que l'écran de votre application se lance, rédigez un code similaire à :

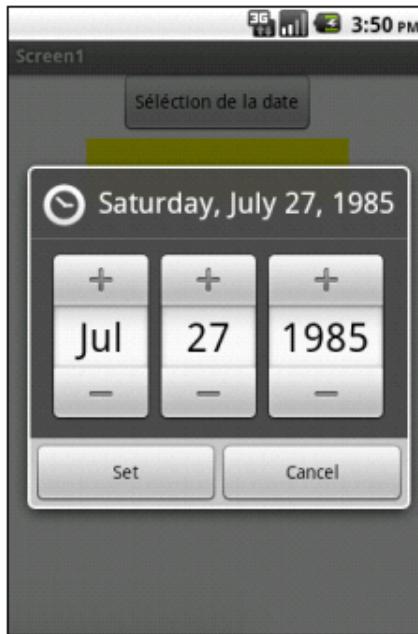
```
when Screen1.Initialize
do call DatePicker1.LaunchPicker
```

```
SetDateToDisplay(number year, number month, number day)
```

Cette fonction vous permet de définir la date à afficher par défaut lorsque la pop-up s'affiche. Par exemple, si nous souhaitons que la date du 27 juillet 1985 soit définie au lancement de votre application, nous pourrions alors rédiger le code suivant :

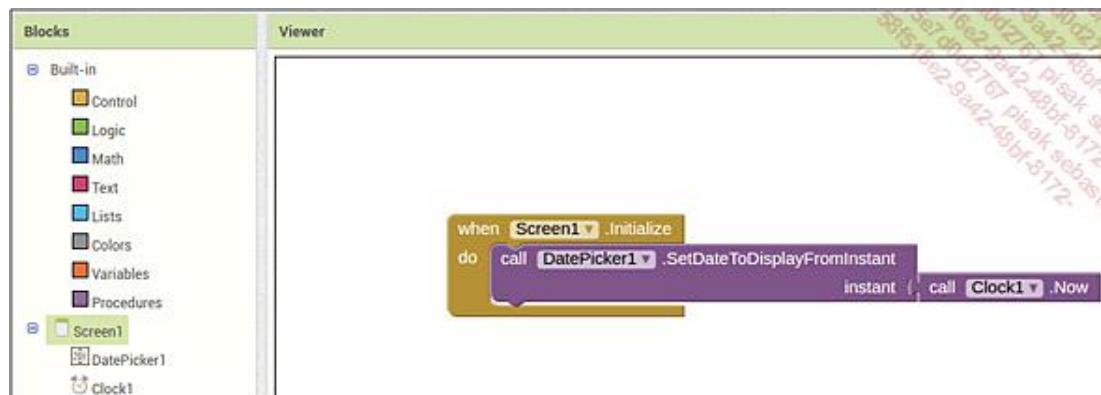


Ce qui signifie que lorsque nous cliquerons sur le bouton déclenchant l'affichage de la pop-up, nous obtiendrons :



```
SetDateToDisplayFromInstant( InstantInTime instant )
```

Cette méthode est similaire à la précédente à la différence que vous pouvez indiquer directement l'instant que vous souhaitez, par exemple :



4. Image

Le composant **Image** ne possède que des propriétés. Découvrons-les ci-dessous :

Animation

Cette propriété, assez peu connue, permet de faire défiler l'image en fonction de la valeur que vous lui donnez, par exemple en écrivant un code similaire à :



Dans notre cas de figure, dans cette application, au démarrage de l'écran l'image va défiler lentement de gauche à droite avant de s'arrêter.

La propriété **Animation** peut prendre également les valeurs suivantes : **ScrollRightSlow**, **ScrollRight**, **ScrollRightFast**, **ScrollLeftSlow**, **ScrollLeft**, **ScrollLeftFast**, **Stop**.

RotationAngle

Cette propriété permet d'incliner une image en fonction de l'angle de rotation que vous indiquez ; dans cet exemple, l'angle de rotation est de 35 degrés :



Ce qui donnera ce résultat :



Les autres propriétés de l'image ont déjà été évoquées pour d'autres composants.

5. Label

Le composant **Label** ne possédant que des propriétés que nous avons déjà vues jusqu'à présent, nous ne détaillerons pas plus les fonctions pour ce composant.

6. ListPicker

Ce composant présentant également des fonctions que nous avons déjà vues, nous ne développerons pas plus ce dernier.

7. ListView

Les fonctionnalités de ce composant ont également déjà été abordées.

8. Notifier

AfterChoosing

Vous permet d'indiquer le comportement que vous souhaitez que votre application déclenche une fois la décision prise sur une alerte vous proposant des choix. Nous présentons un exemple par l'intermédiaire de la méthode ShowChooseDialog.

AfterTextInput

Vous permet de déclencher le comportement que vous souhaitez une fois que la fonction ShowTextDialog que nous verrons dans ce chapitre sera activée.

DismissProgressDialog

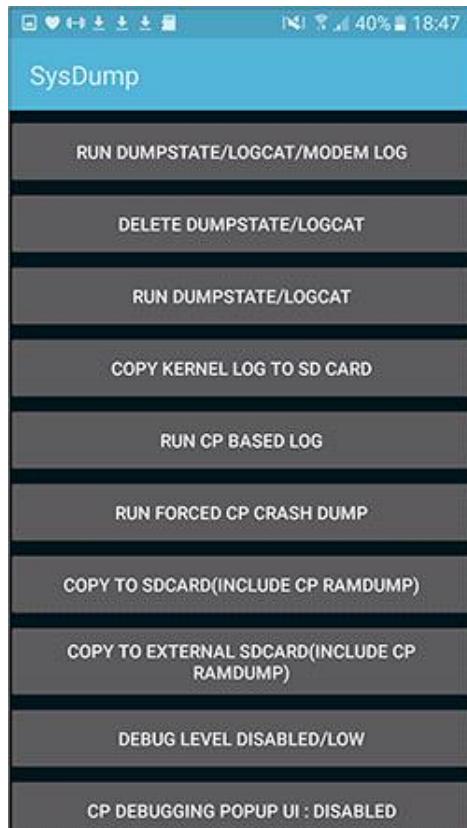
Cette fonctionnalité permet de mettre fin à la fonction ProgressDialog que nous allons voir par la suite.

LogErrorMessage

Les logs sont les messages enregistrés par votre smartphone quand une application tourne. Il s'agit d'informations techniques dont vous n'aurez probablement jamais besoin sauf si vous êtes un vrai développeur.

Admettons que vous êtes un curieux, ce qu'il vous faut savoir c'est que vous pouvez accéder aux logs de votre smartphone. La procédure étant différente pour chacun d'entre eux, nous allons ci-dessous expliquer uniquement celle pour les appareils de la marque Samsung.

- Composez le numéro suivant sur votre téléphone : *#9900#. Vous devriez alors accéder à l'écran suivant :



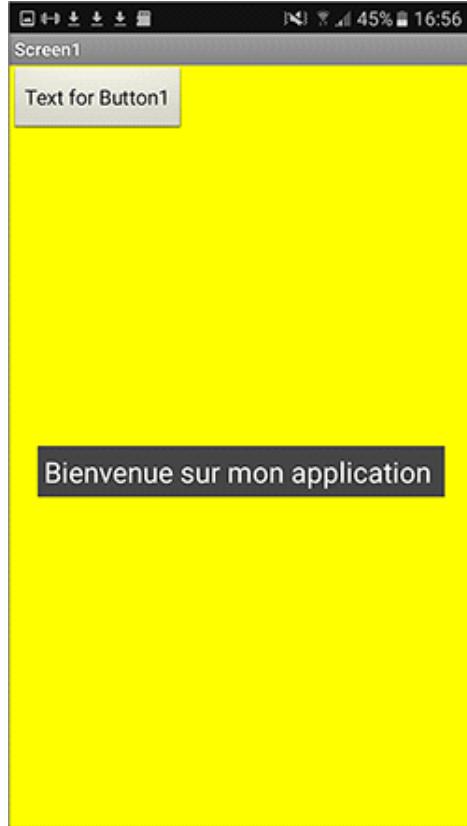
- Cliquez alors sur **RUN DUMPSTATE/LOGCAT**. En principe en cliquant sur cette commande, l'ensemble des logs vont être copiés dans un répertoire de votre téléphone. Dans notre cas de figure il s'agit du répertoire /log/ situé sur notre appareil.

ShowAlert

Cette fonction permet d'afficher une alerte de type pop-up dans votre application. Dans l'exemple ci-dessous, un bouton va déclencher cette pop-up d'alerte :

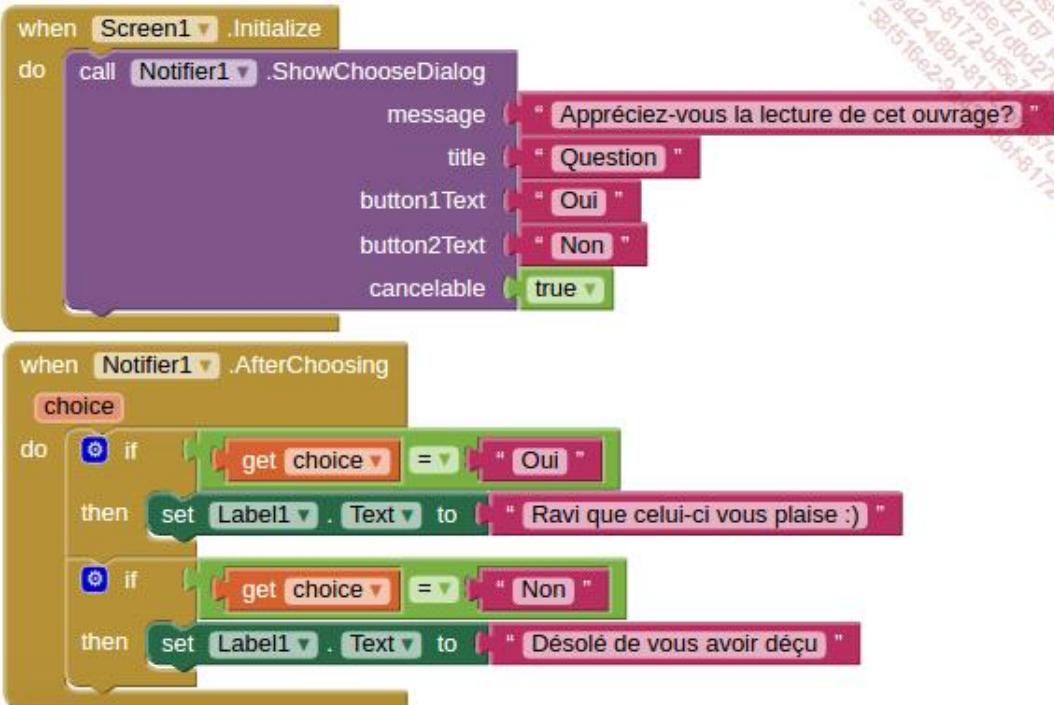


Ce qui donnera, une fois le bouton cliqué :

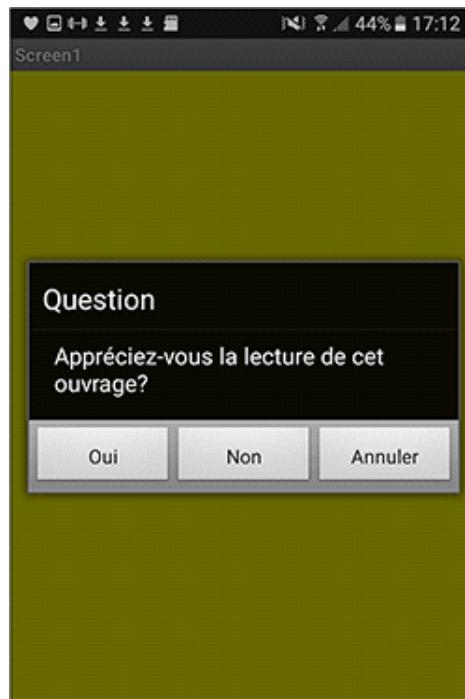


ShowChooseDialog

Cette méthode vous permet de faire apparaître une pop-up dans laquelle vous pourrez faire vos choix. Afin de l'illustrer, nous avons réalisé l'application suivante :



Notre application ressemble donc à :



La réponse que vous donnerez ici permettra soit d'afficher une réponse positive, soit négative, soit rien du tout si vous sélectionnez **Annuler** ; ici nous avons appuyé sur **Non** :

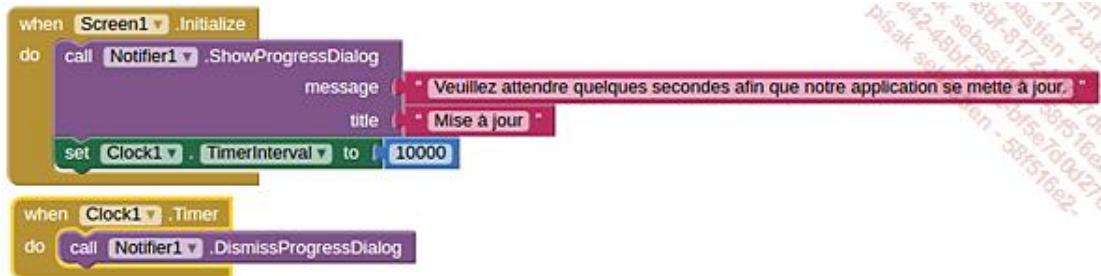


ShowMessageDialog

Cette fonctionnalité vous permet d'afficher une simple pop-up telle que vue précédemment mais qui n'aura qu'un bouton au lieu de trois.

ShowProgressDialog

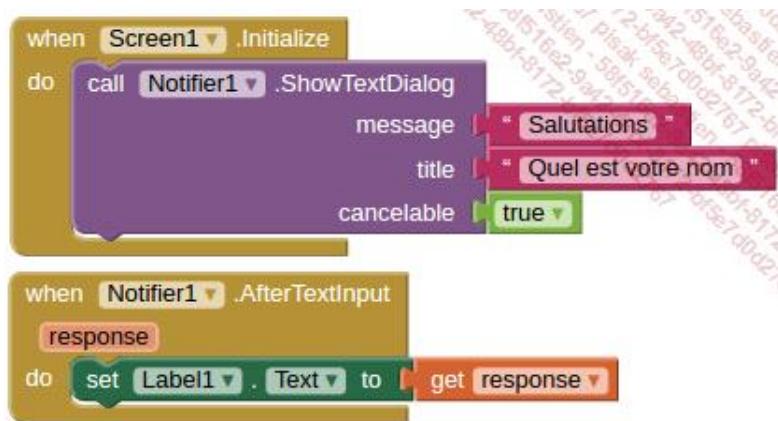
Cette méthode fait apparaître une fenêtre indiquant qu'une action est actuellement en cours et qu'il faut donc attendre. Faites attention, cette fonctionnalité peut bloquer votre application si vous n'avez pas prévu l'utilisation de la fonctionnalité `DismissProgressDialog`. Afin d'illustrer nos propos, nous pouvons imaginer l'application suivante qui va afficher cette fenêtre de progression, et grâce à un composant `Timer` nous allons faire fermer cette fenêtre de dialogue après une période de dix secondes :





ShowTextDialog

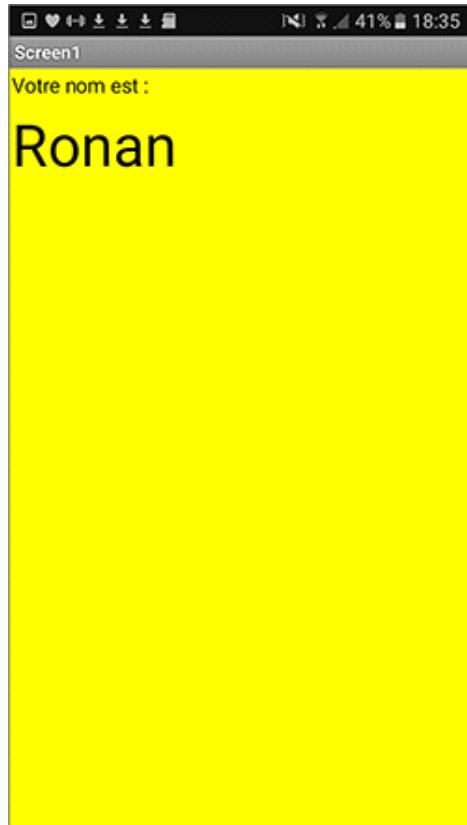
Cette méthode va activer une pop-up vous permettant de saisir du texte. Nous allons utiliser cette fonctionnalité en plus de l'événement AfterTextInput afin d'illustrer nos propos :



Dans cette application, notre nom sera demandé et, une fois le nom donné, celui-ci s'affichera dans un libellé :



Ce qui entraînera le comportement suivant si nous cliquons sur **OK** :



PasswordField

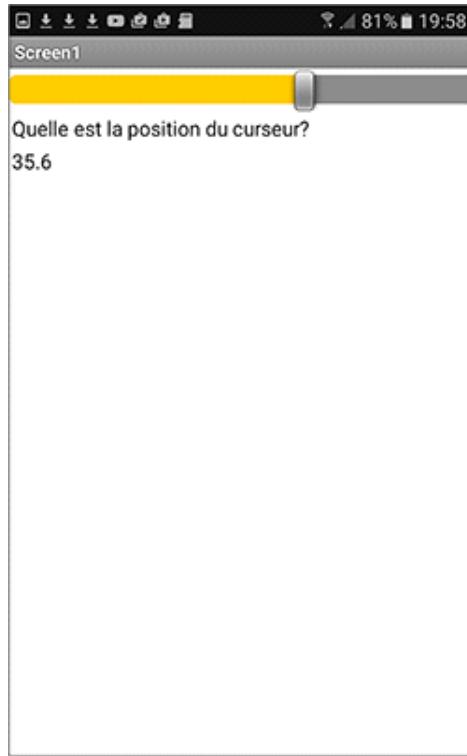
PasswordField n'étant pas très utilisé et possédant les mêmes fonctionnalités que le composant TextBox classique, nous vous invitons à consulter les fonctionnalités de ce composant détaillées juste après dans ce chapitre.

Slider PositionChanged

Cet événement permet d'indiquer que le curseur du slider a changé. Par exemple :



Cette application va nous permettre de connaître la position du curseur :



ColorLeft

Cette propriété vous permet d'indiquer la couleur qui s'affichera à la gauche du curseur.

ColorRight

Cette propriété vous permet d'indiquer la couleur qui s'affichera à la droite du curseur.

MaxValue

Cette propriété permet d'indiquer la valeur maximale que peut prendre le slider.

MinValue

Cette propriété permet d'indiquer la valeur minimale que peut prendre le slider.

ThumbEnabled

Permet d'activer le slider.

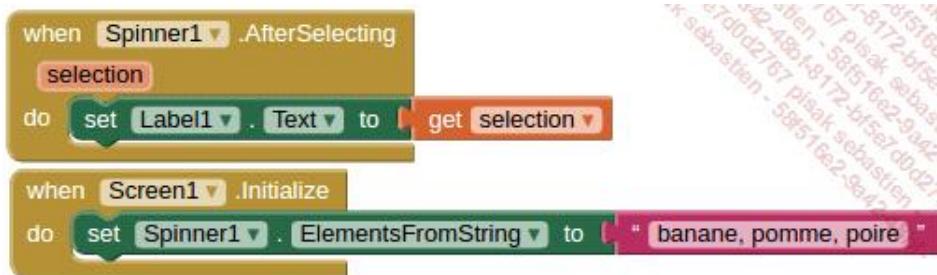
ThumbPosition

Permet d'indiquer la position du curseur.

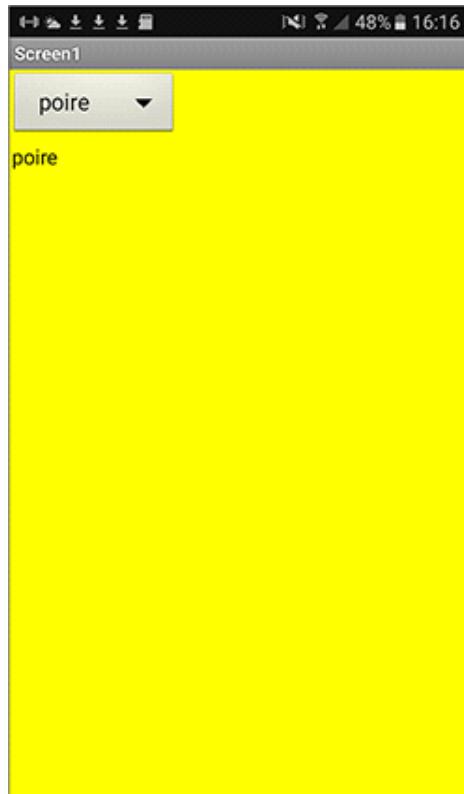
9. Spinner

AfterSelecting

Déclenche un événement permettant de décrire les actions que vous souhaitez entreprendre une fois qu'un élément de la liste a été sélectionné. Afin d'illustrer ce que peut faire cet événement, nous allons réaliser une application qui utilisera la propriété ElementsFromString :



ElementsFromString permet d'indiquer les éléments qui seront présents dans la liste de sélection, chaque élément doit être séparé par une virgule. Ici, lorsque nous cliquons sur le spinner et que celui-ci dévoile la liste déroulante, nous souhaitons que l'élément sélectionné apparaisse dans le composant Label1 :

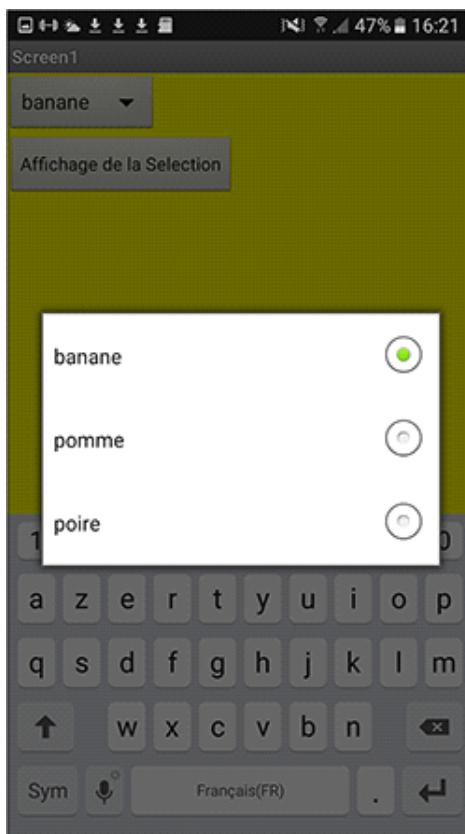


DisplayDropdown

Cette méthode permet de faire apparaître le menu déroulant directement. Par exemple, nous pouvons imaginer une application avec un bouton qui, une fois sélectionné, permettrait de faire apparaître la sélection, par exemple :



Ce qui nous donnerait :



Propriétés

Prompt

Cette propriété vous permet de définir le titre de votre liste de sélection ; par exemple ici, nous avons cherché à donner le nom **Liste de sélection** à notre liste de sélection :



Et nous avons obtenu :



Selection

Permet d'indiquer ce que vous souhaitez voir apparaître par défaut comme élément sélectionné de la liste.

SelectionIndex

Il s'agit de l'équivalent de la propriété précédente à la différence qu'au lieu de faire appel à l'élément sélectionné par son nom, par exemple banane, pomme ou poire, vous allez faire appel à un chiffre qui correspond à l'ordre de l'élément dans l'index ; par exemple, banane a pour index 1, pomme pour index 2 et poire pour index 3.

10. TextBox

GotFocus

Cet événement se déclenche lorsqu'une zone est sélectionnée au détriment d'une autre, c'est par exemple le cas de figure ci-dessous lorsque le composant TextBox numéro 1 est sélectionné par l'utilisateur alors qu'une deuxième TextBox est présente sur l'écran :



LostFocus

Idem que précédemment à la différence que dans ce cas, la TextBox numéro 2 sera sélectionnée au détriment de l'autre.

HideKeyboard

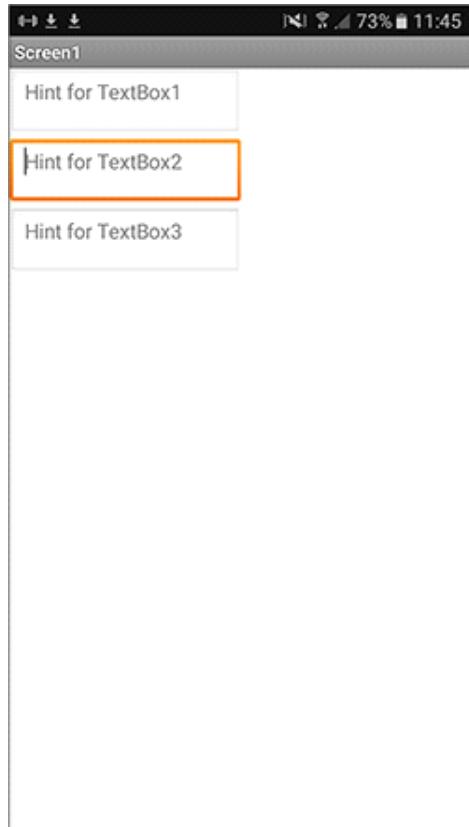
Cette méthode vous permet de cacher le clavier qui s'affichera sur votre écran si vous cliquez sur le champ **TextBox**.

RequestFocus

Cette méthode permet de sélectionner cet élément. Par exemple l'application suivante :



Va automatiquement mettre en avant la TextBox2 sur les trois TextBox de l'application :



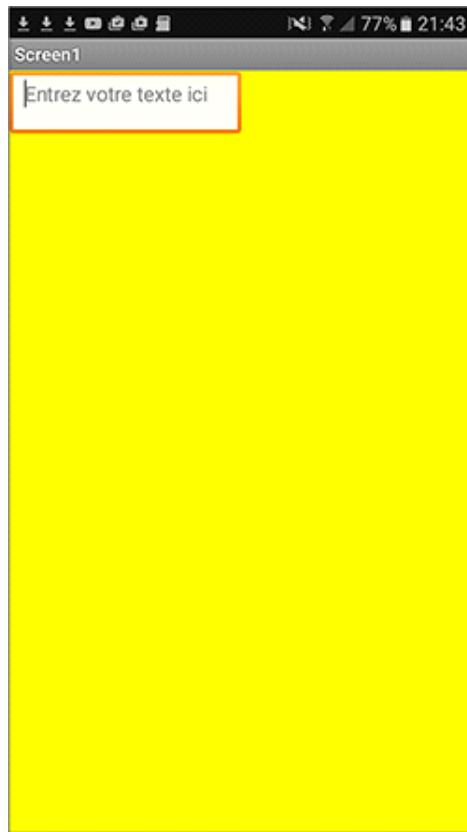
Propriétés

Hint

Permet d'indiquer le texte par défaut qui apparaîtra à l'intérieur du champ de saisie de texte avant que celui-ci ne soit sélectionné, par exemple :



Nous donnera :



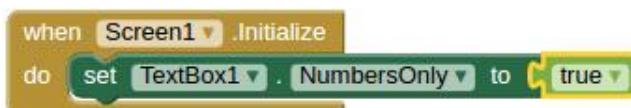
Multiline

Cette propriété, une fois fixée sur true permet de supporter la touche **[Entrée]** lorsque vous écrivez dans un bloc de texte et vous permettra donc d'écrire sur plusieurs lignes.

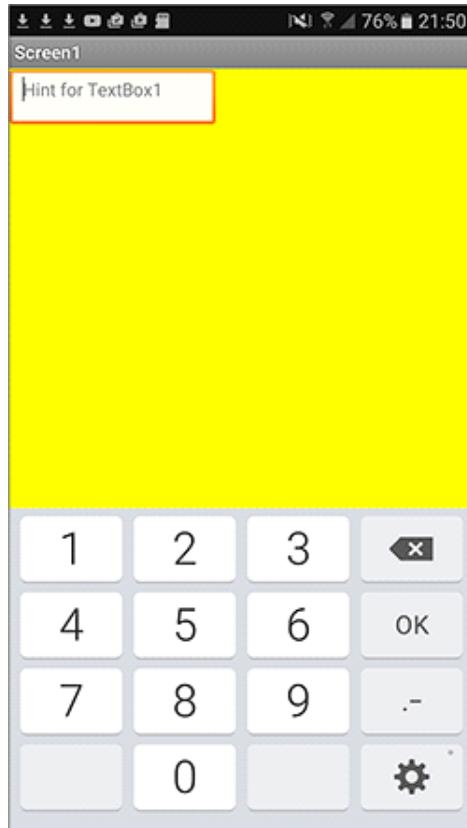
NumbersOnly

Si vous fixez cette propriété sur true alors le champ texte ne supportera que des nombres.

Exemple :



Nous donnera :



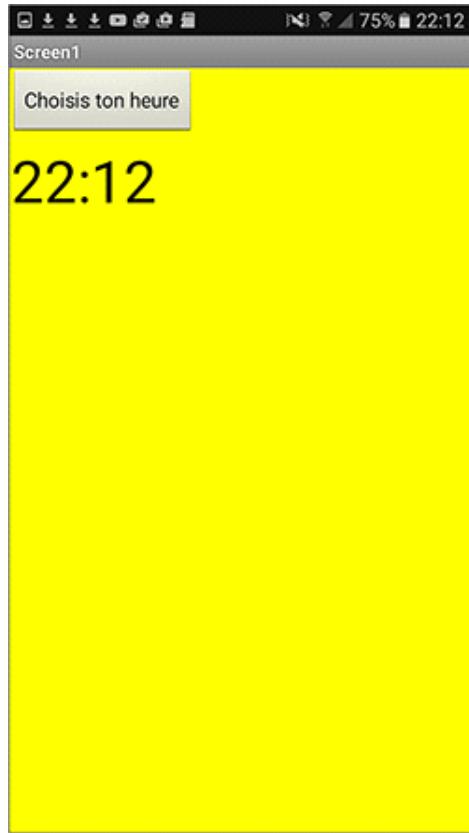
11. TimePicker

AfterTimeSet

Permet d'indiquer l'action que vous souhaitez effectuer une fois l'heure sélectionnée. Par exemple, ci-dessous nous avons réalisé une application permettant d'afficher l'heure dans un champ texte :



Ce qui donnera :



Les événements GotFocus, LostFocus, TouchDown, TouchUp étant très communs entre les différents composants, nous n'allons pas les détailler ici.

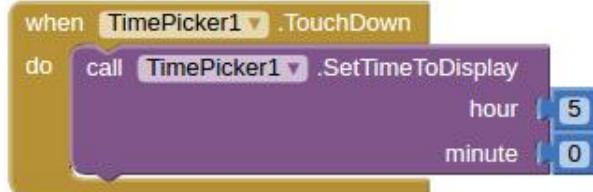
LaunchPicker

Permet de déclencher directement l'affichage de l'horloge.



SetTimeToDisplay

Cette fonction permet d'afficher l'heure sur une heure que vous avez indiquée, par défaut cette heure est celle de votre téléphone au moment où vous appuyez sur le bouton TimePicker. Dans l'exemple ci-dessous, l'heure affichée sera 5 heures du matin :



SetTimeToDisplayFromInstant

Cette fonction est similaire à celle que nous avons vue précédemment pour le composant DatePicker : SetDateToDisplayFromInstant instant.

12. WebViewer

CanGoBack

Cette fonction vous permet de savoir si une page précédente est enregistrée dans l'historique du navigateur.

CanGoForward

Cette fonction vous permet de savoir si une page suivante est enregistrée dans l'historique du navigateur.

ClearCaches

Cette fonction permet de vider le cache du navigateur.

ClearLocations

Cette fonction permet de supprimer les données enregistrées par rapport à la localisation de l'utilisateur.

GoBack

Permet d'accéder à la page web précédente, tout comme avec un navigateur classique.

GoForward

Permet d'accéder à la page web suivante, tout comme avec un navigateur classique.

GoHome

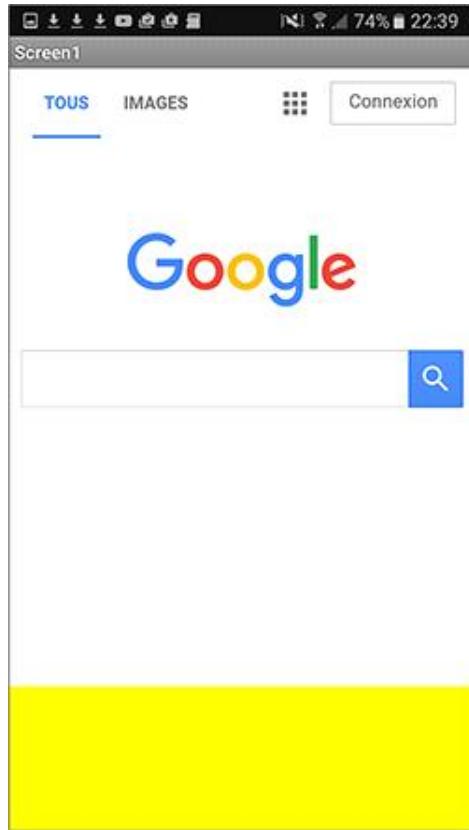
Lorsque cette méthode est déclenchée, le composant WebViewer va retourner à la page d'accueil que vous avez indiquée dans l'application.

GoToUrl

Cette méthode vous permet de vous rendre sur l'URL indiquée dans le composant webviewer ; exemple :



va donner :



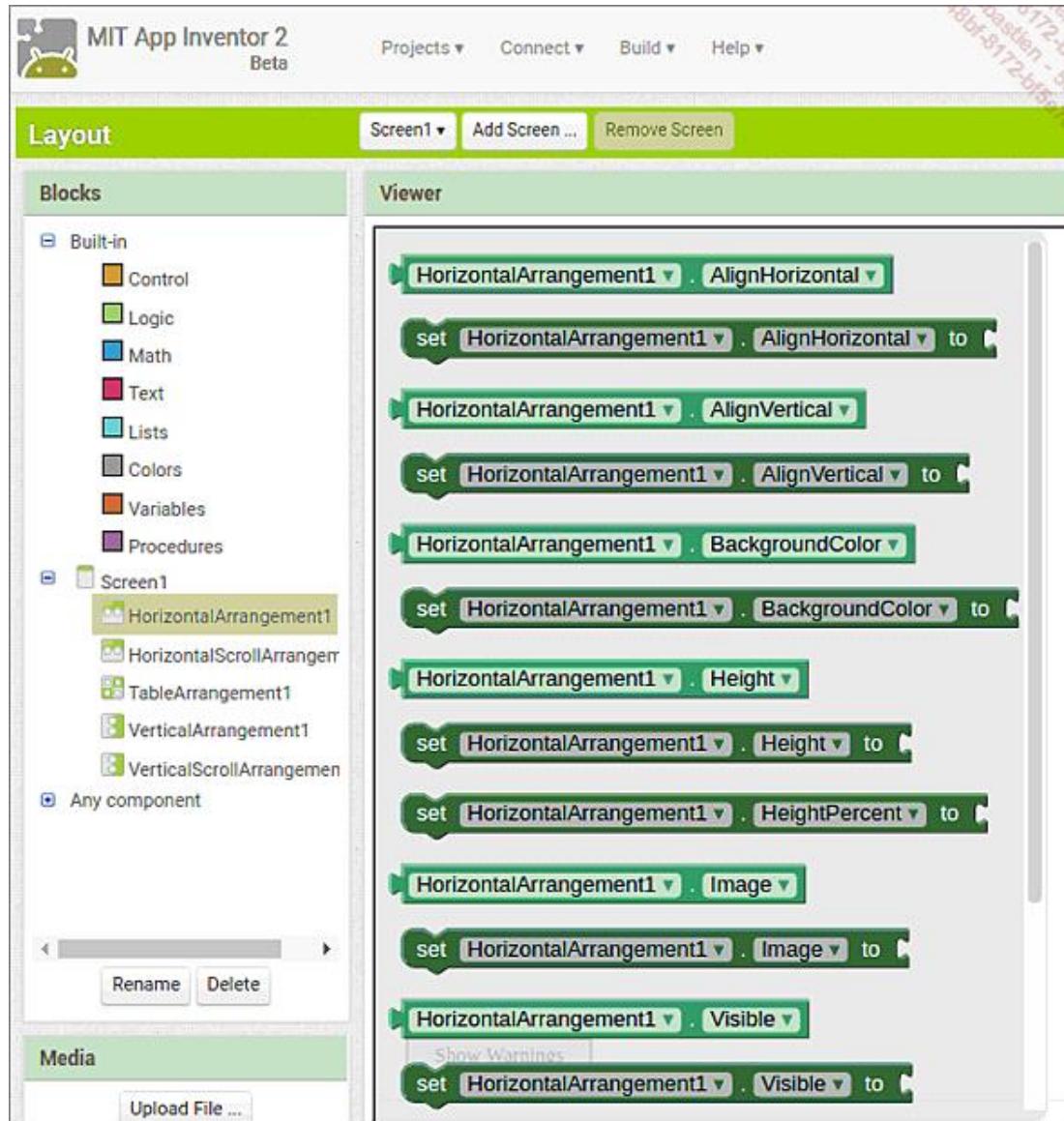
FollowLinks

Si cette option est activée sur true alors vous pourrez naviguer de page en page. Si cette propriété est sur false vous ne pourrez pas naviguer de page en page.

Les fonctions des composants Layout

Nous venons de voir les composants les plus utilisés pour faire fonctionner une application classique. Voyons maintenant comment rendre une application esthétique grâce aux composants **Layout**.

Les blocs **Layout** ne possèdent pas d'événements, vous pouvez simplement utiliser les propriétés de ces derniers dans les blocs logiques :



Les principales propriétés que vous trouverez dans les composants Layout sont :

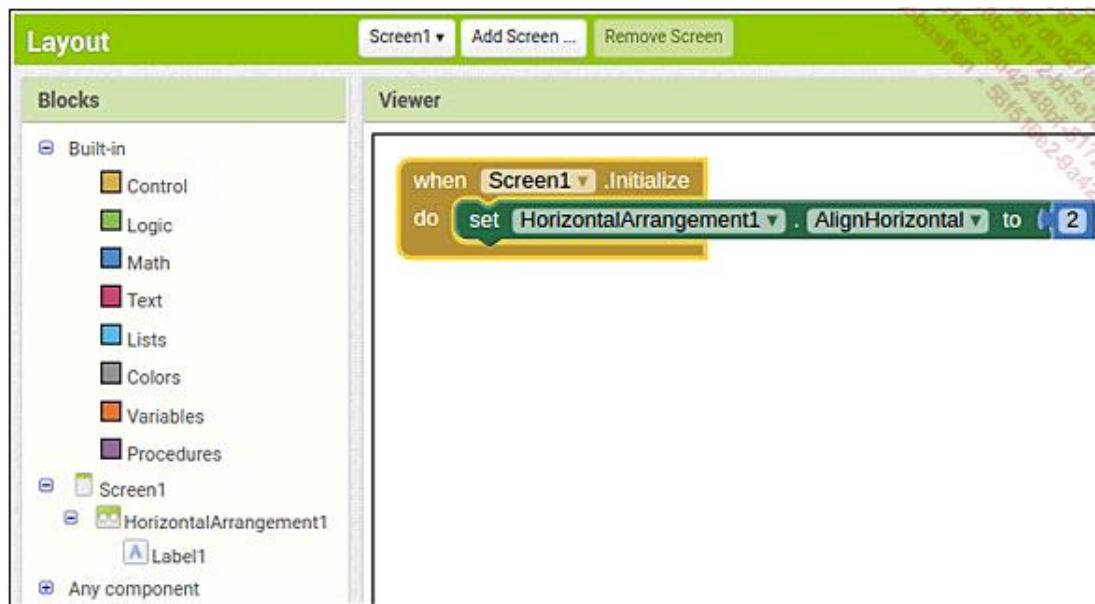
- AlignHorizontal
- AlignVertical
- BackgroundColor
- Height
- HeightPercent
- Image
- Visible

- Width
- WidthPercent

AlignHorizontal

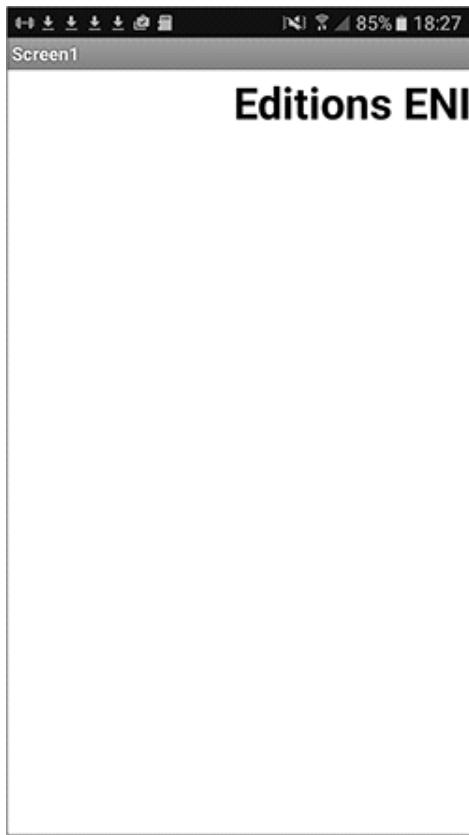
Cette propriété vous permet d'indiquer comment vous souhaitez que le contenu compris à l'intérieur de ce composant soit aligné de manière horizontale.

Par exemple, si vous écrivez le bloc logique de cette manière :



Alors le contenu de celui-ci sera aligné sur la droite. Dans cet exemple, **2** indique qu'il s'agit d'un alignement sur la droite.

Vous trouverez ci-dessous le rendu :



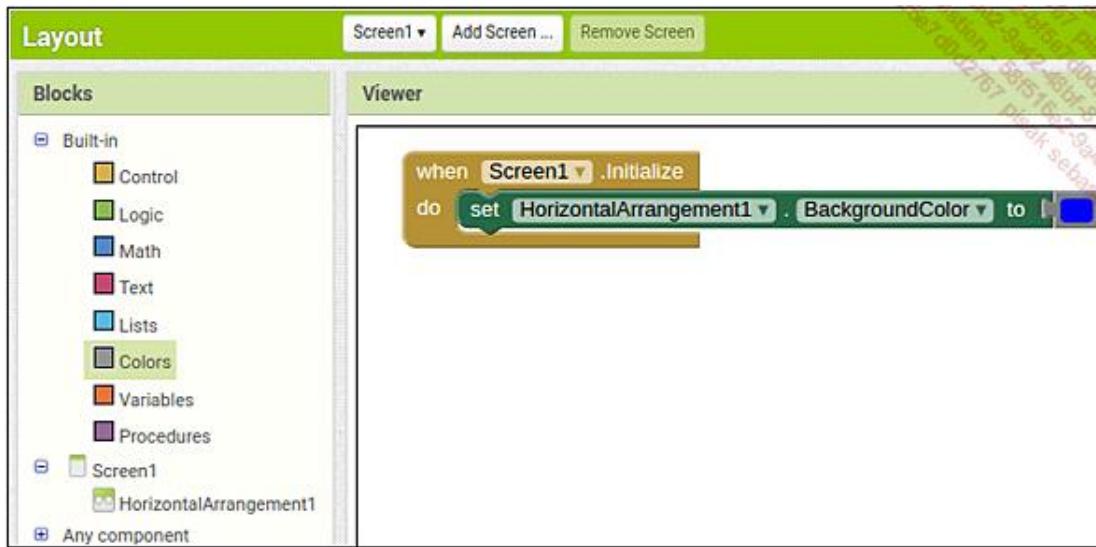
Si vous souhaitez aligner le texte sur la gauche, il faudra utiliser la valeur **1**, la valeur **3** pour le centrer.

AlignVertical

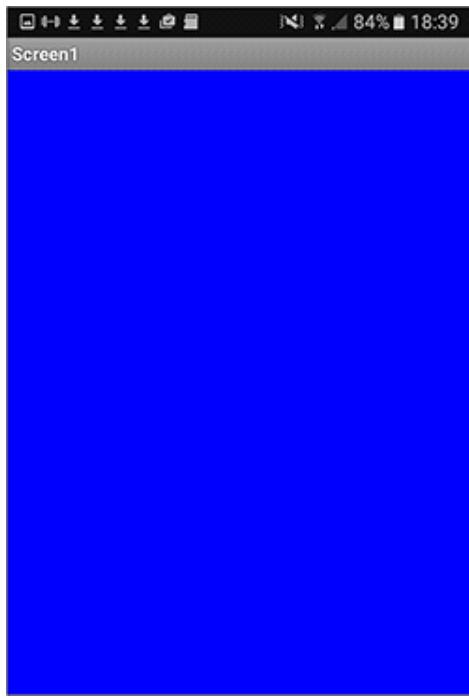
Même principe que pour AlignHorizontal sauf que l'alignement sera de haut en bas.

BackgroundColor

Permet de définir le fond d'écran, pour cela vous devez simplement y ajouter un bloc de couleur. Par exemple :



Ce qui donnera une fois l'application lancée :



Height

Vous permet d'indiquer la hauteur en pixels que vous souhaitez attribuer à cet élément.

Exemple :



Vous permettra d'avoir un bloc d'une hauteur de 50 pixels.

HeightPercent

Vous permet d'indiquer la hauteur en pourcentage que vous souhaitez attribuer à cet élément.

Exemple :



Image

Vous permet de renseigner le chemin de l'image que vous indiquez et de l'afficher sur le bloc. Ce chemin peut être une URL, par exemple : http://www.editions-eni.fr/styles/france/images/logo_ENI.png

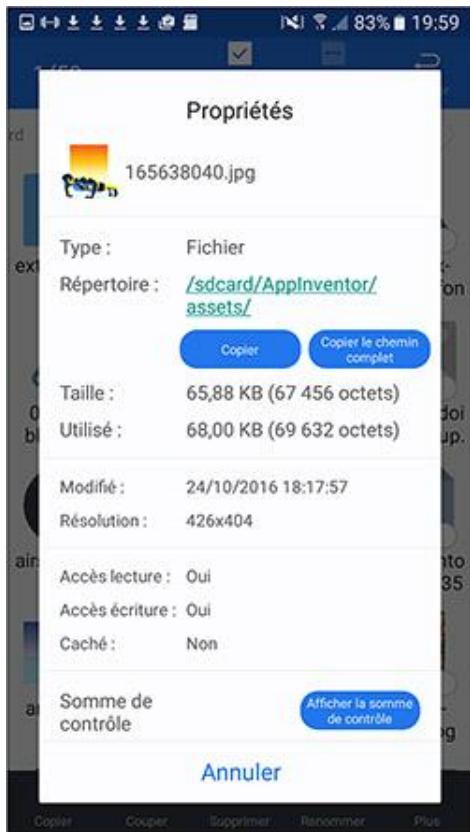


Ce qui donnera :



Vous pouvez également faire appel à une image se trouvant sur le smartphone. Cependant, trouver le chemin d'une image sur le smartphone peut être un peu hasardeux ; pour ce faire, nous vous conseillons d'installer un Explorateur de fichiers, par exemple : <https://play.google.com/store/apps/details?id=com.estrongsl.pop&hl=fr>

Cet explorateur de fichiers vous permettra d'identifier rapidement le chemin de n'importe quel fichier, ici par exemple, pour une image spécifique :



Ce qui vous permettra d'afficher l'image de la manière suivante :



Visible

La propriété **Visible** vous permet d'indiquer si le composant sera visible ou non en utilisant les attributs **true** (sera visible) et **false** (ne sera pas visible).

Width

Idem que pour Height sauf qu'il s'agit ici de la largeur.

WidthPercent

Idem que pour HeightPercent sauf qu'il s'agit ici de la largeur.

Les fonctions des composants Media

Les composants **Media** sont très variés : certains n'offrent pas beaucoup de personnalisation alors que d'autres vous proposeront de nombreuses fonctionnalités.

1. Camcorder

RecordVideo

Comme son nom l'indique, cette fonction vous permet d'enregistrer une vidéo depuis votre smartphone en utilisant la fonction intégrée à celui-ci ; par exemple, le code ci-dessous déclenchera le composant vidéo à l'ouverture de l'écran :

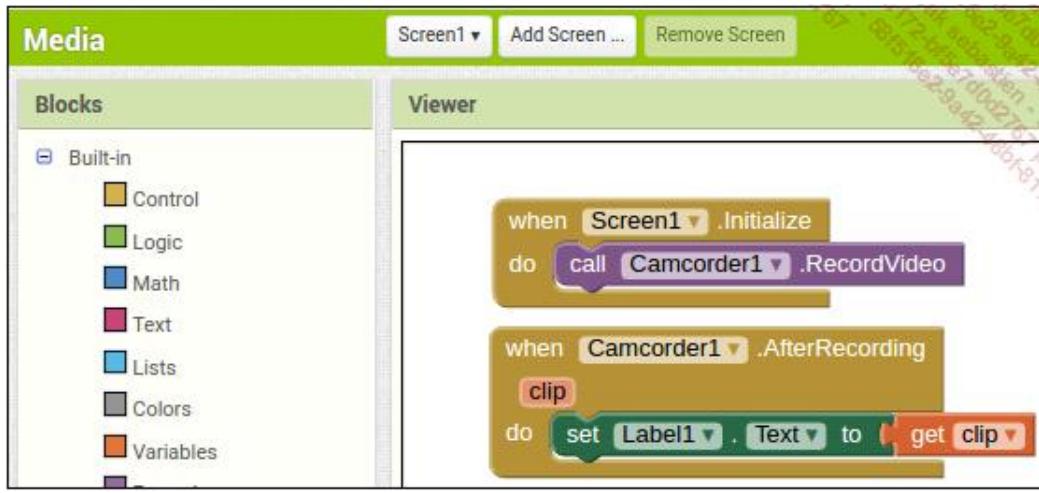


Ce qui donnera :

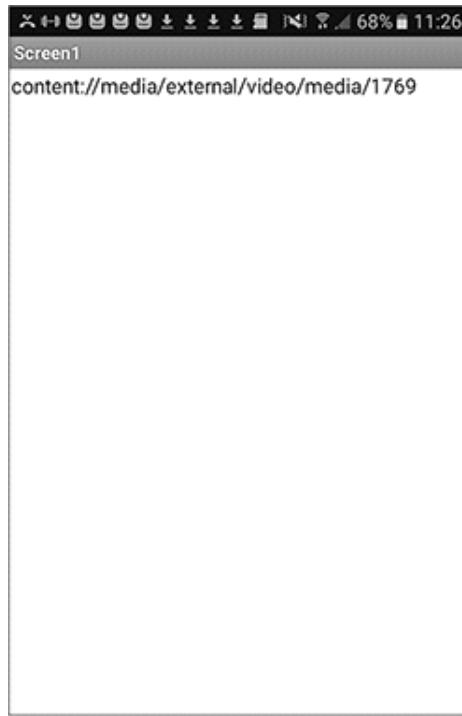


AfterRecording

Cet événement vous permet de connaître le chemin d'accès à la vidéo (représentée par une variable du nom de **clip**) que vous venez d'enregistrer et qui est donc stockée dans votre téléphone ; par exemple, si vous écrivez le code suivant :



Ici, nous demandons à afficher le chemin d'accès à la vidéo enregistrée par notre smartphone ; nous obtenons :



Cet événement peut vous paraître ici d'un intérêt limité, mais si nous l'associons à un lecteur de vidéo, nous obtiendrons la lecture de la vidéo que nous venons d'enregistrer.

2. Camera

Le composant Camera offre exactement les mêmes fonctionnalités que le composant Camcorder, nous n'allons donc pas les détailler à nouveau, la seule différence est qu'au lieu de faire appel à la caméra vidéo, il fait appel à l'appareil photo :



3. ImagePicker

AfterPicking

Cet événement se déclenche une fois qu'une image a été sélectionnée.

BeforePicking

Cet événement se déclenche une fois que vous avez cliqué sur le bouton ImagePicker mais avant qu'une image ait été sélectionnée.

GotFocus

Cet événement se déclenche lorsque le composant ImagePicker est sélectionné au détriment d'un autre.

LostFocus

Cet événement se déclenche lorsque le composant ImagePicker est désélectionné au détriment d'un autre.

TouchDown

Cet événement se déclenche lorsque le bouton est appuyé.

TouchUp

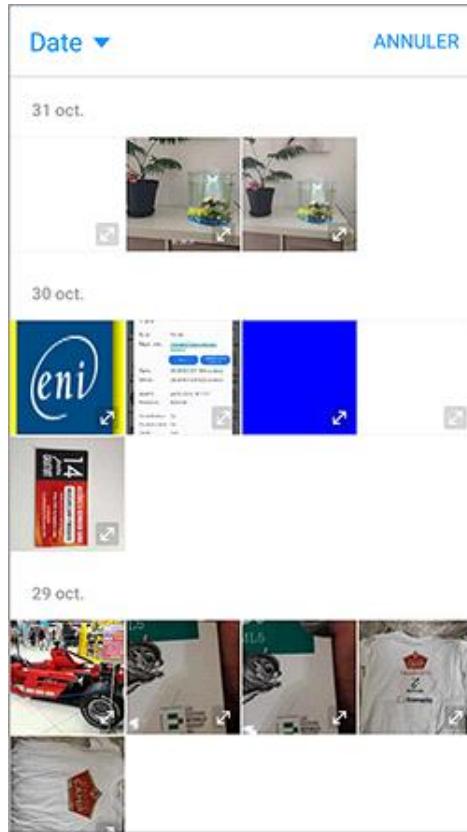
Cet événement se déclenche lorsque le bouton est relâché.

Open

Cette fonctionnalité permet simplement d'ouvrir la galerie de photos de votre smartphone. Par exemple, le code suivant :



va permettre d'ouvrir la galerie photos du smartphone :



Le composant `ImagePicker` permet également d'appeler les propriétés suivantes :

- `BackgroundColor`
- `Enabled`
- `FontBold`
- `FontItalic`
- `FontSize`
- `Height`
- `HeightPercent`
- `Image`
- `ShowFeedback`
- `Text`
- `TextColor`
- `Visible`
- `Width`
- `WidthPercent`

Ces propriétés ayant été déjà présentées pour d'autres composants, nous n'allons pas à nouveau les détailler.

`ShowFeedback`

Cette propriété permet simplement d'indiquer si vous souhaitez ou non que soit visible le fait que le bouton ait été

cliqué. Dans l'exemple ci-dessous, nous indiquons à l'application que nous ne souhaitons pas que le bouton change d'apparence lorsqu'il a été cliqué.



4. Player

Completed

Cet événement permet de déclencher une série d'actions de votre choix lorsque le lecteur est arrivé à la fin de son enregistrement. Par exemple, vous pouvez afficher une info-bulle indiquant que la musique est terminée.

OtherPlayerStarted

Cet événement permet de déclencher des actions si d'autres players commencent à être activés. Par exemple, vous pouvez être intéressé pour mettre en pause votre player si un autre player vient de se déclencher.

PlayerError

Vous permet de déclencher des actions si le player rencontre des erreurs, vous pouvez également faire afficher par l'intermédiaire de la variable message, des informations sur l'erreur.

Pause

Permet de mettre le lecteur en pause.

Start

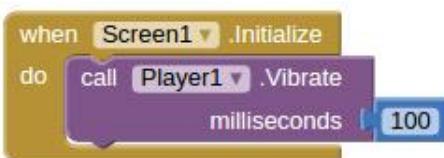
Permet de faire démarrer le lecteur.

Stop

Permet d'arrêter le lecteur.

Vibrate milliseconds

Active la fonction vibrer de votre smartphone. Exemple :



Lorsque cette application est lancée, votre smartphone vibre pendant 100 millisecondes au démarrage.

IsPlaying

Cette propriété vous permet de déclencher des actions si le player est en train de jouer.

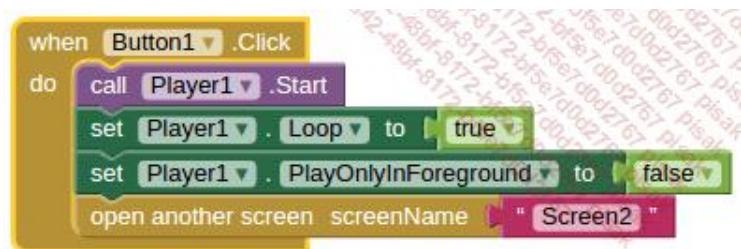
Loop

Permet d'indiquer si vous souhaitez qu'une fois la musique jouée, celle-ci soit jouée à nouveau, et ce de manière infinie. Si tel est le cas, vous devez lui attribuer la valeur true.

PlayOnlyInForeground

Cette propriété permet d'indiquer si vous souhaitez ou non que la musique activée du player soit toujours jouée que vous soyez sur l'écran associé ou non.

Cette option peut prendre deux valeurs, true ou false. Si vous lui attribuez la valeur true, alors la musique s'arrêtera si vous changez d'écran, si vous lui attribuez la valeur false celle-ci continuera ; par exemple, c'est le cas de cette application :



En naviguant de l'écran 1 à l'écran 2 la musique reste active.

Source

Permet d'indiquer le chemin pour accéder au fichier que vous souhaitez lire avec le player.

Volume

Permet d'indiquer le niveau sonore du player.

5. Sound

Sound étant un composant très similaire à Player, nous allons nous intéresser uniquement aux fonctionnalités différentes.

Resume

Permet de reprendre le son là où il a été arrêté après avoir utilisé la fonction **Pause**.

MinimumInterval (ms)

Cette propriété correspond au temps qui s'écoule entre la lecture de plusieurs sons. Imaginons par exemple que vous écriviez le code suivant :



Cela signifie que lorsque vous allez cliquer sur le bouton qui déclenche le son alors il n'y aura pas d'écart entre ce son et le nouveau son qui sera joué lorsque vous cliquerez à nouveau sur le bouton. Les sons vont donc se chevaucher. En général, vous souhaiterez éviter cela et aurez une valeur **MinimumInterval** correspondant à la durée du son qui est joué.

6. SoundRecorder

Les fonctionnalités proposées pour le composant SoundRecorder sont très similaires à celles du composant Camcorder. Nous n'allons donc pas les développer à nouveau.

SavedRecording

Cette propriété permet d'indiquer le chemin d'accès au dossier dans lequel vous souhaitez enregistrer les fichiers enregistrés sur votre smartphone.

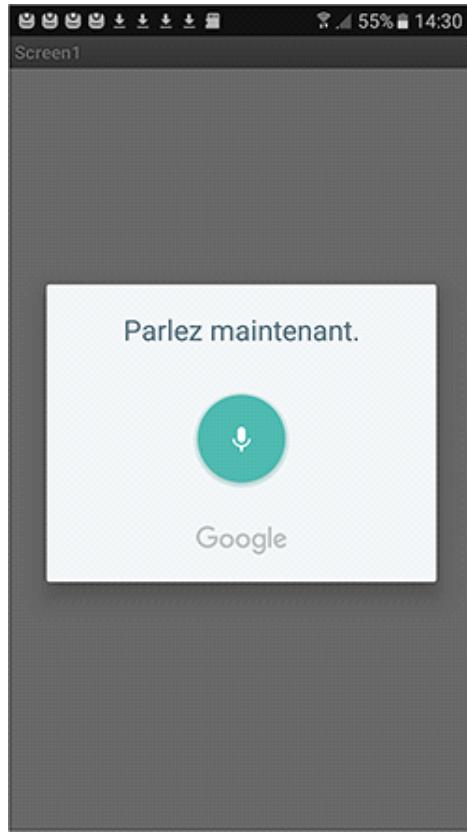
7. SpeechRecognizer

GetText

Cette fonctionnalité permet d'appeler le composant SpeechRecognizer. Par exemple, en écrivant le code suivant :



vous obtiendrez :



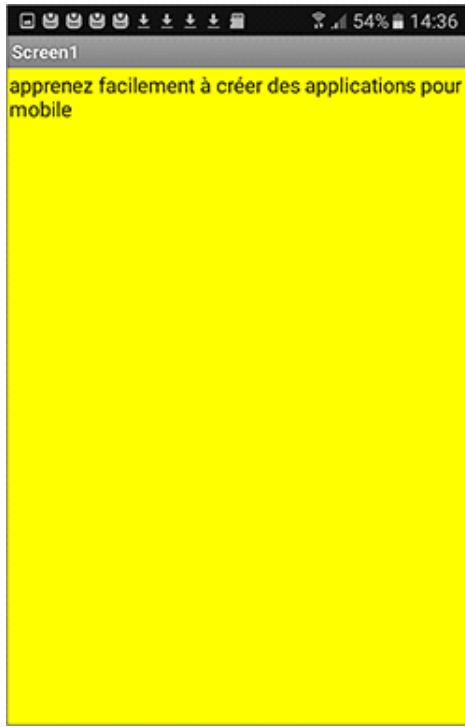
Cependant, en tant que tel, rien ne va se passer par la suite, il va donc falloir que vous fassiez appel aux blocs d'événements associés à ce composant. C'est ce que nous allons voir ci-dessous.

AfterGettingText

L'événement **AfterGettingText** permet d'indiquer le comportement que nous souhaitons que l'application exécute lorsque le texte a été récupéré une fois le composant `SpeechRecognizer` écouté ; nous avons décidé ici de mettre le résultat dans un champ **Label.Text** :



La phrase que nous avons prononcée est "apprenez facilement à créer des applications pour mobile"... et le résultat est :



BeforeGettingText

Permet de déclencher une action avant que le composant SpeechRecognizer renvoie le texte.

8. TextToSpeech

SpeakMessage

Cette fonctionnalité permet à l'application de lire le message que vous lui indiquez, par exemple :



Cette application va vous lire à haute voix le message "Bonjour et bienvenue dans mon appli mobile".

Les événements AfterSpeaking et BeforeSpeaking sont très similaires à ceux que nous avons déjà rencontrés précédemment, nous n'allons donc pas les détailler ici.

Country

Permet d'indiquer quel accent doit prendre la voix dictant votre message. Par exemple, l'accent d'un américain sera différent de celui d'un britannique. Les valeurs que peuvent prendre cette propriété sont listées dans l'écran **Designer** dans la partie **Propriétés**, par exemple : AUS, AUT, BEL, BLZ...

Language

Similaire à la propriété précédente, à l'exception que les valeurs sont ici de, en, es, fr, it et correspondent à la langue utilisée.

Pitch

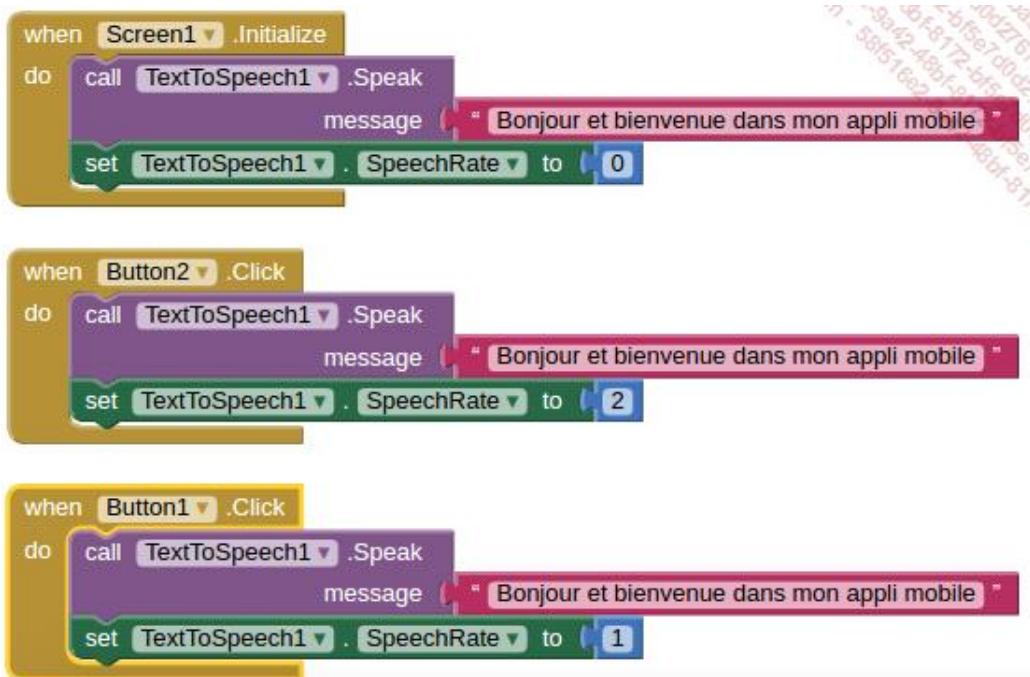
Cette propriété permet de régler le niveau de la synthèse vocale. Cette valeur est comprise entre 0 et 2 où **0** devrait correspondre à un son plus robotique que **1** et **2**. Nous vous conseillons de tester l'ensemble de ces trois valeurs par exemple :



SpeechRate

Cette propriété correspond à la vitesse de prononciation où **0** correspond à une vitesse lente et **2** à une vitesse rapide.

Pour tester la différence entre les trois options, vous pouvez créer l'application suivante :



9. VideoPlayer

Nous n'allons pas ici présenter les deux événements que sont `Completed` et `VideoPlayerError` car ces deux fonctions sont similaires à celles que nous avons vues pour le composant **Player** précédemment.

GetDuration

Permet de récupérer la durée d'une vidéo.

10. YandexTranslate

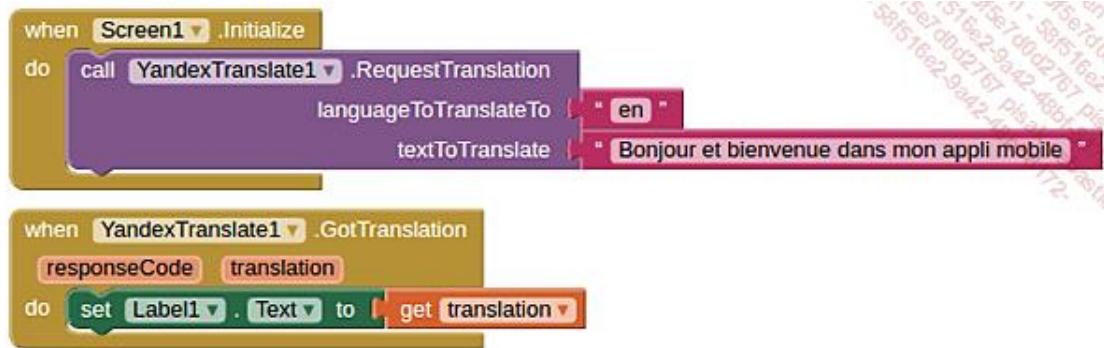
```
RequestTranslation languageToTranslateTo textToTranslate
```

Cette fonction permet d'appeler le composant YandexTranslate et de traduire le texte indiqué. Afin de pouvoir l'utiliser de manière utile, il vous faudra connaître l'événement à lui associer à savoir GotTranslation.

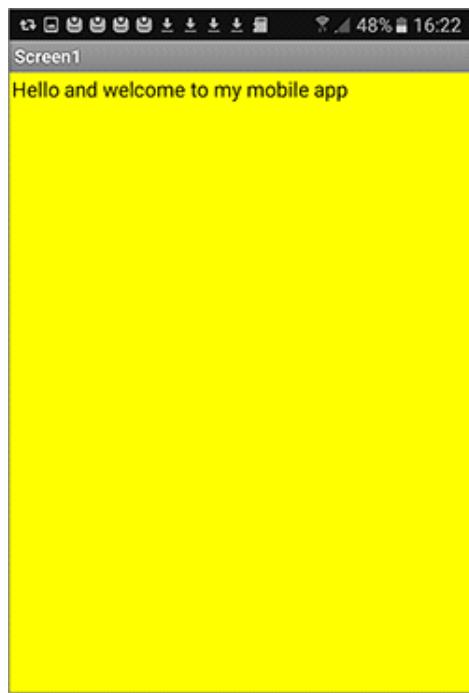
Vous aurez également besoin de connaître les correspondances de langue de Yandex Translate que vous trouverez à l'URL suivante : <https://tech.yandex.com/translate/doc/dg/concepts/api-overview-docpage/>

GotTranslation

Cet événement permet d'indiquer les actions que vous souhaitez effectuer une fois la traduction récupérée. Par exemple, si nous créons l'application suivante :



où **en** indique que nous souhaitons traduire le texte "Bonjour et bienvenue dans mon appli mobile" en anglais, voilà le résultat obtenu :



Les fonctions des composants Drawing and Animation

Bien que peu nombreux, les composants **Drawing and Animation** (**Ball**, **Canvas**, **ImageSprite**) possèdent beaucoup de fonctionnalités. Étant donné que les composants **Ball** et **ImageSprite** sont dépendantes de **Canvas**, nous allons présenter le composant **Canvas** en premier.

1. Canvas

Clear

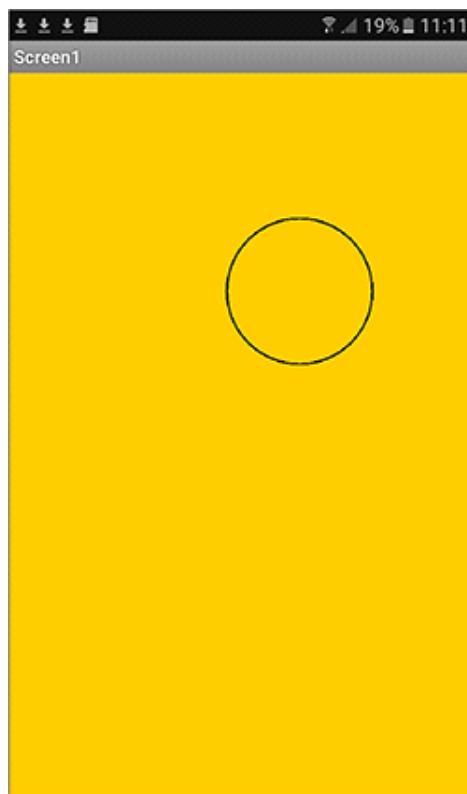
Cette méthode permet d'effacer tout ce qui est écrit sur le canvas (cadre).

DrawCircle centerX centerY radius fill

Cette fonction permet de tracer un cercle sur le canvas (cadre) dont le centre sont les coordonnées X et Y que vous mentionnez. Radius fait ici référence au rayon du cercle en pixels, et fill indique si vous souhaitez que celui-ci soit rempli ou non. Ainsi, lorsque vous écrivez cela :



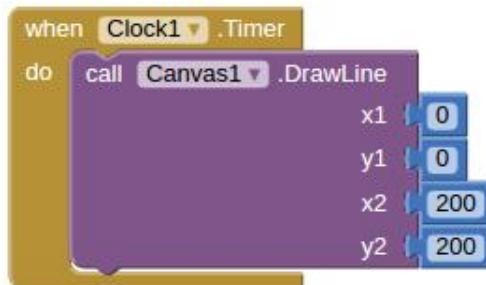
Vous obtenez :



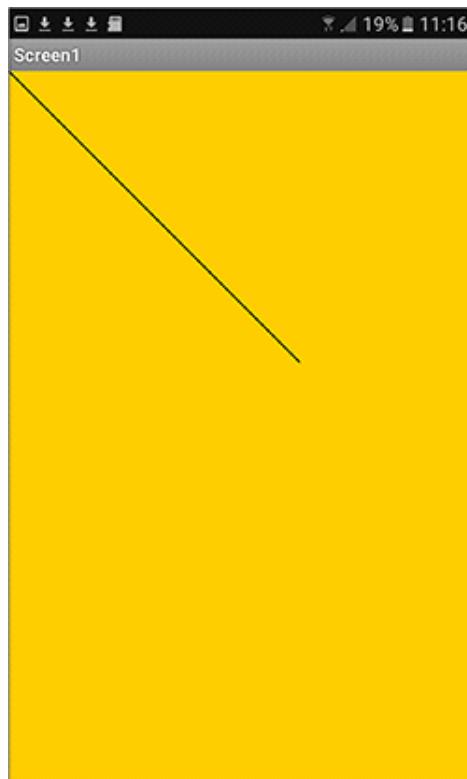
```
DrawLine x1 y1 x2 y2
```

Permet de tracer une ligne en fonction des coordonnées x1 y1 et x2 y2 que vous lui indiquez.

Exemple :



Ce qui donne :

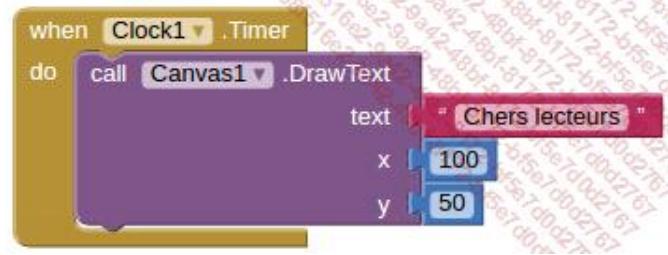


```
DrawPoint x y
```

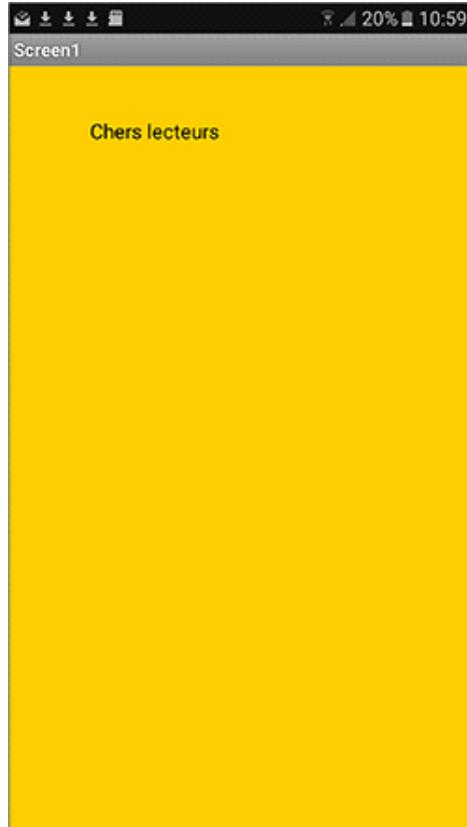
Comme le nom de cette méthode le suggère, elle permet de tracer un point sur le canvas dont les coordonnées sont x y. Nous n'allons pas détailler davantage l'utilisation de cette fonction.

```
DrawText text x y
```

Comme son nom l'indique, cette méthode permet d'afficher un texte sur le canvas qui commencera dès les coordonnées x et y. Par exemple, le code suivant :

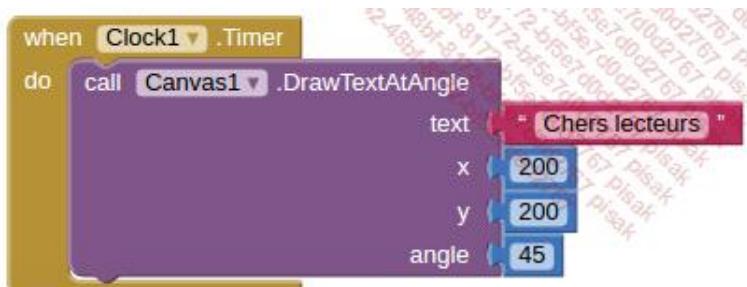


Donnera :

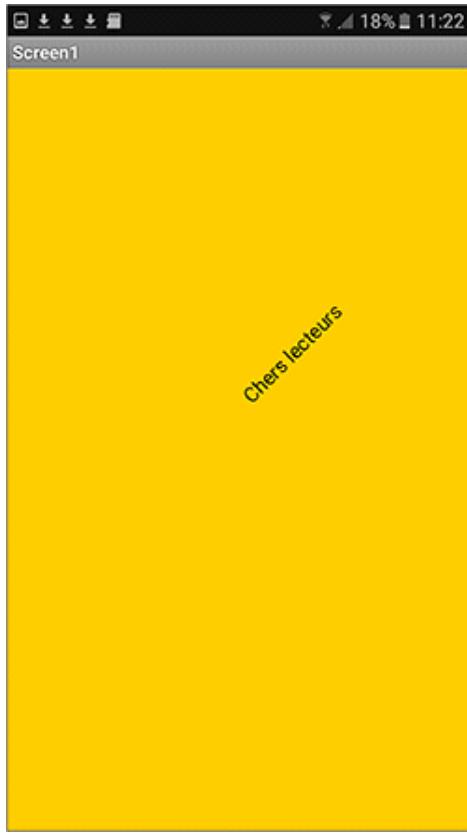


DrawTextAtAngle text x y angle

Permet d'écrire un texte incliné en fonction de l'angle d'inclinaison que vous lui indiquez en degrés, exemple :



Ce qui donnera :



GetBackgroundColor x y

Permet de récupérer la couleur de fond des coordonnées x et y indiquées.

GetPixelColor x y

Permet de récupérer la couleur du point x et y indiqué.

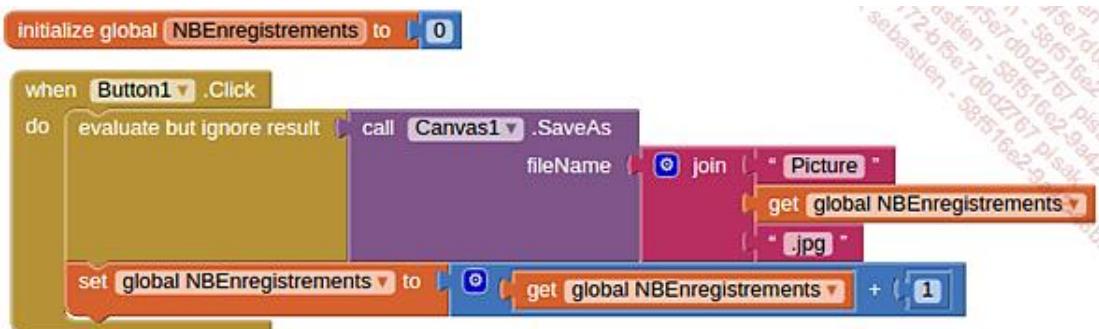
Save

Permet de sauvegarder le canvas dans votre smartphone. Les noms de fichier par défaut sont en général de type : app_inventor_XXXX.png

SaveAs fileName

Permet de sauvegarder le canvas en indiquant le nom de fichier que vous souhaitez lui donner.

Voici un exemple d'application :



Grâce à cette application, chaque clic effectué sur un bouton permet l'enregistrement du canvas dans une image différente. La première image portera le nom de Picture1.jpg, la deuxième Picture2.jpg et ainsi de suite.

Pour rappel, vous pouvez installer un explorateur de fichiers sur votre smartphone pour retrouver facilement le chemin d'accès aux images.

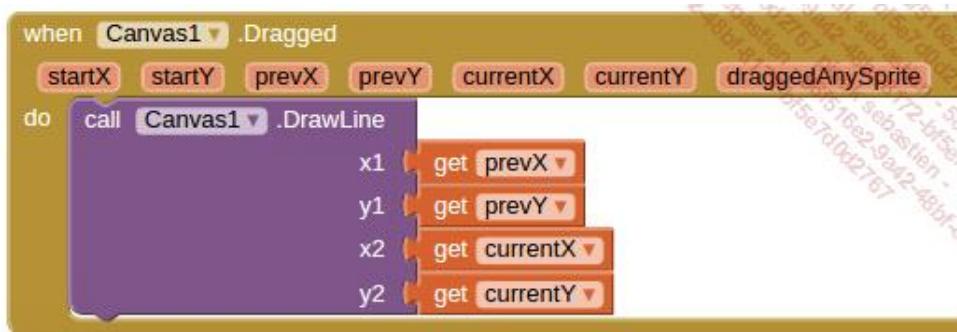
`SetBackgroundColor x y color`

Permet de définir la couleur d'un point en fonction de ses coordonnées x et y.

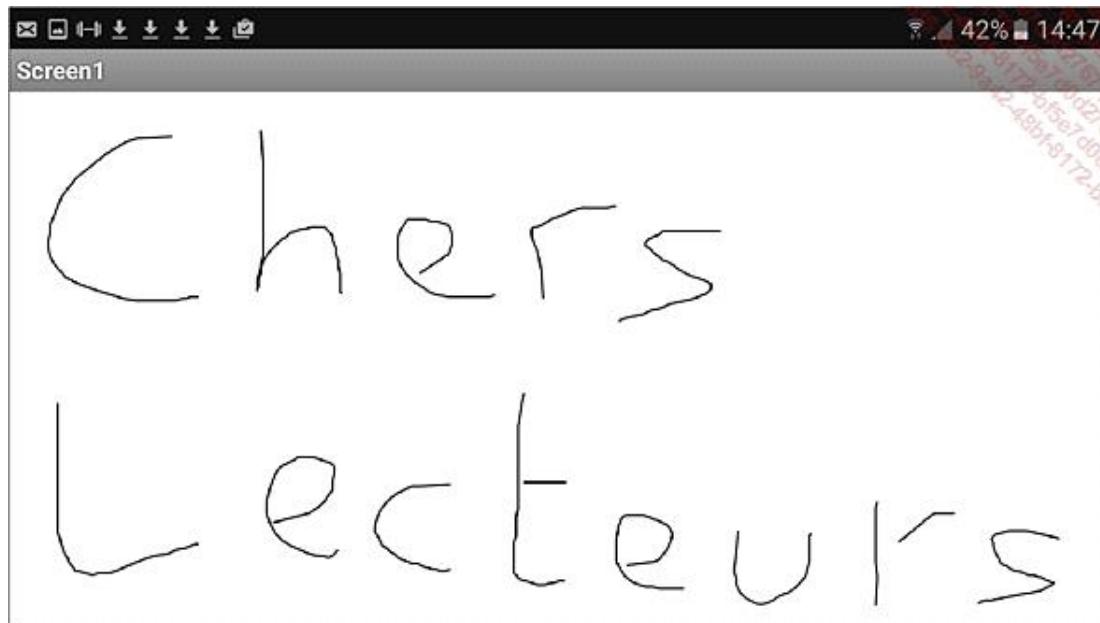
Les événements du Canvas

Dragged

Cet événement sert à faire déplacer des sprite (des images sur le canvas) ou à dessiner d'un point à un autre de l'écran. L'idéal pour l'illustrer est de réaliser une application de ce type :



qui va permettre de faire des dessins avec le doigt sur l'écran du smartphone, par exemple :



TouchDown

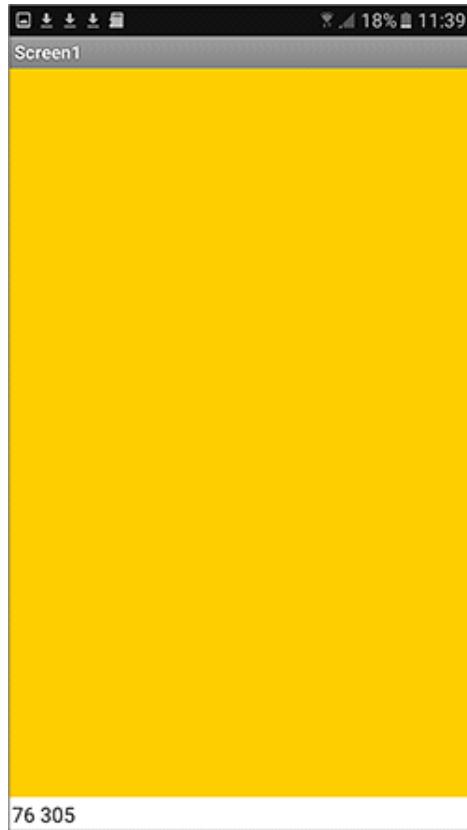
Il s'agit de l'événement qui est déclenché lorsque l'utilisateur entre en contact avec l'écran.

TouchUp

Il s'agit de l'événement qui est déclenché lorsque l'utilisateur enlève son doigt qui était au contact de l'écran.

Touched

Cet événement vous permet de récupérer les coordonnées x et y sur lesquels l'utilisateur a appuyé lorsqu'il a mis son doigt sur l'écran, par exemple ici :



Nous savons que l'utilisateur a appuyé sur un point de coordonnées x = 76 et y = 305.

Les propriétés du canvas

LineWidth

Permet d'indiquer l'épaisseur des lignes tracées.

PaintColor

Permet d'indiquer la couleur des lignes qui seront tracées sur le canvas.

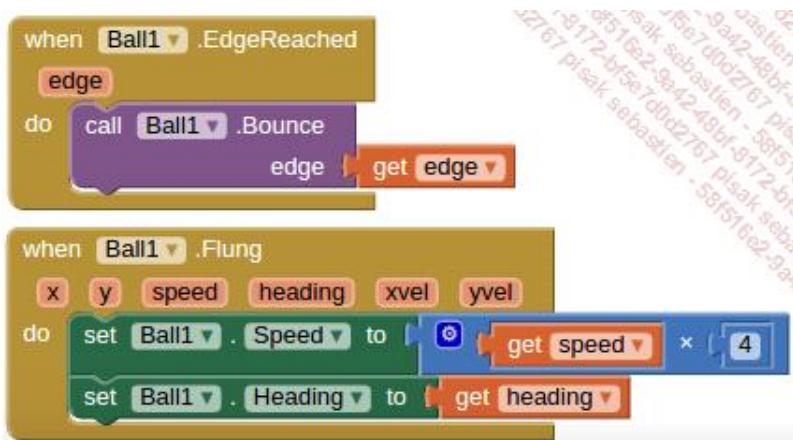
2. Ball

Le composant **Ball** vous permet d'utiliser une balle sur votre canvas.

Bounce edge

Cette méthode fonctionne avec l'événement EdgeReached. Lorsque la balle touche le bord d'un canvas, alors elle rebondit.

Imaginons une application pour laquelle une balle rebondirait de manière indéfinie en fonction d'un lancer que nous aurions effectué, cette application ressemblerait à :



Dans cet exemple, lorsque nous faisons glisser notre doigt sur la balle, celle-ci prend une certaine vitesse et se déplace sur l'écran, lorsque notre balle atteint les bords de l'écran, celle-ci rebondit par rapport au bord de l'écran.

CollidingWith other

Cette fonction s'utilise principalement avec l'événement CollidedWith et permet d'indiquer que lorsque la balle est entrée en collision avec un autre composant alors une action va se passer.

MoveIntoBounds

Cette fonction permet de remettre un composant à sa place si celui-ci quitte les limites du canvas.

MoveTo x y

Cette méthode va déplacer la balle aux coordonnées x et y indiquées.

PointInDirection x y

Cette méthode permet d'indiquer dans quelle direction vous souhaitez que votre balle se déplace. Par défaut, les propriétés d'une balle sont définies pour que la balle ait une vitesse égale à 0. Ce qui fait que vous ne verrez pas la balle bouger. En revanche, si vous lui attribuez la valeur vitesse égale à 1, alors vous verrez la balle se diriger dans la direction que vous lui avez indiquée.

PointTowardstarget

Cette méthode permet de diriger la balle dans la direction du sprite indiqué.

Dragged

Cet événement permet de déplacer la balle sur le canvas. Typiquement, si vous créez une application telle que celle-ci :



Alors vous pourrez déplacer la balle où bon vous semble sur l'écran en laissant votre doigt appuyé sur celle-ci. Lorsque vous retirez votre doigt, la balle prend les dernières coordonnées du point de pression que vous avez eu avec le canvas et laisse la balle à cet emplacement.

Flung

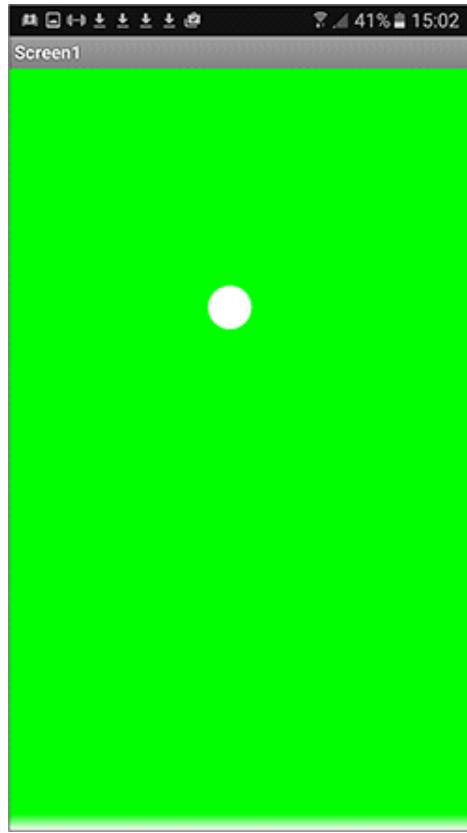
Permet de récupérer les coordonnées de départ lorsque l'utilisateur a commencé à toucher la balle avec le doigt.

Pour en voir rapidement l'utilité et le fonctionnement, nous pouvons réaliser une application telle que celle-ci :



Celle-ci ne comprend que deux composants, une balle et un canvas.

Nous avons volontairement multiplié par 4 la vitesse de la balle afin que l'application soit plus dynamique. De base, la vitesse de l'événement Flung n'est pas très rapide. Ici, lorsque notre doigt va agir sur la balle, la balle va prendre la direction montrée par le mouvement du doigt sur l'écran :

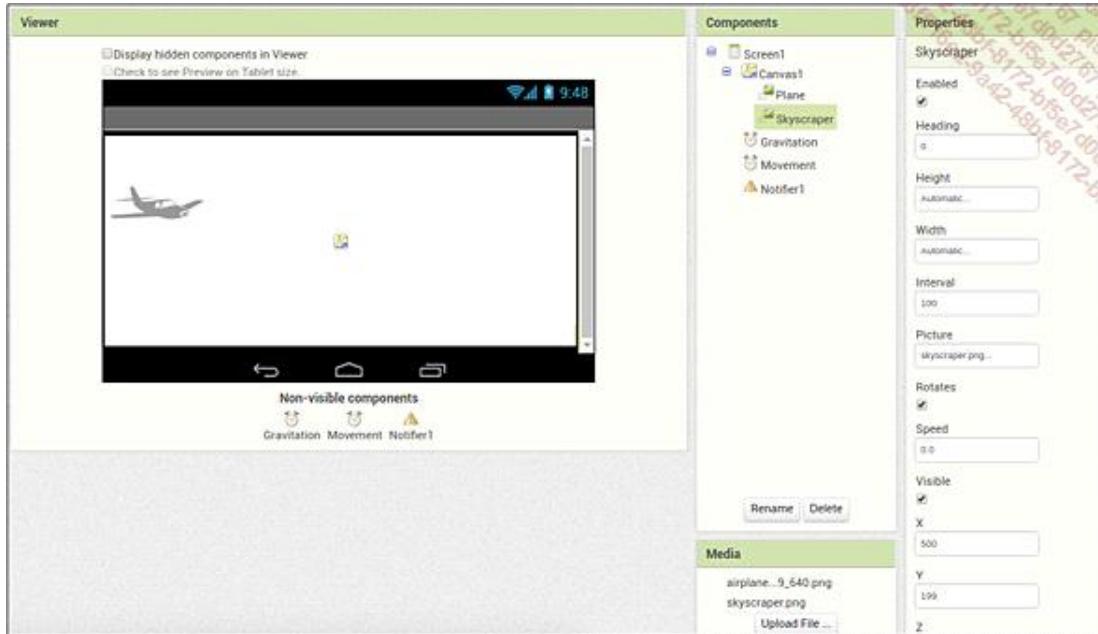


Les autres événements ayant déjà été abordés, nous préférons ne pas les détailler à nouveau.

3. ImageSprite

Les fonctionnalités offertes par le composant **ImageSprite** offrent à peu près la même expérience que le composant **Ball**.

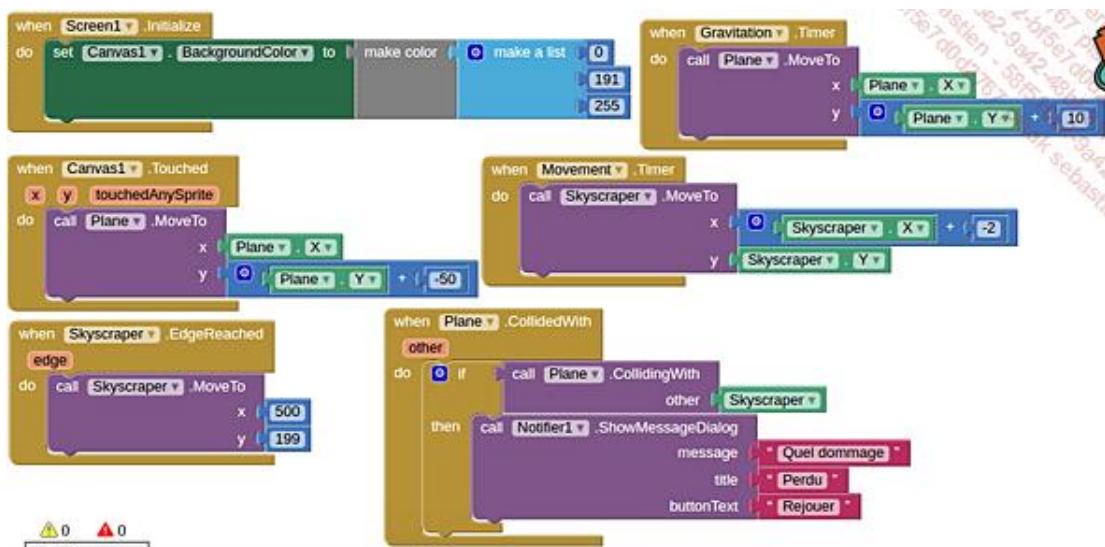
Afin de pouvoir illustrer les fonctionnalités supportées par ce composant, nous allons réaliser une application de type Flappy Bird :



Cette application contient :

- un composant Canvas qui représente notre écran de jeu,
- un composant Image symbolisé ici par un avion (**Plane**),
- un composant Image symbolisé ici par un gratte-ciel (**skyscraper**) dont l'affichage n'apparaît pas ici car ses coordonnées sortent légèrement en dehors de l'écran,
- un composant Clock qui s'appelle **Gravitation**, celui-ci a pour objectif de diriger notre avion vers le bas sur le canvas,
- un composant Clock qui s'appelle **Movement**, celui-ci a pour objectif de diriger notre gratte-ciel de droite à gauche sur l'écran,
- un composant **Notifier** qui a pour but d'afficher un message si l'avion et le gratte-ciel entrent en collision.

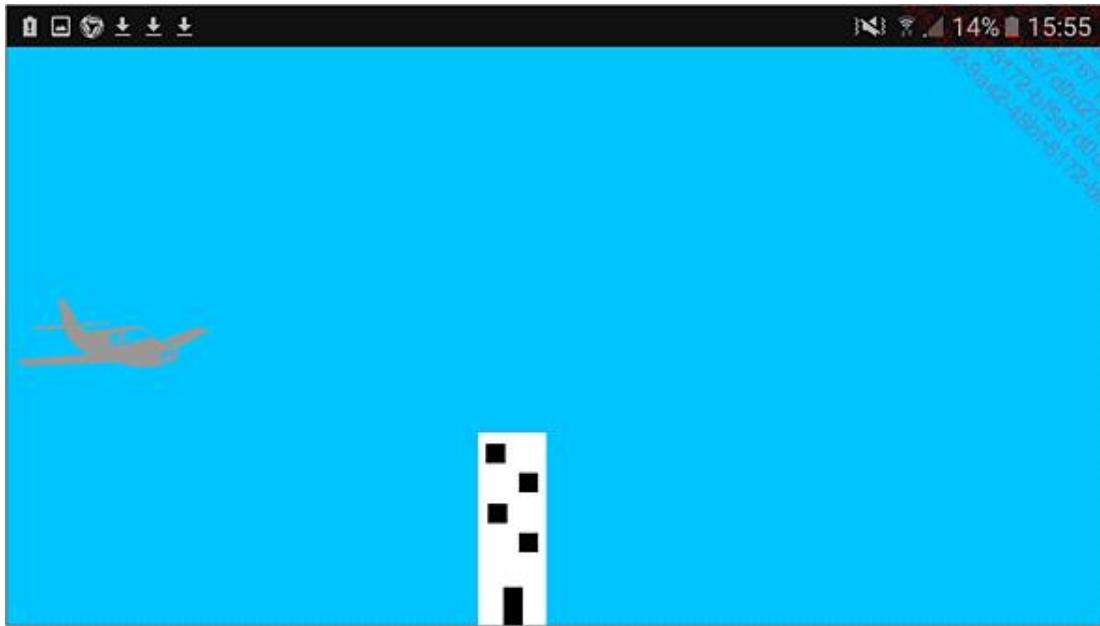
Le code de cette application ressemble à cela :



Voici comment lire cette application de gauche à droite :

- Lorsque l'écran démarre, le canvas prend la couleur d'un ciel bleu azuré.
- À chaque fois que le composant **Clock Gravitation** se déclenche, l'avion descend sur l'écran de **10**.
- Lorsque le canvas est touché, l'avion remonte de **50**.
- Lorsque le composant **Clock Movement** se déclenche, le gratte-ciel se déplace de **2** pixels vers la gauche.
- Lorsque le gratte-ciel touche un bord du canvas, celui-ci est déplacé au point suivant x = **500** et y = **199**.
- Lorsque l'avion entre en collision avec le gratte-ciel alors s'affiche une fenêtre avec un message pour l'utilisateur.

Une fois l'application lancée, cela donnera ce résultat :



Ici, notre application ne fait rien de bien extraordinaire, cependant vous pouvez déjà imaginer l'ajout d'un système de scoring, de tir pour l'avion, d'obstacles aériens...

Nous venons de voir dans cette application les fonctions suivantes :

- CollideWith
- EdgeReached
- Touched
- CollidingWith other
- MoveTo

Les fonctionnalités offertes par les images étant similaires à celles des balles et des canvas, nous n'allons pas redétailler à nouveau les autres fonctions.

Les fonctions des composants Sensor

1. AccelerometerSensor

Le composant Accéléromètre (**AccelerometerSensor**) ne possède que deux événements :

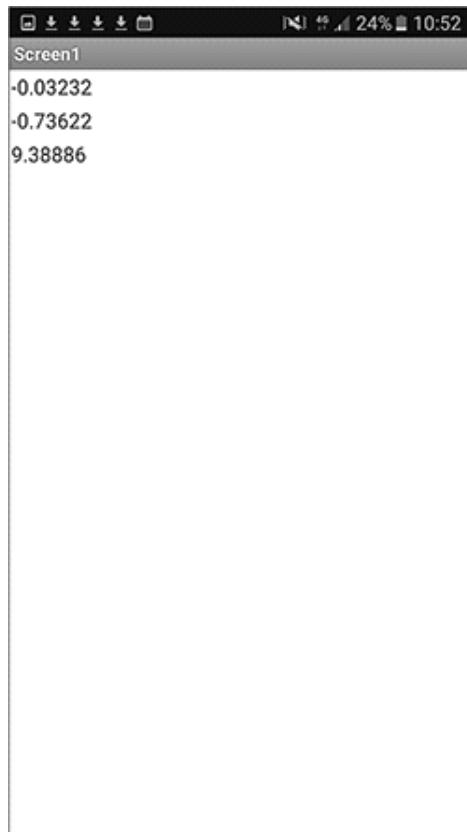
- AccelerationChanged
- Shaking

AccelerationChanged

Cette fonction permet d'indiquer si le smartphone a bougé de position en fonction des coordonnées x, y et z. Il s'agit d'un composant très sensible. En réalisant l'application suivante :



vous verrez que les coordonnées x, y, z de votre téléphone changent en continu même quand celui-ci est immobile :



Shaking

Cette fonction permet d'indiquer que vous êtes en train de secouer le téléphone. Nous pourrions imaginer l'application suivante :



qui affichera à l'utilisateur un message lui indiquant que le téléphone est secoué ou a été secoué :



2. BarcodeScanner

DoScan

Cette fonction permet d'activer votre appareil photo et en même temps utilise une fonctionnalité de reconnaissance d'image. Si cette image paraît ressembler à un code-barres alors l'application réagira. Par exemple, si vous réalisez l'application suivante :



celle-ci lancera la fonctionnalité de scan de code-barres dès l'ouverture de l'application.

AfterScan

Cet événement permet de renvoyer le résultat obtenu par le scan du code-barres ; par exemple, cette application :

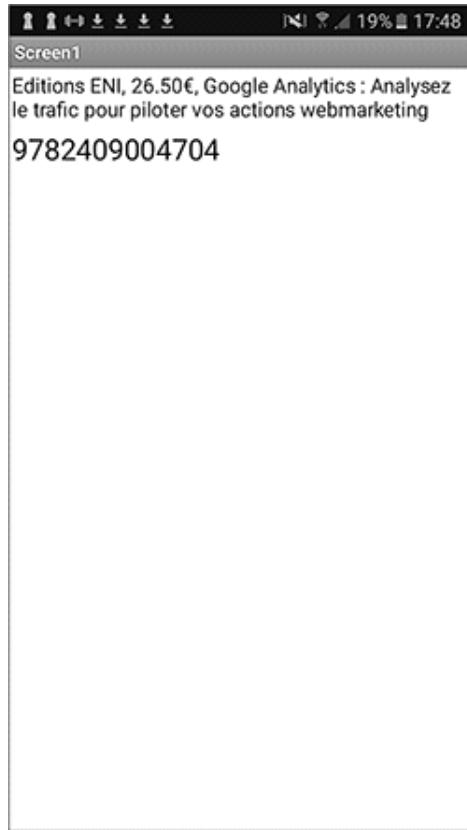


va renvoyer le numéro du code-barres. Il faudra utiliser une base de données pour faire correspondre les numéros envoyés avec des références.

Par exemple :



Dans le cas de cette application, si le résultat obtenu par le scan du code-barres correspond au numéro indiqué alors des informations spécifiques à celui-ci seront affichées :



Ici, il s'agit du numéro ISBN du livre dont le titre apparaît.

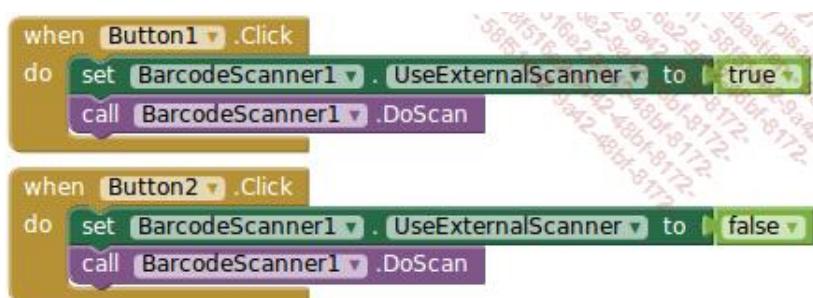
Les propriétés

Result

Permet d'obtenir la valeur renvoyée par le scan effectué.

UseExternalScanner

Par défaut App Inventor 2 utilisera son propre programme de scan. Si vous souhaitez en utiliser un autre, il vous suffit de définir la valeur de **UseExternalScanner** à **false**. Le mieux pour se rendre compte de la différence entre le programme de scan d'App Inventor 2 et celui de votre téléphone est de tester les deux avec une application de ce type :



3. Clock

Il est difficile de présenter l'ensemble des fonctionnalités proposées par ce composant tellement il y en a. De plus, la

plupart d'entre elles ne sont quasiment jamais utilisés. Nous nous sommes donc focalisés sur les plus utiles.

La fonctionnalité de loin la plus utilisée est l'événement `Clock Timer`. Cette fonctionnalité permet de faire réagir votre application à chaque fois qu'un intervalle de temps (défini par défaut sur 1000 ms) est atteint, par exemple si nous réalisons l'application suivante :



Alors, notre application va afficher dans une zone de texte, l'heure qu'il est (`FormatTime`), maintenant (`Now`) et le mettre à jour toutes les secondes (`Timer`).



`DurationToDays duration`

Permet de retourner la durée en jours écoulés depuis 1970 en fonction du nombre de secondes que vous indiquez.

`DurationToHours duration`

Permet de retourner la durée en heures écoulées depuis 1970 en fonction du nombre de secondes que vous indiquez.

`DurationToMinutes duration`

Permet de retourner la durée en minutes écoulées depuis 1970 en fonction du nombre de secondes que vous indiquez.

DurationToSeconds duration

Permet de retourner la durée en secondes éoulées depuis 1970 en fonction du nombre de secondes que vous indiquez.

DurationToWeeks duration

Permet de retourner le nombre de semaines éoulées depuis 1970. Pour cela vous aurez besoin d'indiquer le nombre de millisecondes éoulées depuis. Par exemple, en réalisant une application telle que :



Cela vous renverra la valeur 2 450, soit 2 450 semaines éoulées depuis 1970.

FormatDate instant pattern

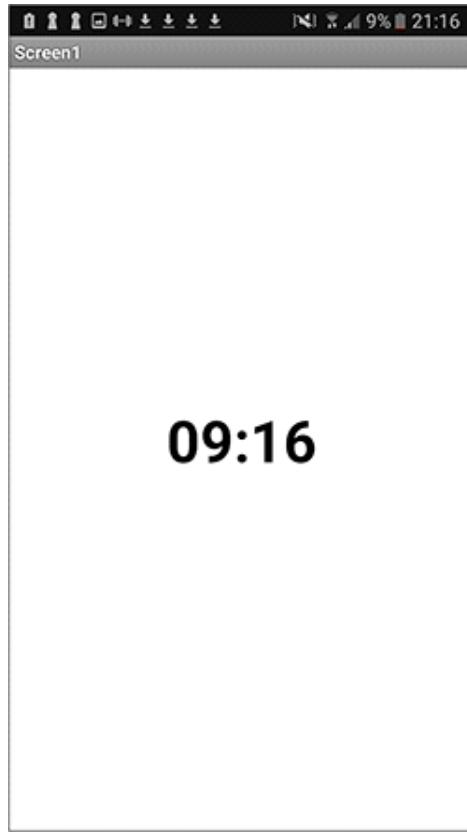
Permet d'afficher la date en fonction du format indiqué.

FormatDateTime instant pattern

Cette fonction permet d'afficher l'heure en fonction du format désiré. Par exemple, en écrivant le code suivant :



vous obtiendrez le résultat suivant :



FormatTimeinstant

Permet de retourner une donnée du composant Clock sous la forme d'une heure. Par exemple, 21:12:31.

GetMillisinstant

Permet de retourner le nombre de millisecondes écoulées entre la date que vous indiquez et l'année 1970.

Hour instant

Permet d'afficher l'heure. Par exemple 21 pour pour 21 heures 09.

MakeInstantFromMillis millis

Permet d'afficher le temps en format informatique depuis 1970.

Minuteinstant

Permet de faire afficher les minutes de l'heure actuelle. Par exemple, 37 s'il est 18h37.

Monthinstant

Permet de faire afficher le mois courant sous la forme d'un chiffre. Par exemple : le 1 pour janvier, 2 pour février, 3 pour mars, 4 pour avril, 5 pour mai ...

MonthNameinstant

Permet d'afficher le mois courant. Par exemple : janvier, février, mars, avril, mai, juin, juillet, août, septembre...

Now

Permet d'interroger le composant Clock au moment même de l'exécution de la fonction.

Secondinstant

Permet de faire afficher le nombre de secondes actuelles en train de s'écouler, de 0 à 59.

System time

Permet de vous remonter en millisecondes, le temps, l'heure de l'horloge interne de votre smartphone depuis 1970.

Weekdayinstant

Permet de remonter le jour de la semaine sous forme d'un chiffre. Par exemple : le 1 pour le dimanche, le 2 pour le lundi, le 3 pour le mardi ...

WeekdayNameinstant

Permet de remonter le jour de la semaine. Par exemple : lundi, mardi, mercredi...

Yearinstant

Permet de renvoyer l'année en cours.

4. GyroscopeSensor

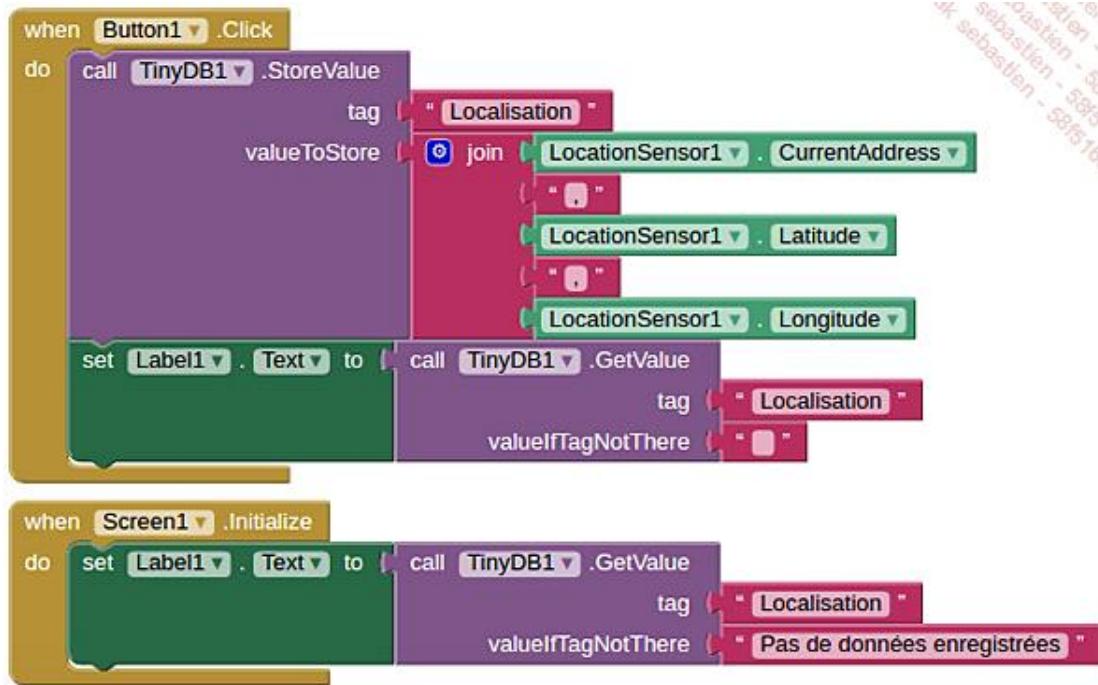
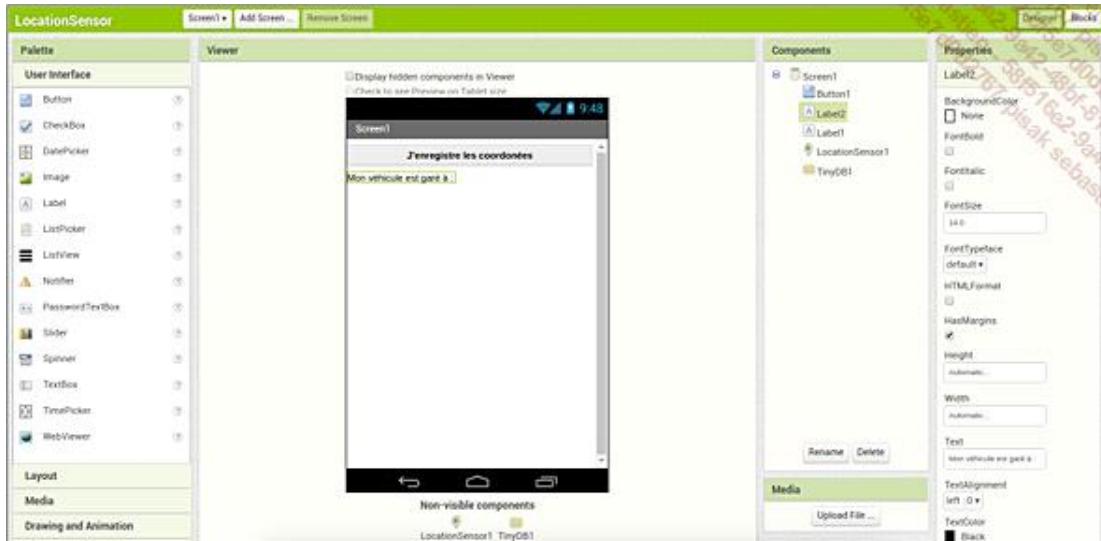
Permet de connaître l'orientation de votre téléphone en fonction de coordonnées x, y et z.

5. LocationSensor

Afin d'illustrer l'utilisation de ce composant, nous pouvons nous inspirer de ce tutoriel officiel d'App Inventor 2 : <http://appinventor.mit.edu/explore/ai2/android-wheres-my-car.html>. Celui-ci consiste à mémoriser dans son appareil l'endroit où on a garé son véhicule. Pour ce faire il suffit de créer une application possédant :

- un composant GPS (localisation de l'appareil)
- un tinydb (sauvegarde en mémoire de la position)
- un bouton
- deux libellés

Notre application ressemblera à :



Cette application va vous permettre d'enregistrer les coordonnées de votre véhicule en un clic au moment où vous appuierez sur le bouton. Ces coordonnées seront alors enregistrées dans une base de données et seront affichées en même temps. Vous pouvez alors éteindre votre appareil. Dès que vous lancerez à nouveau l'application, celle-ci vous affichera les coordonnées qu'elle avait enregistrées dans sa base.

Événements

LocationChanged(number latitude, number longitude, number altitude)

Cette fonction permet de remonter à l'appareil ses nouvelles coordonnées de localisation.

StatusChanged(text provider, text status)

Cette fonction est appelée lorsque le statut du fournisseur des coordonnées de localisation change.

Méthodes

number LatitudeFromAddress(text locationName)

Permet d'obtenir les coordonnées latitude de l'adresse donnée.

number LongitudeFromAddress(text locationName)

Permet d'obtenir les coordonnées longitude de l'adresse donnée.

LocationChanged

Cet événement vous indiquera les nouvelles coordonnées GPS indiquées.

Accuracy

Cette propriété vous indique le niveau de précision en mètres du composant **LocationSensor**.

Altitude

Cette propriété vous renvoie l'altitude de votre appareil si disponible.

AvailableProviders

Vous renvoie la liste des fournisseurs de services de localisation.

CurrentAddress

L'adresse physique à laquelle votre appareil est en train d'être géolocalisé.

Par exemple : 2, rue de la Paix 75000 Paris.

DistanceInterval

Cette propriété permet d'indiquer l'intervalle de distance à partir duquel vous souhaitez mettre à jour vos données de localisation.

Enabled

Permet d'activer le composant localisation.

HasAccuracy

Si cette option est activée, et si votre appareil en est équipé, alors il renverra le niveau de précision en mètre.

HasAltitude

Si cette propriété est activée et qu'App Inventor 2 récupère la donnée, alors il affichera l'altitude de l'appareil.

HasLongitudeLatitude

Si cette option est activée, et si votre appareil en est équipé, alors il renverra la longitude et la latitude de l'appareil.

Latitude

Indique la latitude de l'appareil.

Longitude

Indique la longitude de l'appareil.

ProviderLocked

Permet de figer l'utilisation d'un opérateur pour obtenir la localisation.

ProviderName

Permet de renvoyer le nom du fournisseur actuel des coordonnées de localisation.

TimeInterval

Permet d'indiquer le temps à partir duquel vous souhaitez que l'appareil interroge le composant de localisation.

6. NearField

"Avant de vous arracher les cheveux", sachez que l'utilisation de vos capteurs NFC ne marchera pas avec AI 2 si ces derniers sont vides. Voici à titre d'exemple ce qu'est un capteur NFC, nous avons achetés ceux-ci sur <https://whiztags.com> :



Si tel est le cas, vous verrez le message suivant s'afficher lorsque vous utiliserez votre application AI 2 avec ce capteur :



Afin de pouvoir utiliser ces tags NFC avec AI 2, il faut y avoir injecté des données au préalable. Pour ce faire, utilisez des applications telles que NFC Tools que vous trouverez sur le PlayStore :



- Dès que l'application est lancée, cliquez sur l'onglet **Ecrire**, puis sur **Ajouter un enregistrement - Texte** et écrivez ce que vous souhaitez (ici nous avons choisi d'écrire **ai2**) :



- Afin de pouvoir écrire sur le tag NFC, cliquez sur **Ecrire / X BYTES**. L’application va vous demander alors d’approcher votre tag NFC afin de pouvoir écrire dessus :



Lorsque l’écriture est effective, un message de confirmation s’affiche :



Votre tag NFC peut désormais être utilisé pour App Inventor 2 ; nous allons pouvoir présenter les différentes fonctionnalités de ce composant.

TagRead

Cet événement permet d'indiquer que le tag NFC a été lu et vous permet de lui renvoyer son message, par exemple une application telle que celle-ci :



Vous renverra, dans notre cas de figure, l'information **ai2**. En général, les tags NFC sont plutôt utilisés pour renvoyer vers un lien ou un type de contenu textuel (carte de visite virtuelle...).

TagWritten

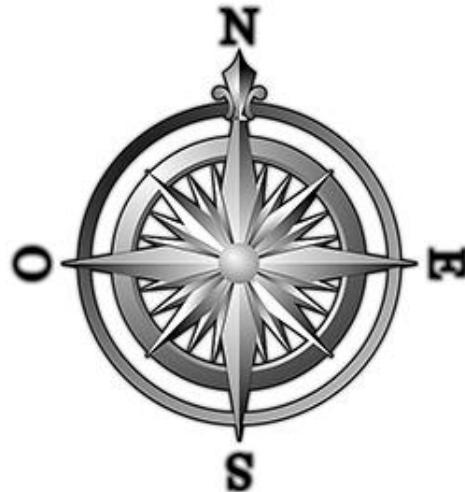
Cet événement se déclenche une fois que vous avez écrit des données sur le capteur NFC.

7. OrientationSensor

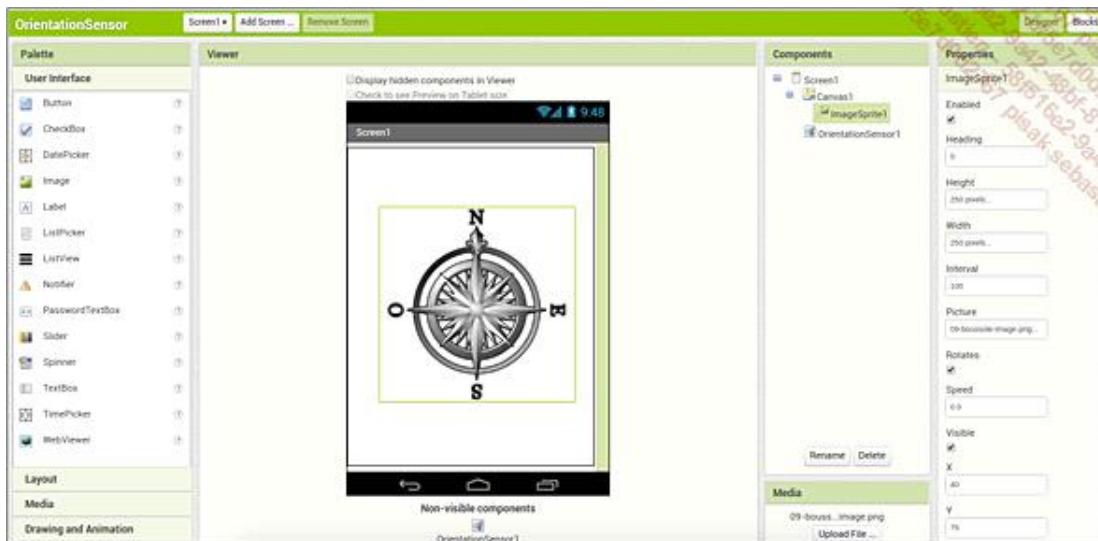
Comme son nom l'indique, ce composant permet d'indiquer l'orientation de votre appareil. L'idéal afin de présenter l'ensemble des fonctionnalités proposées par ce composant est de réaliser une application. Nous avons fait le choix de présenter ici la réalisation d'une boussole. Nous sommes partis pour cela de l'exemple suivant : <https://www.youtube.com/watch?v=vexhogWvxg4> réalisé par Adel Kassah

Pour réaliser ce tutoriel, vous aurez simplement besoin de l'image d'une boussole ou tout simplement d'une image montrant sous la forme d'une étoile les points cardinaux.

Dans notre cas de figure, nous avons utilisé l'image suivante : <https://pixabay.com/fr/boussole-rose-des-vents-au-nord-150121/>



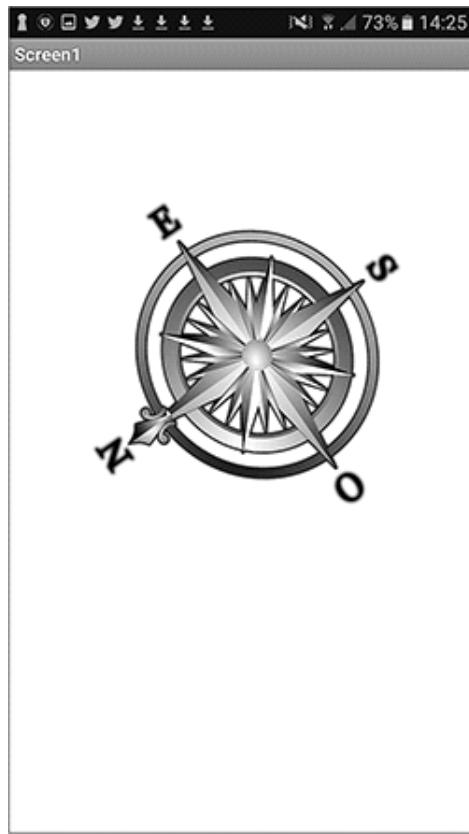
Afin de pouvoir réaliser cette application de boussole nous allons avoir besoin de mettre en place un canvas ainsi qu'une image sprite (qui sera ici l'image de la boussole). En effet, cette image doit bouger en fonction de l'orientation du téléphone :



Concernant le code, nous allons utiliser le code suivant :

```
when OrientationSensor1 .OrientationChanged
    azimuth pitch roll
do set ImageSprite1 . Heading to get azimuth
```

Dans cet exemple, l'orientation de l'image sera celle renvoyée par la valeur **azimuth** qui correspond à l'orientation où **0** correspond au nord, **90** à l'est et ainsi de suite.



OrientationChanged

Comme nous venons de le voir dans cette application, cet événement permet d'indiquer que l'appareil a changé d'orientation. Cet événement est composé de trois variables :

- **azimut** : indique l'orientation de votre appareil de 0 à 360 degrés.
- **pitch** : indique l'inclinaison de votre appareil. Par exemple, si celui-ci est posé à plat sur une table, cette valeur sera proche de 0. Si vous le mettez à la verticale, il aura une valeur proche de -90 et si vous inclinez vers le bas, une valeur proche de 90.
- **roll** : indique le niveau de "rotation" de votre appareil. Par exemple, si vous le couchez sur l'arrête, il aura un niveau proche de 90, si vous le laissez à plat il sera proche de 0.

Propriétés

Les propriétés du composant OrientationSensor sont assez explicites à l'exception de deux : **Magnitude** et **Angle**.

Magnitude

Cette propriété indique également l'inclinaison de l'appareil, et ce sur une échelle de 0 à 1.

Angle

Cette propriété indique la direction vers laquelle l'appareil est incliné.

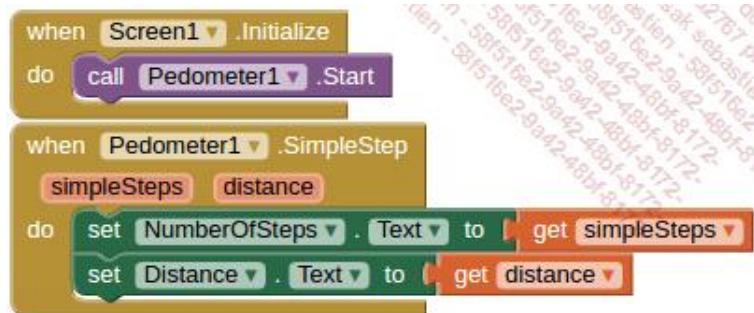
8. Pedometer

Le composant Pedometer est en fait une variante du composant Accelerometer. Celui-ci est configuré de telle manière qu'il est possible de l'utiliser pour estimer le nombre de pas parcourus. Pour en savoir plus sur le lien entre les composants Accelerometer et Pedometer, nous vous conseillons de vous référer à ce très bon article : <http://appinventor.mit.edu/explore/blogs/karen/2016/07.html>

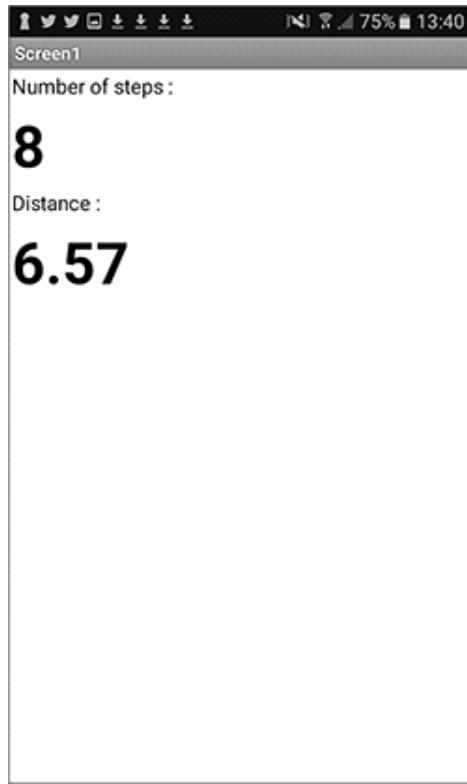
SimpleStep

Permet de mesurer le nombre de pas effectués et la distance parcourue exprimée en "mètres" (d'après nos calculs, 1 pas pour l'application correspond à une distance de 0,69 : nous pensons donc qu'il s'agit de mètres).

Afin de pouvoir faire fonctionner cet événement, il vous faudra utiliser la méthode Play, tout comme dans l'exemple ci-dessous :



Ce qui donnera :



WalkStep

L'événement WalkStep est similaire au précédent à la différence que celui-ci ne va pas comptabiliser les pas si l'appareil ne bouge pas sur toutes ces dimensions. En effet lorsque vous marchez normalement, vos jambes avancent mais également effectuent de légers mouvements vers la droite et la gauche. Mettre en place un tel

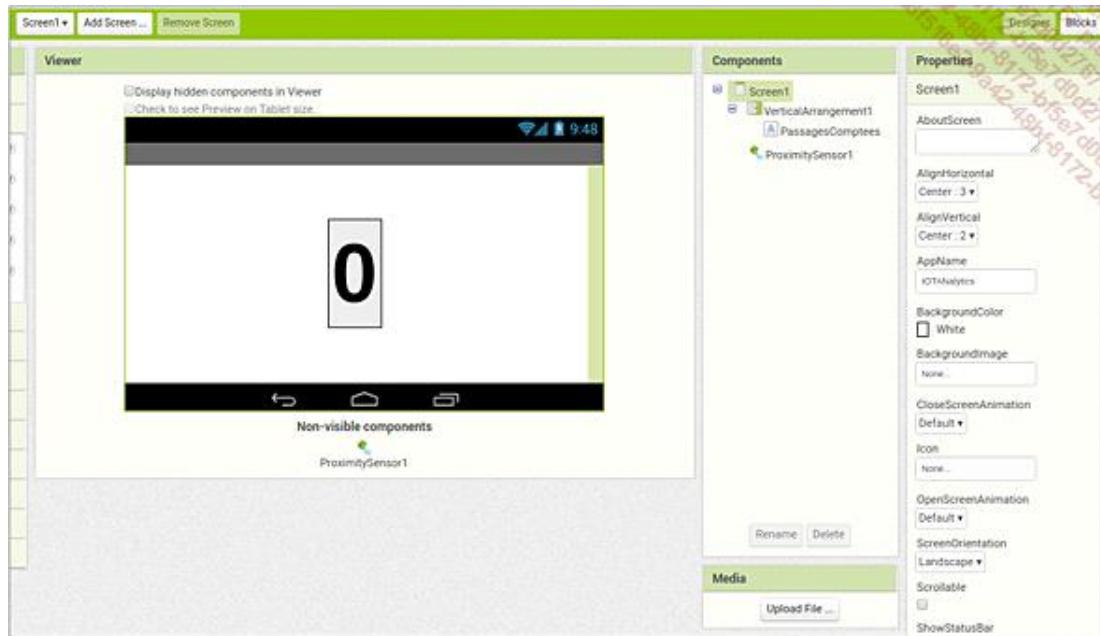
système de calcul permet d'avoir une comptabilisation plus juste. La mise en place du code reste la même que dans l'exemple précédent.

L'ensemble des fonctionnalités restantes étant assez explicites, nous avons décidé de ne pas les détailler.

9. ProximitySensor

ProximityChanged

Cet événement permet d'indiquer que le capteur a rencontré un objet. Afin de l'illustrer, le mieux est de vous présenter une application. L'objectif de cette application est de montrer le nombre de passages effectués devant le capteur :



Au niveau du code, voici comment est composée l'application :

Nous démarrons le nombre de passages à 0. Lorsque le capteur réagit et qu'il indique la valeur **0** alors nous incrémentons de **1** la variable de départ et décidons d'afficher le nouveau nombre de passages.

Les fonctions des composants Social

De nombreux événements ont déjà été présentés au sein des sous-parties précédentes, aussi allons-nous nous concentrer uniquement sur les fonctionnalités qui n'ont pas encore été présentées.

1. ContactPicker

Événements

Les événements supportés par ContactPicker ont déjà été évoqués par le passé, nous n'allons pas les détailler à nouveau mais sachez que ce composant supporte les événements suivants : AfterPicking, BeforePicking, GotFocus, LostFocus, TouchDown, TouchUp.

Méthodes

Open

Cette fonctionnalité va ouvrir le carnet de contacts de votre smartphone.

Exemple avec l'application suivante :



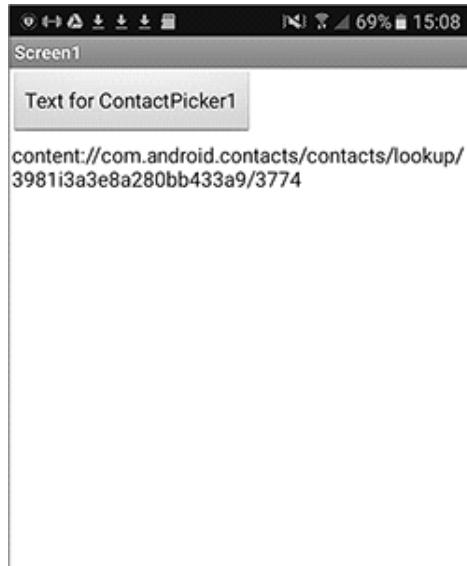
Un simple clic sur le bouton **ContactPicker** vous affichera la liste des contacts de votre téléphone.

ViewContact uri

L'URI contact d'Android correspond à l'emplacement sur votre téléphone du contact que vous avez sélectionné. Pour pouvoir afficher l'URI d'un contact en particulier vous pouvez, par exemple, procéder de la manière suivante :



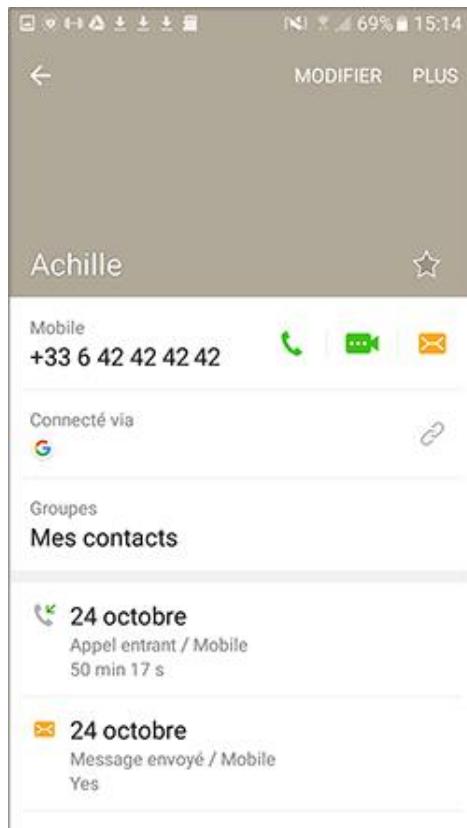
Lorsque nous appuierons sur le bouton ContactPicker, nous obtiendrons donc l'URI du contact sur notre appareil :



Ainsi, si nous réalisons une application de ce type :



Un simple clic sur le bouton **ContactPicker** nous permettra d'accéder à la fiche contact sur votre mobile :



Propriétés

Les fonctionnalités offertes par le composant ContactPicker vous permettent de récupérer les principales informations telles que le nom du contact, son adresse e-mail, son numéro de téléphone... Vous pourriez par exemple imaginer une application telle que celle-ci pour sélectionner ces informations.



2. EmailPicker

Le composant EmailPicker ne possède que des événements, méthodes et fonctionnalités de propriétés déjà évoqués dans ce chapitre. Nous avons donc pris le parti de ne pas les détailler à nouveau.

3. PhoneCall

MakePhoneCall

Cette méthode permet d'appeler le numéro de téléphone que vous avez indiqué ; par exemple, l'application ci-dessous va appeler le numéro que vous avez défini lorsque vous appuierez sur le bouton :



IncomingCallAnswered

Cet événement permet d'afficher le numéro de téléphone qui est en train de vous appeler et pour lequel vous avez pris l'appel. Afin de tester cette fonctionnalité.

PhoneCallEnded

Cet événement est très similaire au précédent. Il vous permet de savoir qu'un terme a été mis à la communication avec le numéro appelant et vous indique grâce à la variable statuts s'il s'agit d'un appel pour lequel vous avez décroché (statut = 2), que vous avez refusé (statut = 1) ou que votre interlocuteur a raccroché (statut = 3).

PhoneCallStarted

Cet événement permet d'indiquer qu'un appel est soit entrant soit sortant grâce à l'utilisation d'une variable. Si l'appel est entrant la variable prend le statut 1, si c'est votre appareil qui appelle, celui-ci prend le statut 2.

PhoneNumber

Cette propriété permet d'indiquer le numéro de téléphone que vous souhaitez composer.

4. PhoneNumberPicker

Nous n'allons pas détailler les fonctions de ce composant car elles ont été intégralement citées pour d'autres. Vous pouvez donc vous y référer.

5. Sharing

Le composant **Sharing** possède assez peu de fonctionnalités. Détaillons ici les méthodes que vous pourrez utiliser.

`ShareFile file`

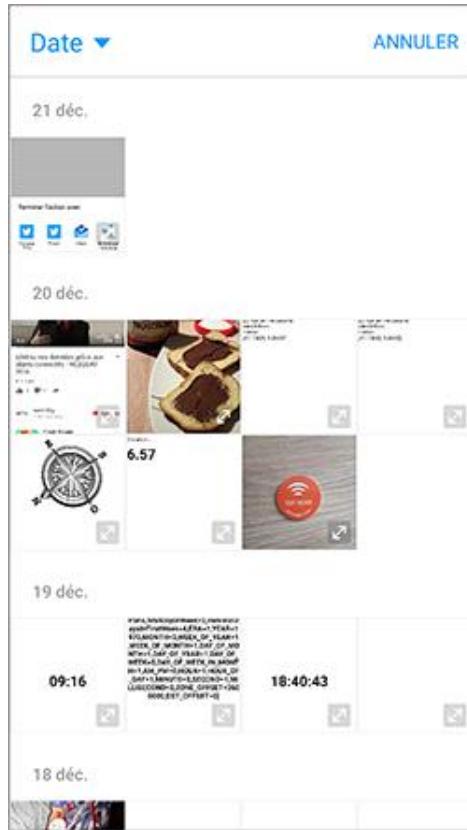
Permet de partager un fichier en fonction du chemin d'accès que vous indiquez. Pour ce faire, nous allons créer une application qui comprend deux composants :

- un composant **Sharing**
- un composant **ImagePicker** (ce dernier va faciliter l'insertion du chemin d'accès au fichier)

Voici à quoi va ressembler notre application :



Lorsque nous allons cliquer sur le bouton **ImagePicker**, l'application va nous permettre de sélectionner une image directement dans notre appareil :

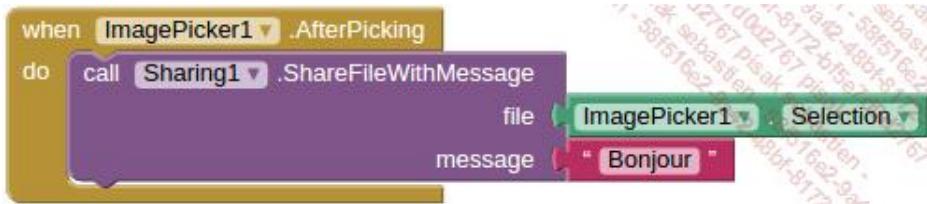


Une fois cette image sélectionnée, le composant Sharing va directement interagir pour nous présenter toutes les applications de notre appareil qui permettent de faire partager cette dernière :

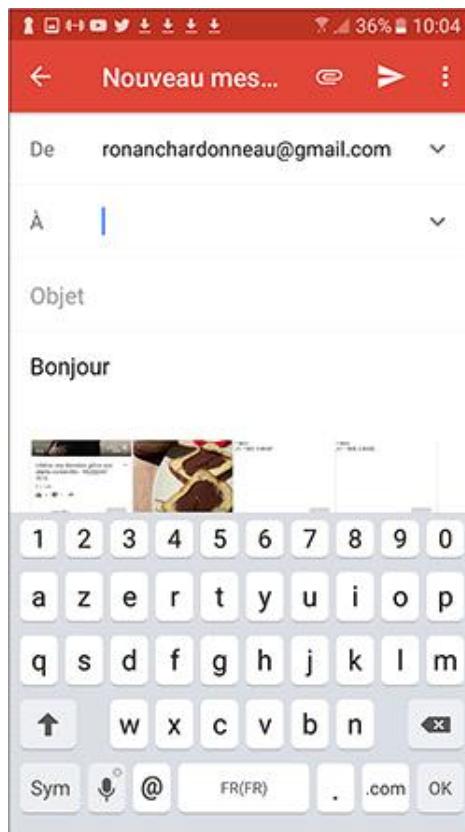


`ShareFileWithMessage file message`

Il s'agit d'une méthode similaire à la précédente, à l'exception que celle-ci inclura un message ; par exemple, pour l'application suivante :



si nous choisissons de partager le message via Gmail :



alors celui-ci aura directement comme corps de message le mot Bonjour.

ShareMessage message

Cette fonctionnalité permet de partager un message, tout simplement, avec le contenu que vous lui indiquez. Il s'agit du même exemple que précédemment mais sans rajouter de pièce jointe.

6. Texting

Ce composant fonctionne de la même manière que lorsque vous envoyez un SMS à quelqu'un de votre carnet de contacts. Par exemple, l'application ci-dessous :



va vous permettre d'envoyer le message indiqué ci-dessus au numéro indiqué lorsque vous appuierez sur un bouton que vous avez rajouté à votre application.

GoogleVoiceEnabled

Cette fonctionnalité, lorsqu'elle est définie sur true, permet d'utiliser votre connexion wi-fi pour envoyer le message via Google Voice (Google Hangout).

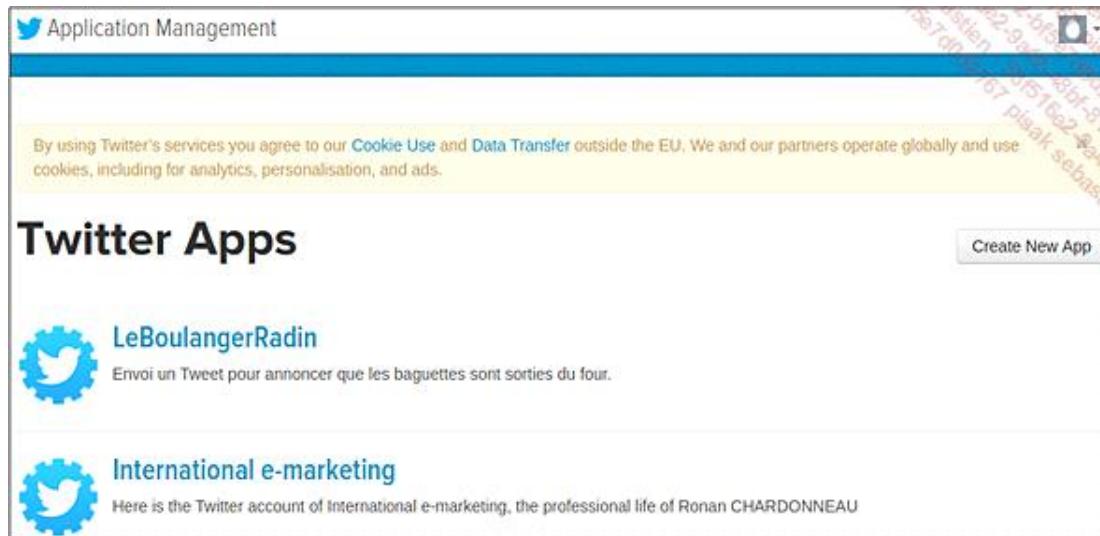
ReceivingEnabled

Permet d'indiquer si vous souhaitez être prévenu ou non de la réception des messages. Lorsque vous attribuez la valeur **1**, aucun message ne sera reçu. Si la propriété est définie sur **2**, alors les messages seront reçus en tâche de fond. Si elle est définie sur **3**, alors vous pourrez les utiliser à l'intérieur même de l'application.

7. Twitter

Utiliser les fonctionnalités de Twitter dans App Inventor 2 n'est pas très compliqué, en revanche des problèmes au niveau des autorisations avec la plateforme Twitter peuvent arriver. Le plus important à savoir ici c'est que pour utiliser ce composant, vous devrez créer un accès développeur sur Twitter.

- Pour ce faire, rendez-vous sur <https://apps.twitter.com/>.
- Une fois sur la page, cliquez alors sur **Create New App** :



- Remplissez ensuite le formulaire suivant :

Create an application

Application Details

Name *

SimpleTwitter

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description *

Permet simplement de tester les fonctionnalités d'App Inventor 2

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website *

http://ronan-chardonneau.fr

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in the source attribution for tweets created by your application and will be shown in user-facing authorization screens. (If you don't have a URL yet, just put a placeholder here but remember to change it later.)

Callback URL

http://ronan-chardonneau.fr

Where should we return after successfully authenticating? OAuth 2.0 applications should explicitly specify their auth_callback URL on the request token step, regardless of the value given here. To restrict your application from using callbacks, leave this field blank.

Developer Agreement

Yes, I have read and agree to the [Twitter Developer Agreement](#).

[Create your Twitter application](#)

N'oubliez pas de remplir l'information **Callback URL**, sinon vous ne pourrez pas utiliser votre application avec ce composant.

→ Après avoir cliqué sur **Create your Twitter Application**, cliquez sur l'onglet **Keys and Access Tokens** :

The screenshot shows the Twitter Application Management interface. At the top, there is a banner with legal text about cookie use and data transfer. Below the banner, the application name "SimpleTwitterAI2" is displayed. The "Settings" tab is selected. Under "Application Settings", there are fields for "Consumer Key (API Key)" and "Consumer Secret (API Secret)", both of which are redacted. The "Access Level" is set to "Read and write (modify app permissions)". The "Owner" and "Owner ID" fields are also redacted. In the "Application Actions" section at the bottom, there are buttons for "Regenerate Consumer Key and Secret" and "Change App Permissions".

Vous disposez désormais des accès qui vous seront nécessaires pour faire fonctionner l'ensemble des fonctionnalités.

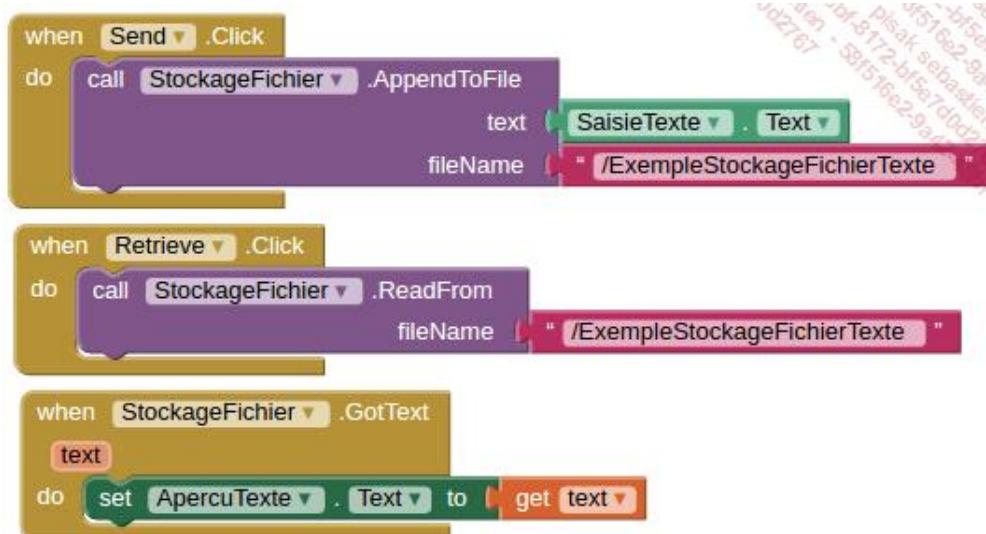
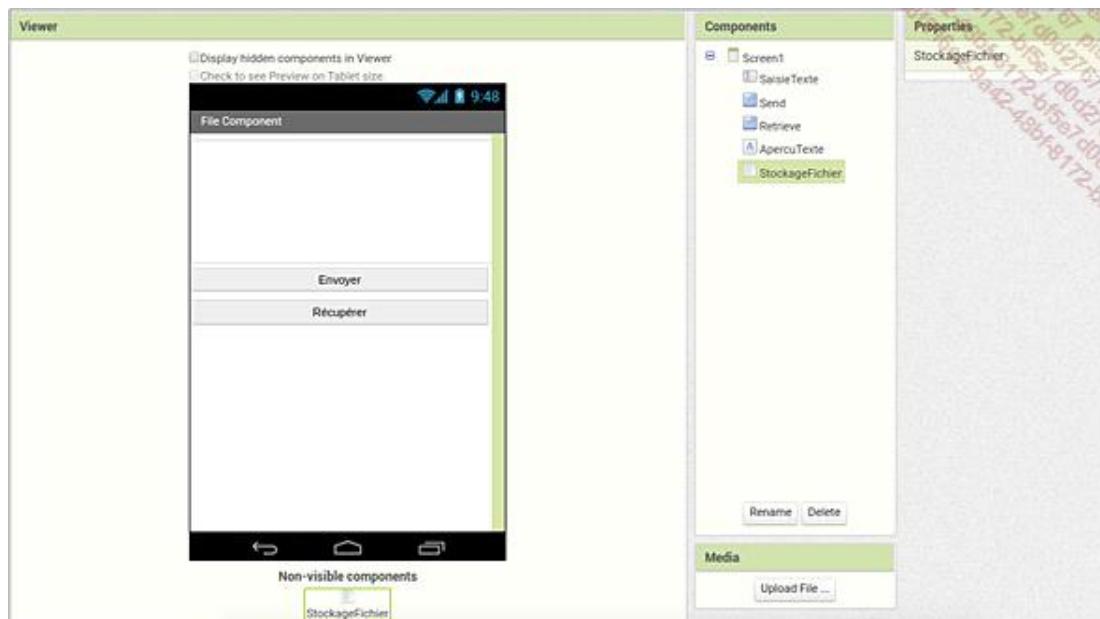
Les fonctions des composants Storage

Les fonctions de stockage, comme leur nom l'indique, vont vous permettre de garder en mémoire des données pour pouvoir les restituer par la suite. Découvrons les différentes solutions de bases de données offertes par App Inventor 2.

1. File

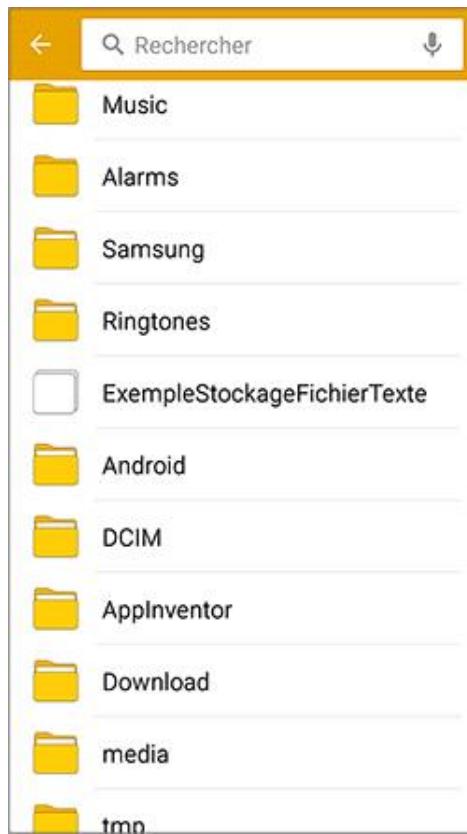
Différentes fonctionnalités vont être proposées par l'intermédiaire de ce composant. Ce qu'il faut surtout retenir ici, c'est que le fichier doit être sauvegardé avec un slash / devant afin que celui-ci soit enregistré au sein de la carte SD de votre téléphone et de le faire terminer par une extension en .txt pour que celui-ci soit considéré comme un fichier texte.

Afin de pouvoir tester ce composant, vous pouvez facilement répliquer le programme suivant :



Que permet de faire cette application ?

Lorsque l'utilisateur saisit du texte dans le composant intitulé **SaisieTexte** et appuie sur le bouton **Envoyer**, cela écrit dans un fichier appelé **ExempleStockageFichierTexte** qui se trouve sur la carte SD du téléphone.



En cliquant sur **Récupérer**, le contenu du fichier sera affiché à l'écran de l'utilisateur.

2. FusiontablesControl

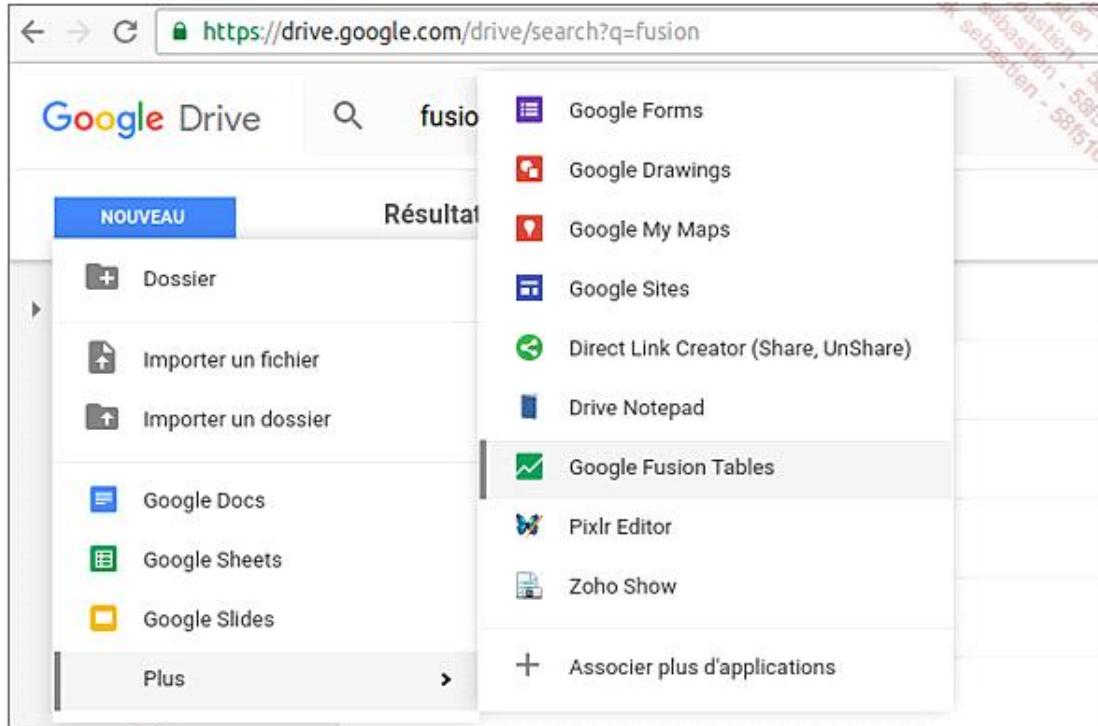
Les tables de fusion sont un type de base de données de Google. Cela permet notamment de les intégrer facilement à un site Internet.

App Inventor 2 propose un très bon tutoriel en anglais (<http://appinventor.mit.edu/explore/ai2/pizzaparty.html>) montrant la puissance de ce composant. Dans notre cas de figure nous allons réaliser un tutoriel un peu plus succinct afin de vous montrer les fonctionnalités principales :

- GotResult
- ForgetLogin
- GetRows
- GetRowsWithConditions
- InsertRow
- SendQuery

Notre tutoriel va consister à créer une application permettant d'indiquer via une application, qu'une personne est présente ou non à un endroit donné, en renvoyant le résultat sur un site Internet.

La première chose à savoir avec les tables de fusion, c'est qu'il s'agit d'un service de Google que vous trouverez directement dans Gooale Drive :



Comme vous pouvez le constater avec l'image ci-dessous, il s'agit ni plus ni moins que d'un simple tableur, un peu à la manière de Google Sheets :

A screenshot of a 'New Table' interface from Google Fusion Tables. The title 'New Table' is at the top. Below it, the text 'Edited at 10:32' is shown. The menu bar includes 'File', 'Edit', 'Tools', and 'Help'. The toolbar features 'Rows 1', 'Cards 1', and a 'Map of Location' button. A 'Filter' dropdown shows 'No filters applied'. Navigation buttons indicate '1-1 of 1'. The main area contains four columns: 'Text', 'Number', 'Location', and 'Date'. Each column has a single empty cell.

Ici, nous avons une base de données qui, par défaut, possède quatre en-têtes de colonne et quatre valeurs vides.

Afin de pouvoir utiliser les tables de fusion de Google, vous devez vous créer un compte. En effet, afin que votre base de données ne soit pas accessible de tous et surtout afin de garantir sa sécurité, il faut créer des accès. Pour ce faire, rendez-vous à la page suivante : <https://console.developers.google.com/apis/credentials> (documentation officielle : https://docs.google.com/document/d/1HifuZqz5xu0KPS-e4oUv-t-nQoUQ8VMNyh_y6OjZkc0/pub).

- Cliquez sur **Créer des identifiants** puis sur **Clé de compte de service**.

- Sélectionnez **App Engine default service account** puis l'option **P12** et cliquez sur **Créer**.

Une fois créé, vous allez obtenir un petit fichier qu'il faudra insérer dans votre application, dans les propriétés de votre composant :

- Une fois cela fait, vous devez renseigner diverses informations telles que l'adresse e-mail associée à votre compte Google Developer, en général celle-ci se termine par @appspot.gserviceaccount.com.
- Vous devez également renseigner la valeur ApiKey que vous trouverez dans le back-office du service de

Une fois l'ensemble des informations rempli dans les propriétés de votre table de fusion, vous devez définir les caractéristiques de votre table de fusion dans Google Drive. Pour notre exemple, nous souhaitons simplement avoir une colonne pouvant comporter du texte avec, pour en-tête, la valeur **Présent** :

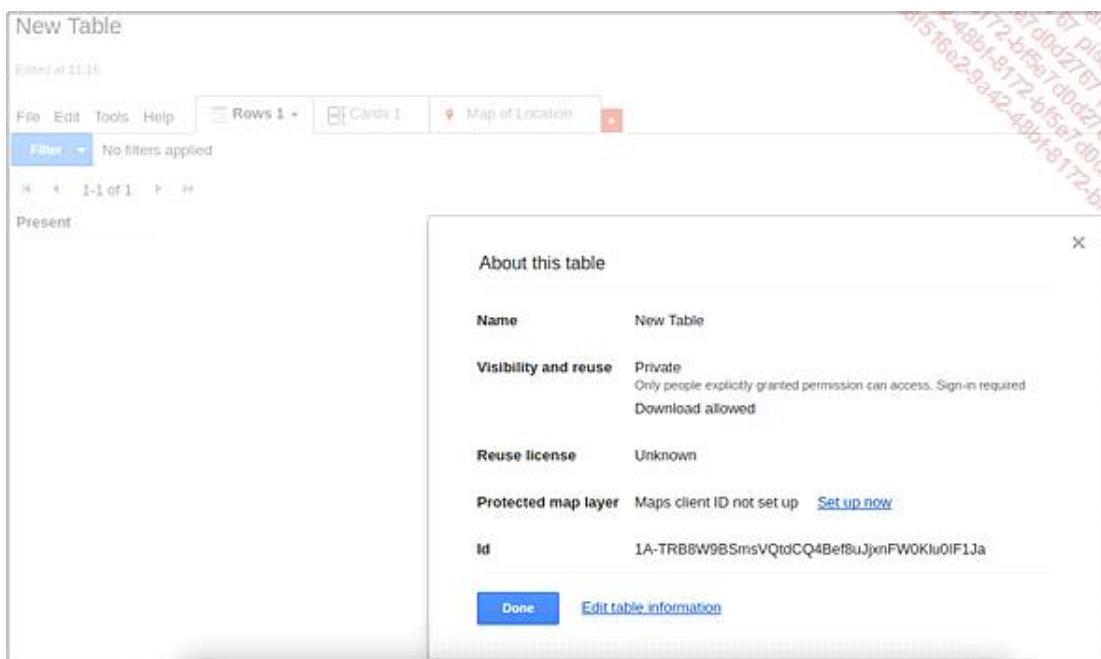
- Cliquez en haut de page sur **Edit**, puis sur **Change columns** et définissez alors les colonnes de la manière suivante :

Column name	Present
Description	
Type	Text
Format	None

- Cliquez sur **Save**.

Votre table est désormais créée ; cependant, pour pouvoir y injecter des données depuis App Inventor 2, vous aurez besoin de connaître l'identifiant de cette table.

- Pour ce faire, cliquez sur **File**, puis sur **About this table** :



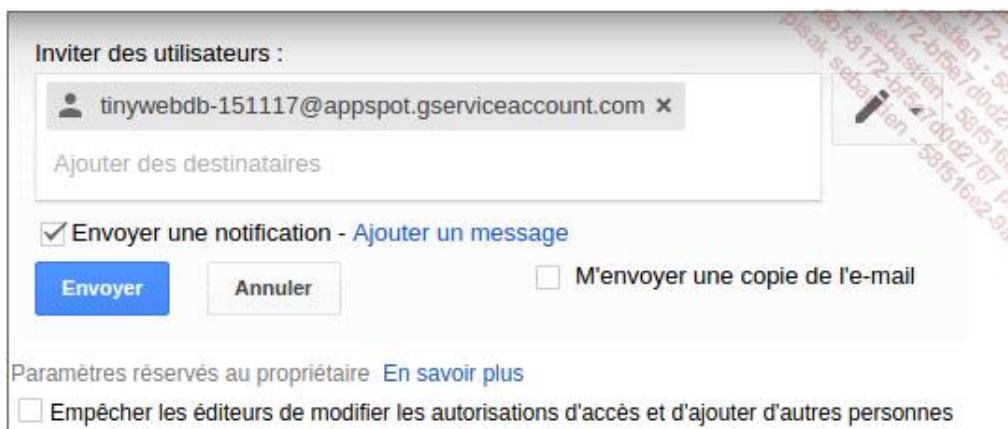
L'identifiant de cette table va vous permettre de commencer le tutoriel.

- Afin de pouvoir écrire dans la table, vous allez également avoir besoin de donner les accès à l'adresse e-mail indiquée dans l'application, pour ce faire cliquez sur **Publish** :



et cliquez sur **Change visibility**.

- Ajoutez alors comme utilisateur l'adresse renseignée :



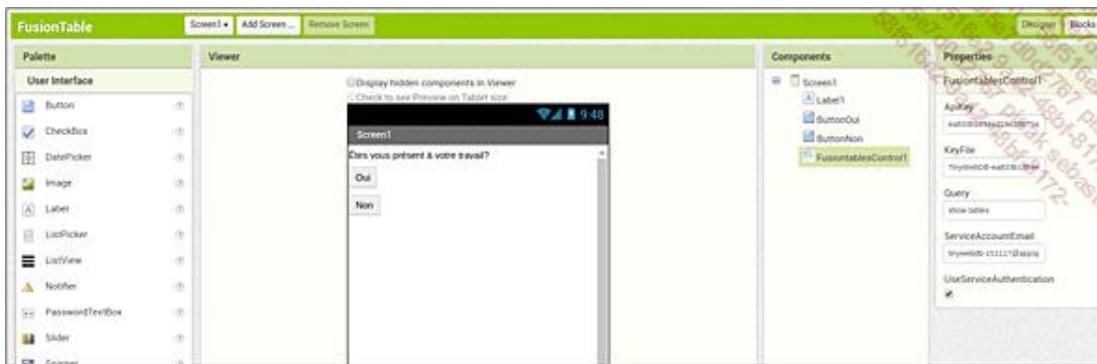
→ Cliquez sur **Envoyer**.

→ Activez également l'API Data Fusion Table dans Google Developer Console :

The screenshot shows the Google APIs Developer Console interface. In the top left, it says "Google APIs" and "TinyWebDB". On the right, there's a search bar. Below that, a navigation bar has "API" selected, followed by "Gestionnaire d'API", a back arrow pointing to "Fusion Tables API", and a blue "ACTIVER" button. To the right of the navigation bar, there's a large red watermark with a long string of characters: "d0d2761pisak sebastien - 1972-48bd-1972-6e23-0000000000000000". The main content area has a sidebar on the left with "Tableau de bord" (selected), "Bibliothèque", and "Identifiants". The main panel title is "À propos de cette API" with the subtitle "The Fusion Tables API lets you manage your data in Google Fusion Tables.". Below that is "Utilisation d'identifiants avec cette API" and "Accès aux données utilisateur avec OAuth 2.0". It explains how to create an OAuth 2.0 client ID and integrates it into the application.

→ Cliquez sur **ACTIVER**.

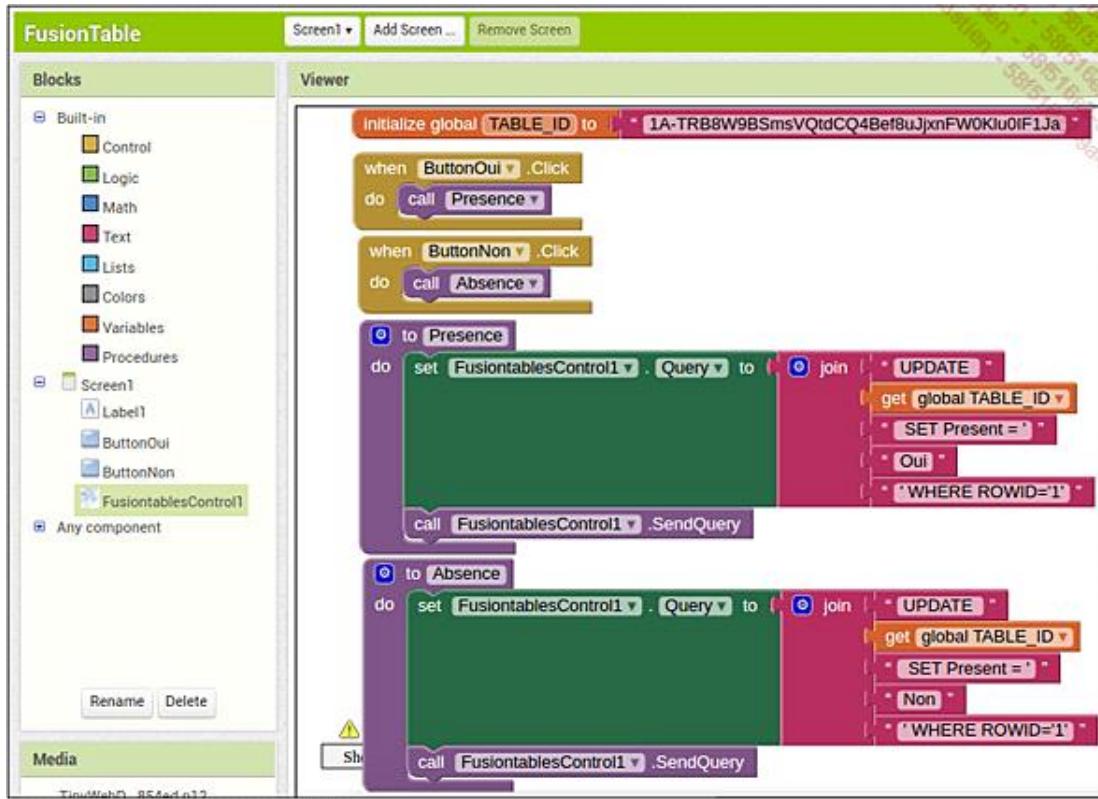
L'application que nous allons créer ressemblera à :



Elle est composée de :

- un libellé
- deux boutons
- une table de fusion

Voici le code :



Une variable, qui représente l'identifiant de votre table de fusion (l'insérer dans une variable nous permet de la réutiliser plus facilement par la suite).

Un bouton Oui qui, lorsqu'il est cliqué, permet de déclencher une requête SQL qui permet de mettre à jour la table de fusion qui a pour en-tête de colonne Present avec la valeur Oui. Une méthode du nom de SendQuery permet d'envoyer la requête aux serveurs de Google.

De la même manière, le bouton Non permet d'envoyer une requête similaire.

Ainsi, la base de données est mise à jour en fonction du bouton sur lequel l'utilisateur clique :

Nous avons à ce stade une application qui permet de dialoguer avec une base de données plus sécurisée et plus complexe que les autres composants proposés dans App Inventor 2 que sont TinyDB et TinyWebDB.

- Pour aller plus loin, nous pouvons également diffuser les données collectées en cliquant sur **Tools** puis sur **Publish**, en copiant-collant le code HTML indiqué et en l'intégrant sur le site de votre choix :

Publish

This table is private and will not be visible. [Change visibility](#)

Send a link in email or IM

<https://fusiontables.google.com/embedviz?viz=GVIZ&t=TABLE&q=select+col0+from+1A-1>

Paste HTML to embed in a website

<iframe width="500" height="300" scrolling="yes" frameborder="no" src="https://fusiontable

Width	<input type="text" value="500"/>	Height	<input type="text" value="300"/>
-------	----------------------------------	--------	----------------------------------

Nous venons de voir dans ce tutoriel les fonctions principales des tables de fusion à travers l'utilisation d'une requête SQL et la méthode `SendQuery`. Pour les autres fonctions :

- GotResult : cet événement permet de faire réagir votre application en fonction de la réponse obtenue lors de l'envoi de la requête.
 - ForgetLogin : permet de passer outre les authentifications.
 - GetRows : permet d'obtenir des lignes d'enregistrement dans la base de données. Dans notre cas de figure, nous avons effectué une requête plus complexe.
 - GetRowsWithConditions : idem que précédemment sauf que vous pouvez sélectionner les lignes de votre choix en fonction de conditions indiquées.
 - InsertRow : permet d'insérer des données dans la base à la manière d'**UPDATE** que nous avons pris ici comme exemple.

3. TinyDB

Le composant TinyDB comporte assez peu de fonctions. Heureusement d'ailleurs, car cela vous permettra d'appréhender très facilement la notion de base de données.

TinyDB est composé de cinq méthodes :

- `ClearAll`
 - `ClearTag`
 - `GetTags`
 - `GetValue`
 - `StoreValue`

Afin de pouvoir les présenter de façon logique, nous allons d'abord introduire la fonction `StoreValue`.

StoreValue

Cette méthode permet de stocker une valeur que vous pourrez réutiliser par la suite. Afin de l'illustrer, nous allons également présenter dans ce mini tutoriel, la fonction `GetValue` car si vous stockez une donnée, c'est probablement pour l'afficher par la suite.

Pour ce faire, nous allons créer une application sous cette forme :

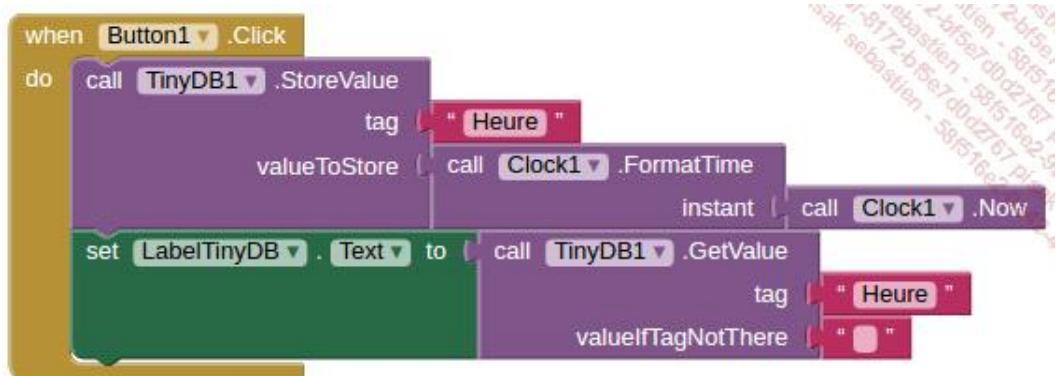


Cette application a pour but d'afficher l'heure à laquelle vous avez dernièrement appuyé sur le bouton.

Nous allons pour cela utiliser cinq composants :

- un bouton (**Button**)
- deux libellés (**Label**)
- un tinyDB
- un réveil (**Clock**)

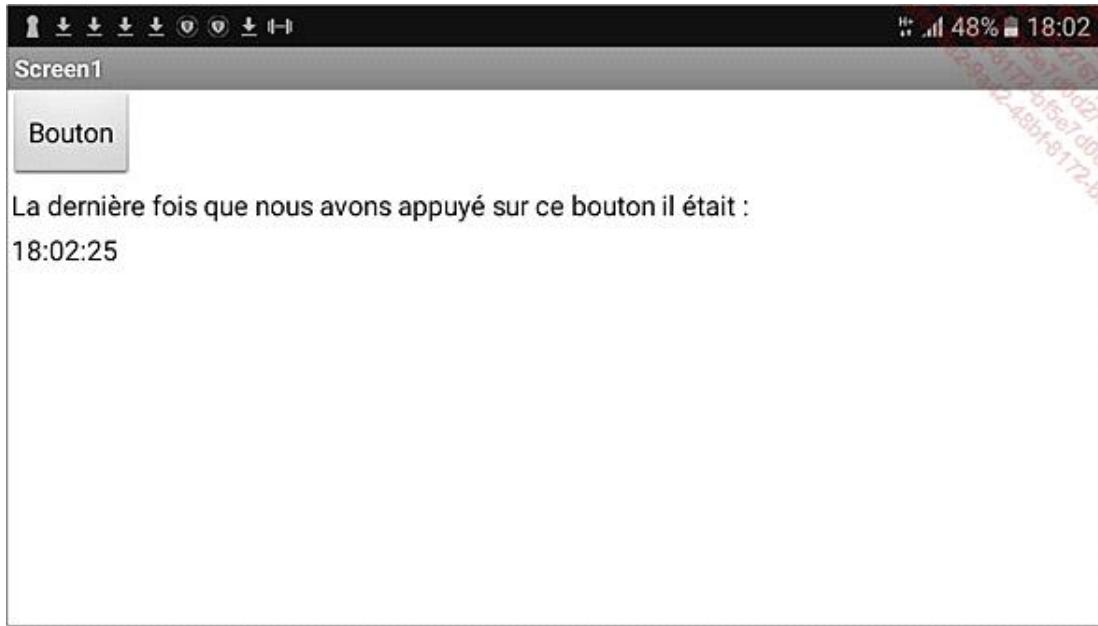
Le code de notre application va ressembler à cela :



Ici, nous avons défini le comportement suivant :

- Si on appuie sur le bouton
- Alors on stocke dans notre base de données, un en-tête de colonne nommé Heure ; cet en-tête prend la valeur de l'heure à laquelle nous avons appuyé sur le bouton.
- Une fois cette valeur enregistrée, le libellé, ici en vert, prend la valeur associée à ce libellé.

Voici concrètement à quoi va ressembler notre application une fois terminée :



Nous venons de voir les deux méthodes les plus importantes et les plus utilisées du composant TinyDB.

Nous avons fait le choix de ne pas présenter de tutoriels pour les trois autres méthodes car leur utilisation est plus rare :

- ClearAll : permet de supprimer tous les tags créés et leurs valeurs associées.
- ClearTag : permet de supprimer un tag spécifique et sa valeur associée.
- GetTags : permet de récupérer les noms des tags (et non toutes les valeurs des tags).

4. TinyWebDB

Comme son nom l'indique, le composant TinyWebDB communique avec le Web et possède naturellement plus de fonctionnalités que la fonctionnalité TinyDB.

Ce composant possède trois fonctions d'événements :

- GotValue
- ValueStored
- WebServiceError

et deux méthodes que sont :

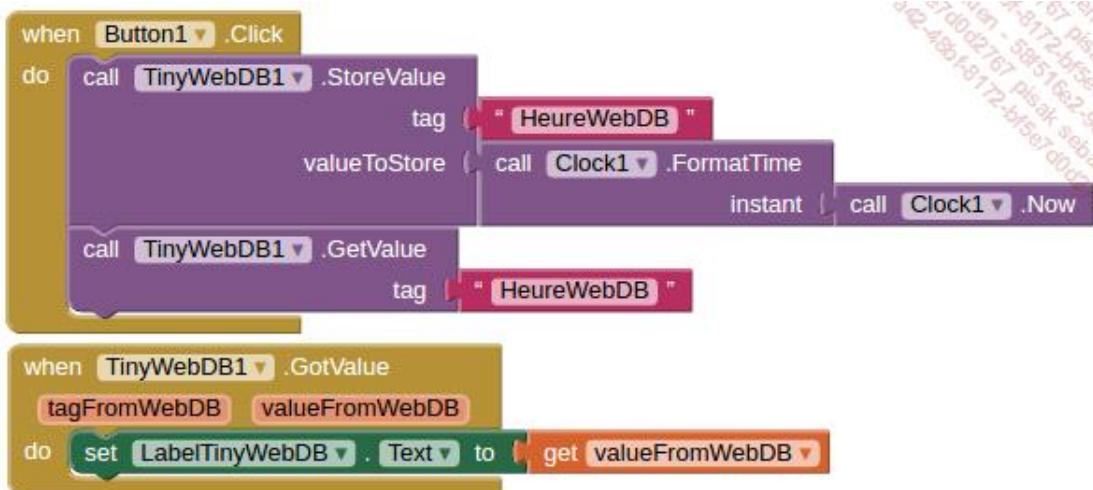
- GetValue
- StoreValue

Afin de pouvoir illustrer l'utilisation de ces fonctionnalités, nous allons reprendre la création d'une application similaire à celle que nous avons créée avec TinyDB. Voici son apparence dans la partie **Designer** :



Nous avons ici simplement substitué le composant TinyDB par TinyWebDB.

Pour la partie code :



Nous avons ici la méthode `StoreValue` qui va avoir exactement le même comportement que pour le composant TinyDB. La fonction `GetValue` de TinyWebDB va servir à aller chercher la valeur stockée dans le Cloud pour le tag `HeureWebDB`.

La fonction événement `GetValue` va quant à elle réagir si une valeur a pu être récupérée. Si c'est le cas, elle va afficher dans le libellé la variable `valueFromWebDB`.

Deux fonctions n'ont ici pas été vues, il s'agit de `ValueStored` et `WebServiceError`. La première va servir à indiquer ce que vous souhaitez faire lorsque l'enregistrement d'une valeur dans la base de données a fonctionné, la seconde sert à vous indiquer le message d'erreur s'il y en a eu un avec la base de données sur le Web.

Pour aller plus loin avec le composant TinyWebDB, sachez qu'App Inventor 2 facilite l'utilisation de TinyWebDB en vous proposant une base de données dans le Cloud directement embarquée. Cependant, sachez que si vous sortez de la plateforme d'App Inventor 2, par exemple en publifiant votre application sur le Play Store, vous ne pourrez utiliser ce composant et devrez trouver un endroit où héberger cette base de données. Pour ce faire, vous pouvez suivre le tutoriel officiel suivant en anglais : <http://appinventor.mit.edu/explore/content/custom-tinywebdb-service.html>

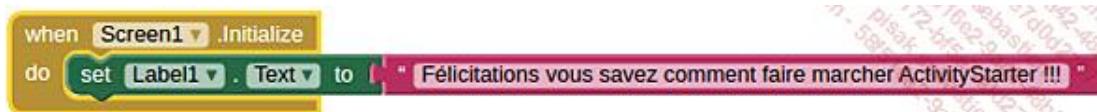
Les fonctions des composants Connectivity

1. ActivityStarter

Le composant **ActivityStarter** permet de lancer d'autres applications de votre appareil à partir d'App Inventor 2. Vous pouvez également émettre et recevoir des données depuis ces autres applications.

Afin de l'illustrer, nous allons créer deux applications, la première va afficher une phrase, la seconde (qui utilisera le composant **ActivityStarter**) va lancer la première.

Voici le code de notre première application :

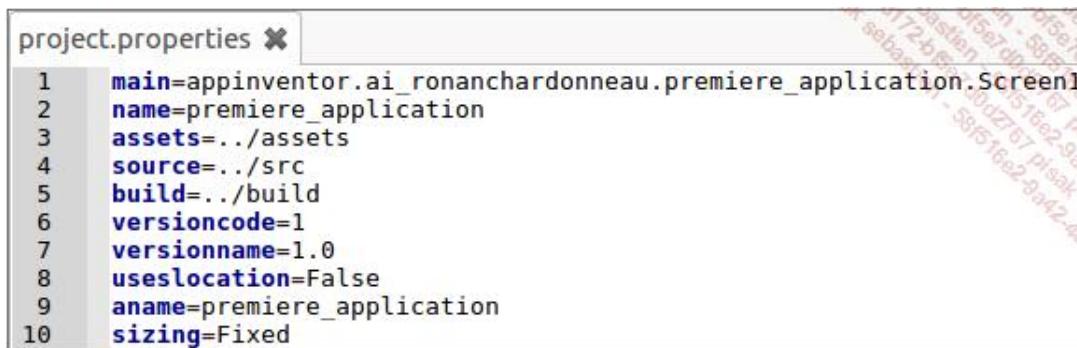


Comme vous pouvez le voir, cette application n'est là que pour afficher un message.

- Installez cette application sur votre téléphone et téléchargez également cette application au format .aia sur votre ordinateur.
- Une fois cela fait, ouvrez le fichier .aia sur votre ordinateur avec un explorateur de fichiers, vous devriez alors obtenir trois répertoires :



- Entrez dans le répertoire /youngandroidproject et ouvrez le fichier intitulé project.properties avec un éditeur de texte :



```
project.properties ✘
1 main=appinventor.ai_ronanchardonneau.premiere_application.Screen1
2 name=premiere_application
3 assets=../assets
4 source=../src
5 build=../build
6 versioncode=1
7 versionname=1.0
8 useslocation=False
9 aname=premiere_application
10 sizing=Fixed
```

Afin de pouvoir travailler avec le composant **ActivityStarter** vous aurez besoin de deux éléments :

- le nom du package, ici appinventor.ai_ronanchardonneau.premiere_application
 - la classe de l'activité, ici appinventor.ai_ronanchardonneau.premiere_application.Screen1
- Une fois ces informations mises de côté, il ne vous restera plus qu'à créer une autre application de ce type :

```

when Screen1 .Initialize
do set ActivityStarter1 . ActivityPackage to " appinventor.ai_ronanchardonneau.premiere_application "
set ActivityStarter1 . ActivityClass to " appinventor.ai_ronanchardonneau.premiere_application.Screen1 "
call ActivityStarter1 .StartActivity

```

Une fois cette application installée, vous verrez qu'au lancement de celle-ci, elle vous redirigera directement vers la première. Félicitations, vous maîtrisez les bases du composant **ActivityStarter**.

Vous pouvez également utiliser les applications natives fournies avec votre appareil. C'est notamment le cas de l'application YouTube.

Par exemple, l'application suivante lancera directement une vidéo YouTube :

```

when Screen1 .Initialize
do set ActivityStarter1 . Action to " android.intent.action.VIEW "
set ActivityStarter1 . DataUri to " https://www.youtube.com/watch?v=Tqwwq3IMwQE "
call ActivityStarter1 .StartActivity

```

Ici, les propriétés Action indiquent l'utilisation d'un navigateur ; DataUri, l'URL à laquelle vous souhaitez que le navigateur accède.



2. Bluetooth

Qu'est-ce que le Bluetooth ?

Bluetooth est un standard de communication permettant l'échange bidirectionnel de données à très courte distance, de l'ordre de quelques mètres, utilisant des ondes radio UHF.

Ce protocole de communication va être très intéressant dans notre cas car il va nous permettre de communiquer avec des objets sans forcément disposer de connexion Internet.

Comment trouver l'adresse Bluetooth de son téléphone ?

Très souvent vous la trouverez en suivant un chemin de ce type : **Paramètres - À propos de l'appareil - Etat - Adresse Bluetooth**. Si aucune adresse n'apparaît, c'est probablement que le Bluetooth de votre téléphone n'est pas activé.

À quoi ressemble une adresse Bluetooth ?

Une adresse Bluetooth est composée de la manière suivante : D0:87:E2:AD:FE:66

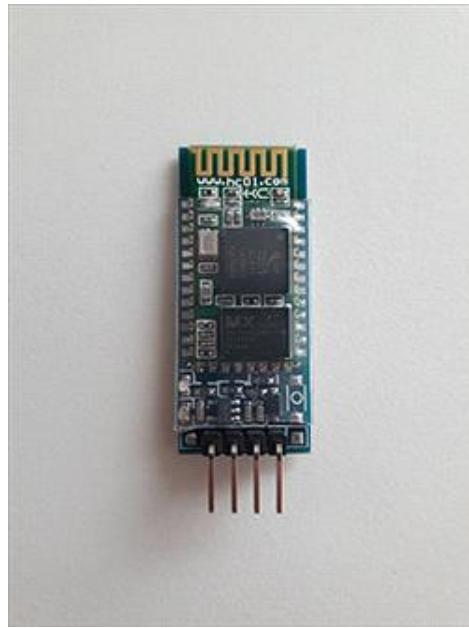
Les nombreux exemples que vous retrouvez dans la littérature pour utiliser le Bluetooth avec App Inventor 2 sont souvent des exemples de communication avec des cartes Arduino ou d'autres téléphones.

Qu'est-ce qu'Arduino ?

Arduino est le nom d'un fabricant italien de circuits imprimés sur lesquels il est possible de brancher toutes sortes d'appareils. Cette carte se programme sur l'ordinateur via un câble USB et permet ensuite de diriger n'importe quel appareil, il suffit pour cela de modifier le code exécuté par l'Arduino. Arduino est également l'un des symboles de l'ère des objets connectés car il permet à une personne avec peu de connaissance en physique de réaliser un prototype fonctionnel d'un objet électronique. Les cartes Arduino existent dans différents types que vous trouverez facilement sur la plupart des places de marché mais également sur le site officiel : <https://www.arduino.cc>. Pour l'exemple de ce livre, nous avons choisi l'une des plus basiques, la carte Arduino Uno (comptez une vingtaine d'euros pour la carte seule, cinquante pour un kit).



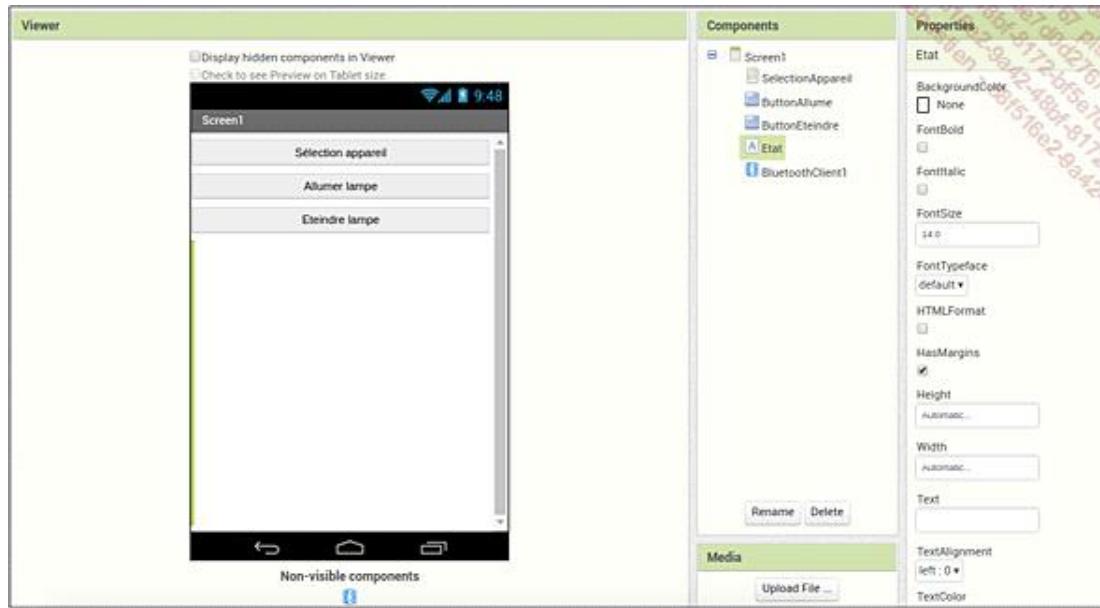
Les cartes Arduino Uno sont assez basiques et ne permettent par défaut de communiquer que via le port USB. Nous allons donc lui ajouter un composant Bluetooth qui va lui permettre de recevoir des instructions depuis notre application App Inventor 2. Les composants Bluetooth pour Arduino s'achètent de la même manière que la carte. Comptez un peu moins de 5 € :



Réalisation d'une lampe connectée avec une carte Arduino

Dans l'exemple qui va suivre, nous allons réaliser une application permettant de communiquer avec une lampe que nous allons créer par l'intermédiaire d'un circuit Arduino. Cette lampe sera connectée via Bluetooth, c'est-à-dire que nous allons envoyer des instructions via Bluetooth par l'intermédiaire de notre application AI 2 au circuit Arduino. En fonction des instructions que nous allons donner, la lampe s'allumera ou s'éteindra.

Voici à quoi va ressembler notre application :



La première étape de notre application consiste à faire connecter notre application au composant Bluetooth de la carte Arduino. Nous allons donc devoir sélectionner ce composant parmi une liste d'appareils Bluetooth à proximité. C'est pourquoi nous allons ajouter le composant **ListPicker** afin de sélectionner l'appareil de notre choix.

Nous allons également ajouter deux boutons qui serviront à allumer et éteindre la lampe.

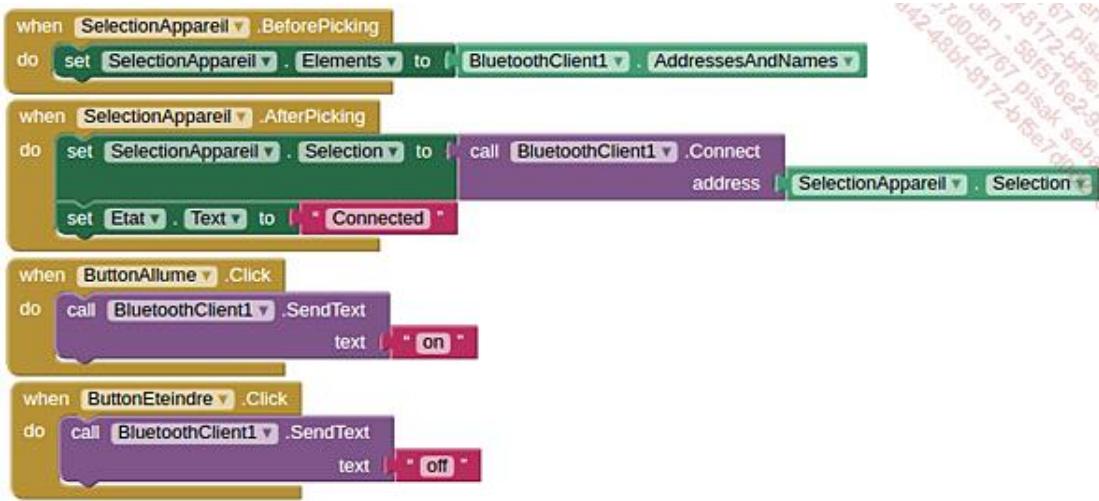
- Un composant Bluetooth afin de pouvoir communiquer avec notre lampe.

- Un composant Label afin de pouvoir nous indiquer l'état de connexion avec notre carte Arduino.

Nous avons donc la configuration suivante :

- un composant ListPicker
- deux boutons (Button)
- un label
- un composant Bluetooth Client

Par le code, nous allons avoir la configuration suivante :



Ici, les instructions de code permettent de mettre dans une liste l'ensemble des appareils Bluetooth détectés à proximité, avec leur adresse et leur nom.

Lorsqu'un de ces appareils est sélectionné grâce au composant **ListPicker**, alors une connexion s'effectue et le libellé prend la valeur **Connected**.

Lorsqu'on clique sur le premier bouton, celui-ci envoie le message **on**. Lorsqu'on clique sur le deuxième bouton, celui-ci envoie le message **off**.

Une fois cette application réalisée, il nous faut mettre en place notre montage Arduino. Celui-ci est composé des éléments suivants :

- une carte Arduino Uno
- un câble USB pour relier notre ordinateur à la carte et injecter le programme
- un composant Bluetooth vu plus haut dans ce chapitre
- une simple LED rouge
- six câbles de connexion Arduino.

→ Dans un premier temps, branchez le câble USB et la carte Arduino à l'ordinateur et injectez le programme suivant dans le logiciel Arduino :

```
int ledPin = 13;
```

```

String readString;

void setup() {
    Serial.begin(9600);
    pinMode(ledPin, OUTPUT);
}

void loop() {
    while (Serial.available()) {
        delay(3);
        char c = Serial.read();
        readString += c;
    }
    if (readString.length() > 0) {
        Serial.println(readString);
        if (readString == "on") {
            digitalWrite(ledPin, HIGH);
        }
        if (readString == "off") {
            digitalWrite(ledPin, LOW);
        }
        readString = "";
    }
}

```

Que nous dit ce code ?

Tout simplement que si jamais la carte Arduino reçoit l'information **on**, alors elle fera passer le courant jusqu'à la LED, si en revanche elle reçoit l'information **off** alors elle ne fera pas passer le courant et la LED sera donc éteinte.

À noter que ce code doit être injecté avant l'ajout des composants sur la carte afin d'éviter tout problème de téléversement (transfert du programme).

Au niveau du câblage :

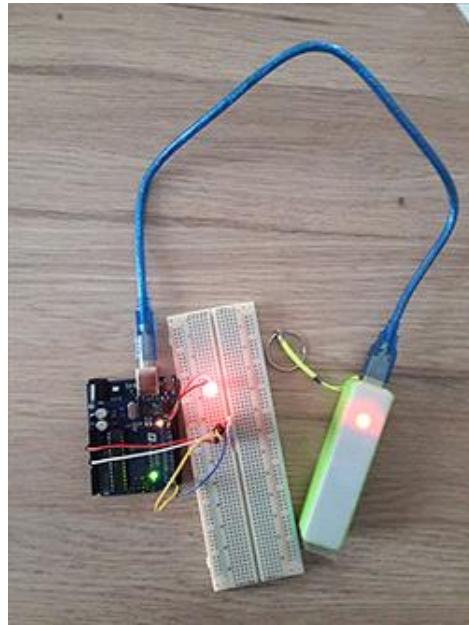
Le composant Bluetooth doit avoir :

- l'entrée TX correspondant à l'entrée RX de la carte Arduino
- l'entrée RX correspondant à l'entrée TX de la carte Arduino
- l'entrée GND associée à une entrée GND de la carte Arduino
- l'entrée VCC associée à l'entrée 3.3V de la carte Arduino

À cela, il faudra ajouter une LED dont la patte la plus longue sera associée à l'entrée numéro **13** de la carte Arduino et la petite patte à une borne GND de la carte Arduino.

Une fois le câblage terminé, il ne vous reste plus qu'à lancer votre application en connectant votre téléphone à Arduino via Bluetooth, puis à envoyer les instructions via les deux boutons **on** et **off**.

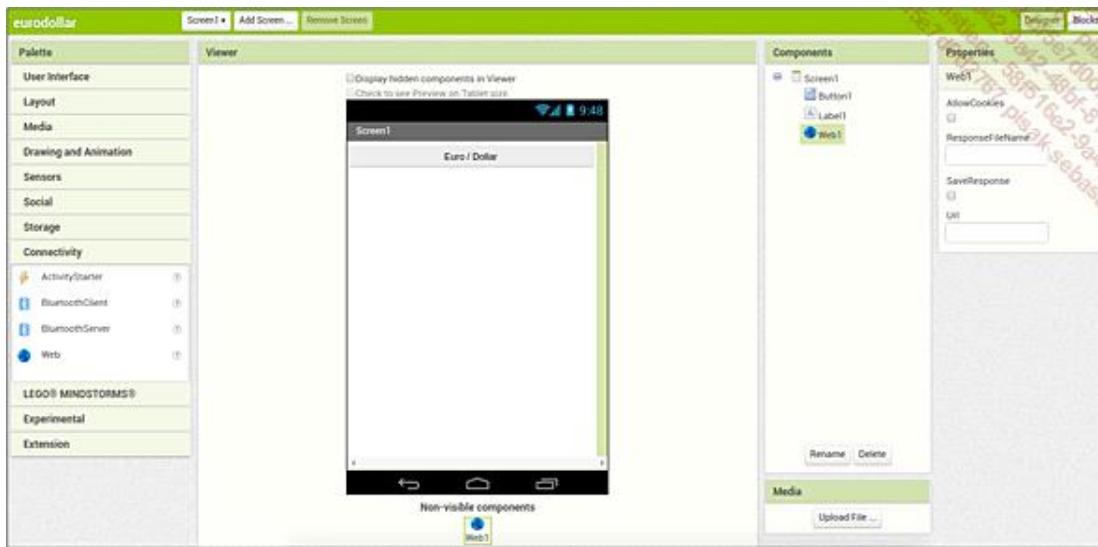
Vous pouvez par la suite donner plus d'autonomie à votre objet en lui associant une batterie USB :



Vous disposez désormais d'une lampe connectée autonome dont vous contrôlez l'allumage via Bluetooth avec votre smartphone. L'exemple ici peut faire sourire car il est assez trivial, mais sachez qu'avec Arduino vous pouvez utiliser la plupart des composants électroniques que vous avez autour de vous : LED, boutons pousoirs, détecteurs de mouvement, écrans LCD, buzzer, détecteur de mouvement, laser... Vos projets pourront prendre une dimension que vous aurez probablement du mal à réaliser tellement les perspectives sont grandes.

3. Web

Pour montrer l'utilité du composant **Web**, nous allons l'illustrer par un exemple simple et connu. Celui-ci va consister à aller chercher une valeur sur Internet, à savoir dans notre cas de figure le taux de change euro/dollar et à l'afficher dans notre application.

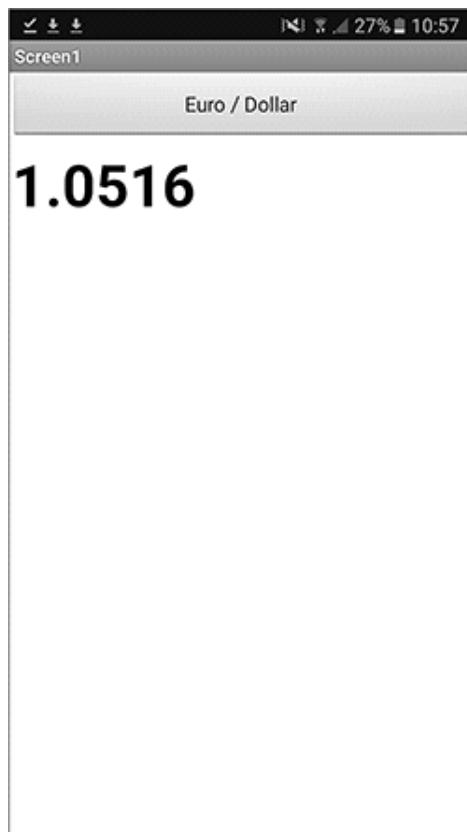


Pour le code, nous allons avoir :



Ici, le code nous indique que si nous cliquons sur un bouton, alors le composant **Web** associé prendra la valeur de l'URL à laquelle est stockée le taux de change euro/dollar. Une fois que cette donnée est récupérée, alors le champ **Label** prendra la valeur du contenu de cette requête web.

Ce qui nous donnera le résultat suivant :



GotFile

Cet événement va vous permettre de vérifier si la requête web a bien été exécutée, le résultat sera affiché sous la forme d'un fichier.

GotText

Cet événement va vous permettre de vérifier si la requête web a bien été exécutée, le résultat sera affiché sous forme de texte.

Les fonctions des composants LEGO® Mindstorms®

Un livre entier pourrait être consacré à des tutoriels pour LEGO Mindstorms EV3 tellement les possibilités sont légion. Nous avons pris le choix dans cet ouvrage de ne présenter que le concept global plutôt que de détailler l'ensemble des fonctionnalités proposées par App Inventor 2.

Se connecter à EV3

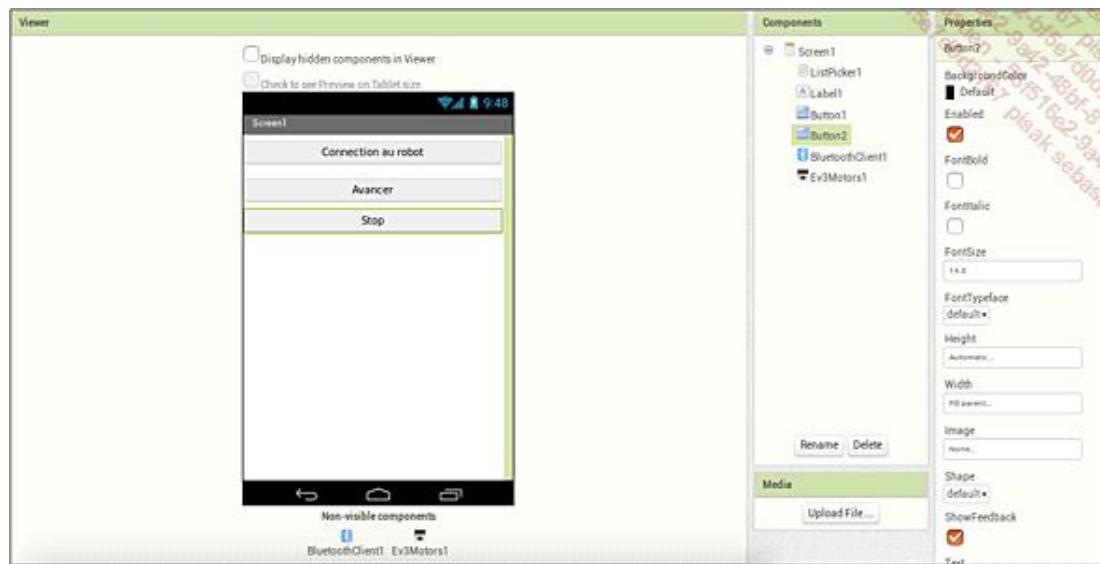
Le plus gros challenge que vous aurez avec EV3 est de vous y connecter, vous devrez pour ce faire, vous connecter via Bluetooth.

Exemple de connexion en mode Bluetooth :

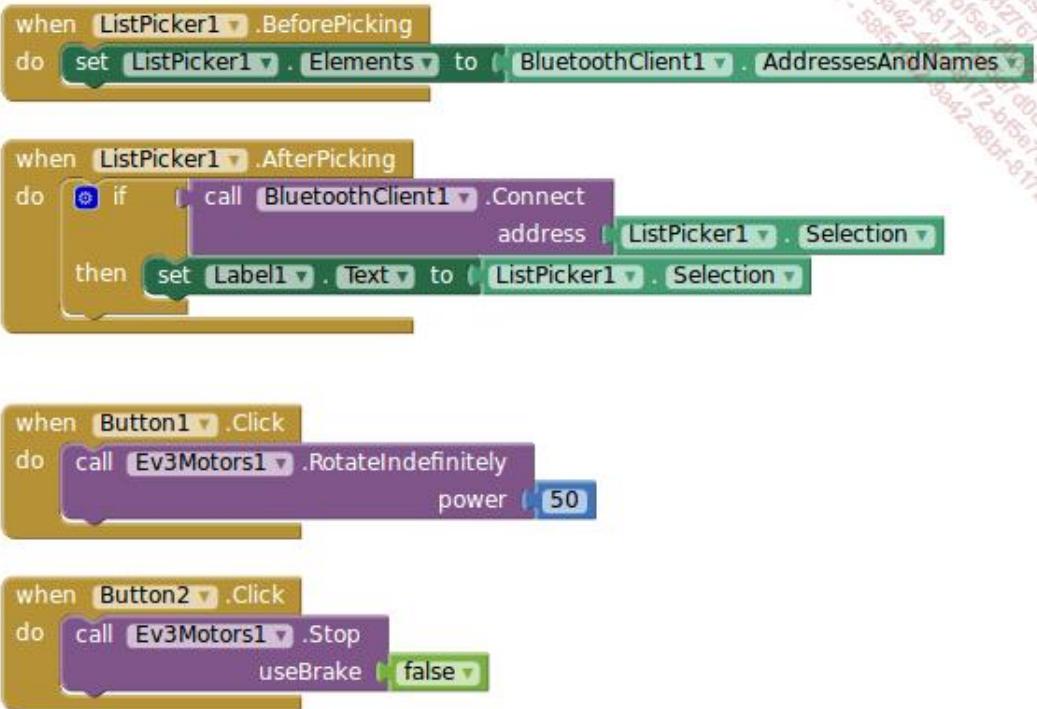
L'exemple que nous allons présenter ici est celui d'une connexion via Bluetooth, permettant par la simple pression d'un bouton de faire fonctionner un moteur, par exemple pour faire avancer un robot. La pression d'un autre bouton permettra d'arrêter le moteur.

Nous utiliserons pour cela :

- un composant Listpicker
- un libellé
- deux boutons
- un composant Bluetooth
- un composant moteur EV3



Pour le code nous aurons :



Comme vous pouvez le constater dans l'application ci-dessus, lorsque vous cliquerez sur le bouton, celui-ci vous présentera les appareils Bluetooth à proximité, auxquels vous souhaitez vous appareiller. Une fois cette connexion effectuée, un libellé prend la valeur de l'appareil à laquelle votre application s'est connectée.

Si vous appuyez sur le bouton **Avancer** (Button1), alors le moteur du robot va s'activer. Si vous appuyez sur l'autre bouton, celui-ci va s'arrêter net. Grâce à cette application, vous pourrez par exemple faire avancer votre robot de manière indéfinie et décider de l'arrêter quand vous le souhaitez.

Les fonctions des composants Experimental

Comme indiqué précédemment, l'utilisation du composant **Firebase**, pourra s'arrêter du jour au lendemain. Nous avons tout de même pris le parti de vous le présenter ici.

1. FirebaseDatabase

Événements

DataChanged

Probablement la fonctionnalité la plus puissante de ce composant. Elle permet d'enregistrer le fait que la valeur dans la base de données associée à un tag a changé.

Cela signifie que concrètement vous pouvez mettre à jour en temps réel une donnée d'une application client en fonction d'une autre application client. Imaginons une application dans laquelle un tag appelé "score" a changé car un joueur a battu le record alors le joueur 2 verra ce nouveau score. En effet, lorsque le joueur 1 a battu le record, ce score a été envoyé à la base de données de Firebase qui, elle, a mis à jour le score. L'application de joueur 2 a vérifié à un moment donné que ce score a été mis à jour et l'a affiché en conséquence.

Imaginons un jeu dont le fonctionnement est le suivant :

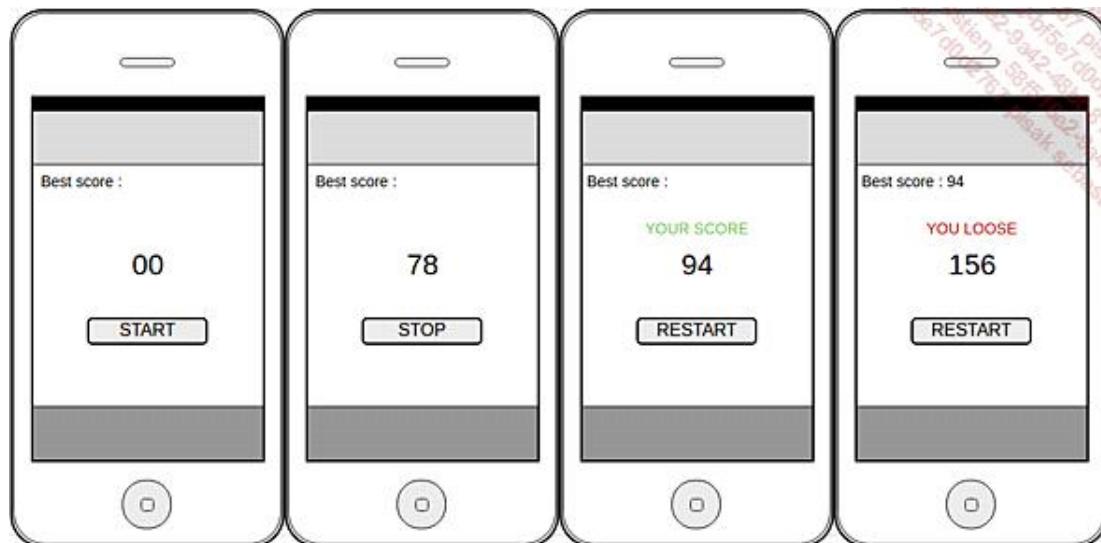
Nom de l'application : Chrono Game

Objectif du jeu : Arrêter le chronomètre le plus proche de la valeur 100

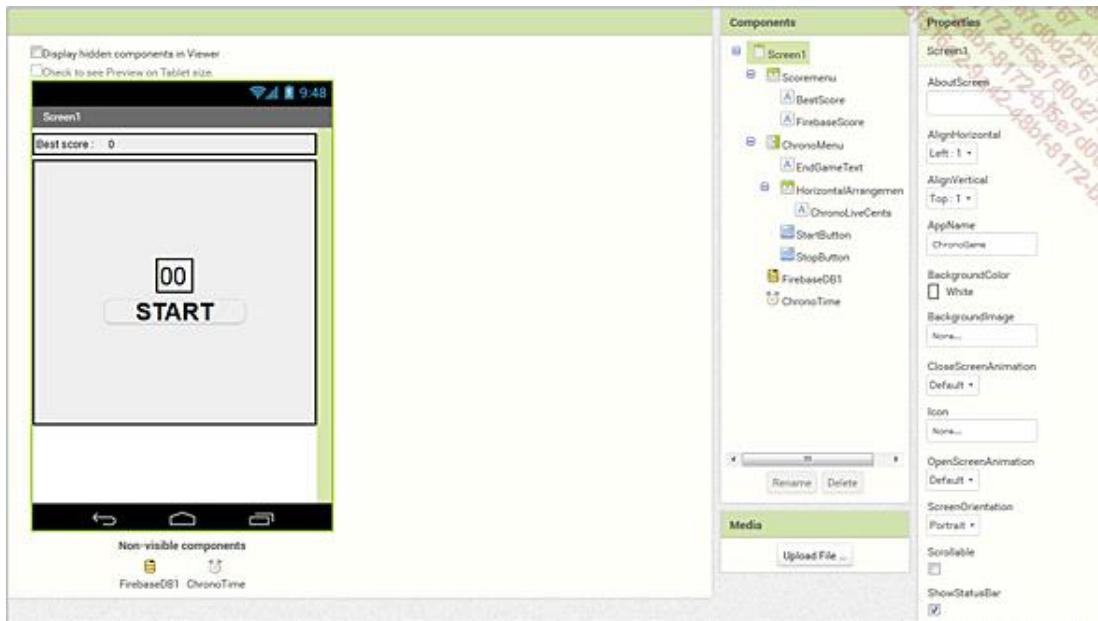
Fonctionnement du jeu : Lorsqu'on clique sur le bouton **START**, le chronomètre démarre et affiche le temps qui passe. Si le temps d'une seconde soit 100 est dépassé alors le compteur sera bloqué et indiquera que la partie est terminée. Le bouton prend alors la valeur **Restart**. Dès qu'on clique sur le bouton **RESTART**, le chronomètre repart à 0 et ainsi de suite.

Si la valeur est inférieure ou égale à 100, alors on regardera si le score enregistré dans Firebase est plus proche de 100, si ce n'est pas le cas alors il faudra le mettre à jour et afficher le meilleur score dans l'application en haut à droite de l'écran.

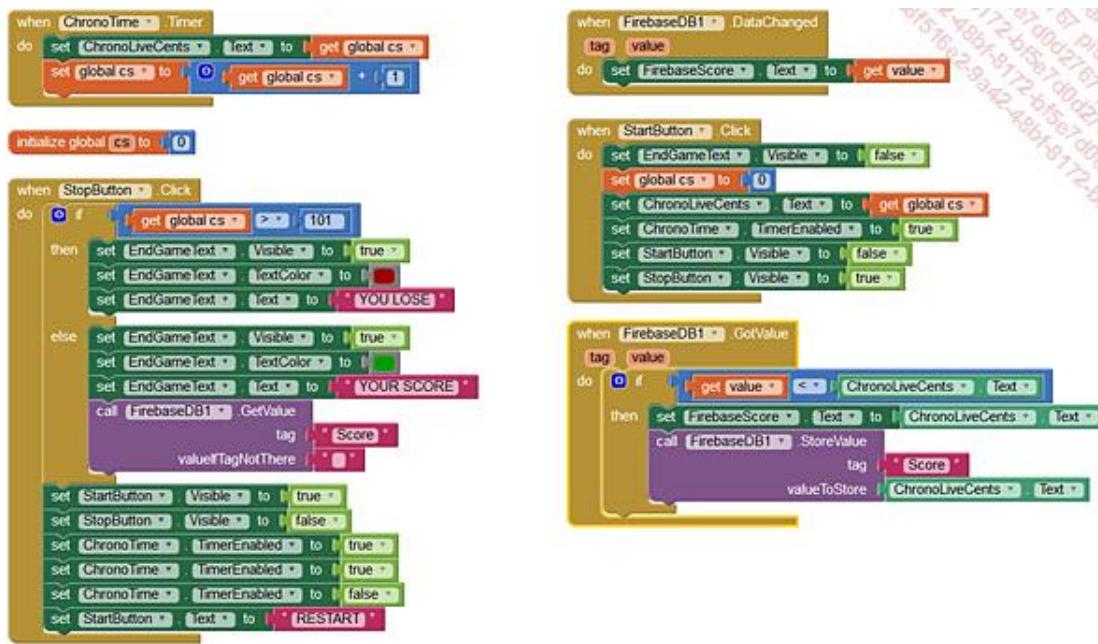
Notre application ressemblera à :



La partie **Designer** de notre application aura la forme suivante :



Et la configuration du code sera la suivante :



Ce qui nous intéresse dans la capture d'écran ci-dessus c'est l'événement `DataChanged`. Cette configuration `DataChanged` sert à vérifier si la valeur de la base de données de Firebase a changé. Si tel est le cas, alors le score doit être mis à jour.

Concrètement ici on peut imaginer deux personnes qui ont installé cette même application ; lorsqu'un joueur fait un meilleur score que celui enregistré dans la base de données de Firebase alors le score est affiché sur tous les smartphones qui possèdent cette application.

FirebaseError

Cet événement indique que la communication avec la base de données de Firebase a rencontré une erreur et vous permet de déboguer votre application en faisant apparaître le message d'erreur associé.

FirstRemoved

Cet événement est déclenché par l'événement RemoveFirst que nous allons voir plus bas. La variable associée, à savoir **value**, est la variable qui était la première dans la liste et qui est maintenant retirée.

GetValue

Cet événement permet d'indiquer que la méthode GetValue a bien été exécutée et que donc la valeur a bien été récupérée de la base de données.

TagList

Cet événement est déclenché quand vous recevez la liste de tous les tags connus. Cet événement est utilisé avec la fonction GetTagList.

Méthodes

AppendValue

Permet d'ajouter une valeur à la liste Firebase.

ClearTag

Cette fonction vous permet de supprimer un Tag de Firebase.

GetTagList

Cette fonction vous permet d'obtenir la liste de tous les tags dans la base de données de Firebase, vous pouvez les faire apparaître via l'événement TagList.

GetValue

Cette méthode permet de récupérer la valeur du tag associé. En l'absence de tag associé, la valeur associée sera vide.

RemoveFirst

Cette fonctionnalité va faire ressortir le premier élément de la liste et le supprimer.

StoreValue

Cette méthode va vous permettre de pouvoir stocker une donnée, l'associer à un tag et l'envoyer dans Firebase.

Unauthenticate

Cette fonction sert à vous déconnecter de votre compte Firebase si vous rencontrez des problèmes avec ce composant. Vous pouvez l'utiliser si vous avez inclus plusieurs composants **Firebase** à votre application. Cependant, et étant donné que Firebase utilise un compte spécifique pour le MIT en tant que composant expérimental, vous ne devriez pas avoir besoin d'utiliser cette fonctionnalité.

Introduction

Dans cette partie, nous allons découvrir comment faire connaître notre application mobile. Nous allons présenter deux méthodes, celle de la galerie App Inventor 2 et celle du Play Store de Google.

Publier son application dans la communauté App Inventor 2

Vous pouvez faire connaître votre application en la publiant dans la galerie App Inventor 2, cela vous permettra de voir si celle-ci peut rencontrer du succès auprès des membres de la communauté.

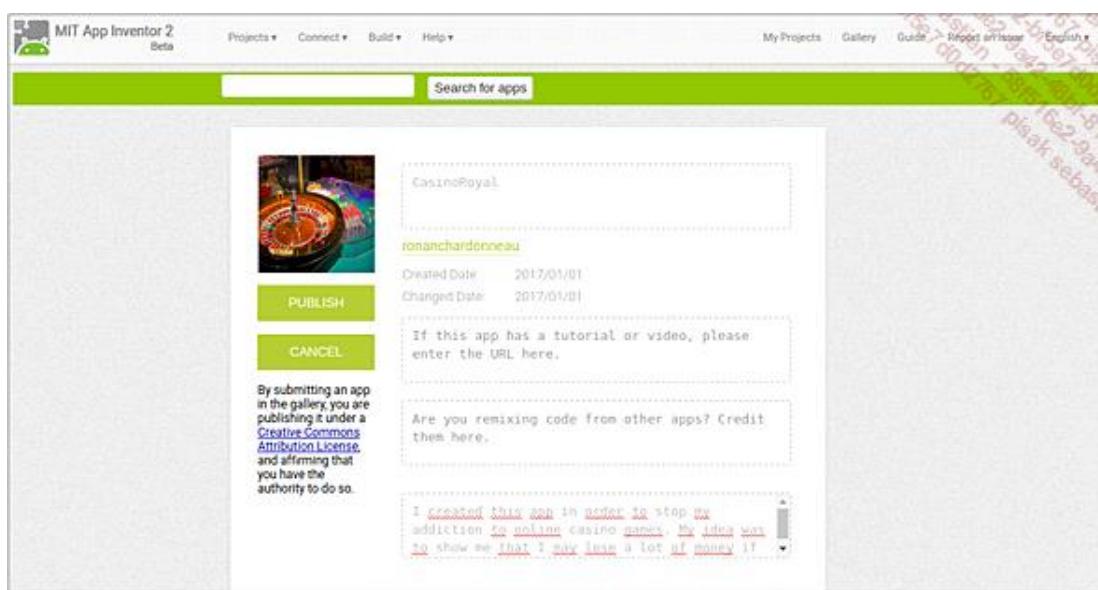
- Pour ce faire, cochez la case correspondante à votre application et appuyez sur **Publish** :



The screenshot shows the 'My Projects' section of the MIT App Inventor 2 interface. A list of projects is displayed with columns for Name, Date Created, Date Modified, and Published status. The 'CasinoRoyale' project is highlighted with a yellow background, indicating it is selected. The 'Published' column shows 'No' for all projects except 'CasinoRoyale', which has 'Yes'.

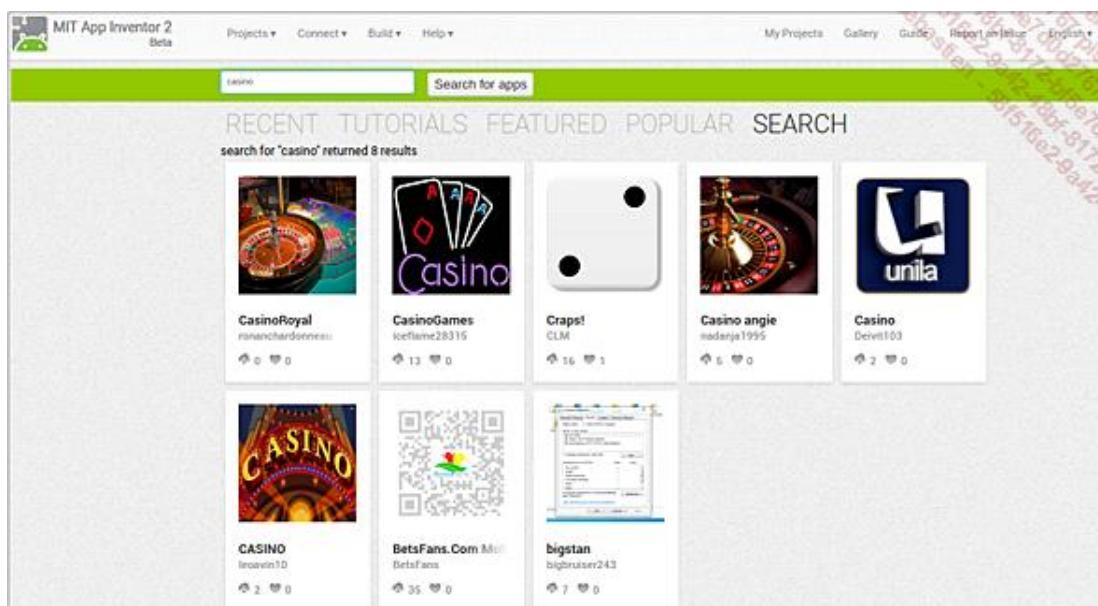
Name	Date Created	Date Modified	Published
SuperTux	Jul 12, 2016, 5:06:00 PM	Jan 1, 2017, 10:50:10 AM	No
eurodollar	Jul 3, 2016, 6:03:33 PM	Jan 1, 2017, 11:06:26 AM	No
CasinoRoyale	Dec 17, 2016, 3:10:46 PM	Jan 1, 2017, 10:50:05 AM	Yes
TwitterWithPiwik	Jul 24, 2016, 11:45:29 AM	Jan 1, 2017, 10:46:33 AM	No
NFC			No

Une fois ce bouton cliqué, App Inventor 2 vous demandera de confirmer si vous avez bien les droits sur l'ensemble des ressources que vous utilisez :



The screenshot shows the 'Publish' confirmation dialog. It displays the project name 'CasinoRoyal' and the author 'ronanchardeorneau'. There are fields for 'Created Date' (2017/01/01) and 'Changed Date' (2017/01/01). Below these are two text areas: one for a tutorial URL and another for crediting remixes. A note at the bottom states: 'I created this app in order to stop my addiction to online casino money. My idea was to show me that I may lose a lot of money if I ...'

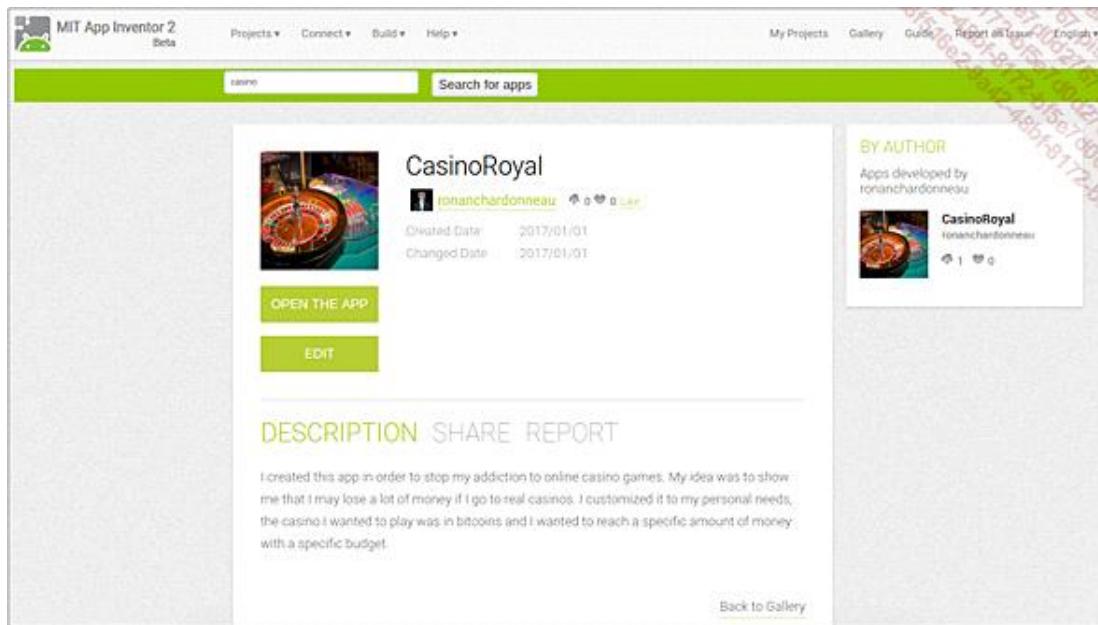
- Cliquez ensuite sur **Publish**. Après quelques minutes, votre application sera directement référencée dans la galerie d'App Inventor 2 :



The screenshot shows the search results for 'casino' on the MIT App Inventor 2 gallery. The results are displayed in a grid format. Each result includes a thumbnail, the app name, the author's name, and the number of likes and comments. The apps shown are: CasinoRoyal (ronanchardeorneau), CasinoGames (icetime28315), Craps! (CLM), Casino angie (maderaj1995), CASINO (louavln10), BetsFans.Com Mu... (BetsFans), bigstan (bigbrusier243), and unila (Dervit103).

Thumbnail	App Name	Author	Likes	Comments
	CasinoRoyal	ronanchardeorneau	0	0
	CasinoGames	icetime28315	13	0
	Craps!	CLM	16	1
	Casino angie	maderaj1995	6	0
	CASINO	louavln10	2	0
	BetsFans.Com Mu...	BetsFans	35	0
	bigstan	bigbrusier243	7	0
	unila	Dervit103	2	0

Il ne vous reste plus qu'à favoriser l'accroissement du nombre de téléchargements et de "like" de votre application pour voir si les utilisateurs l'apprécient. Pour ce faire, n'hésitez pas à communiquer autour de cette dernière sur des réseaux tels que Twitter, vous pouvez également essayer d'être sélectionné dans le classement des meilleures applications AI2 en envoyant votre candidature à la page suivante : <http://appinventor.mit.edu/explore/app-month-program.html>

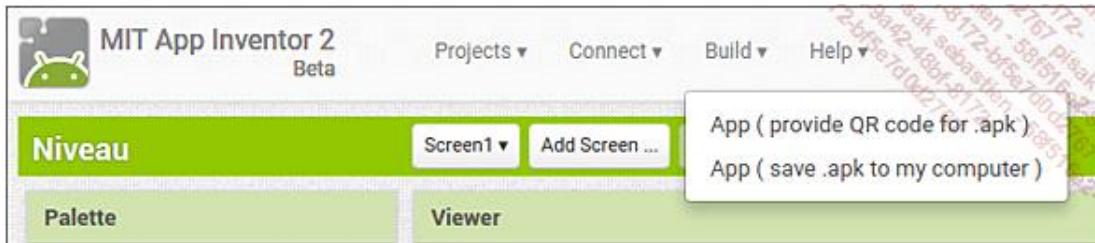


Play Store

Si jamais votre application plaît dans la galerie d'App Inventor 2, vous pouvez aller plus loin en la référençant dans le Play Store de Google afin de la faire connaître au monde entier.

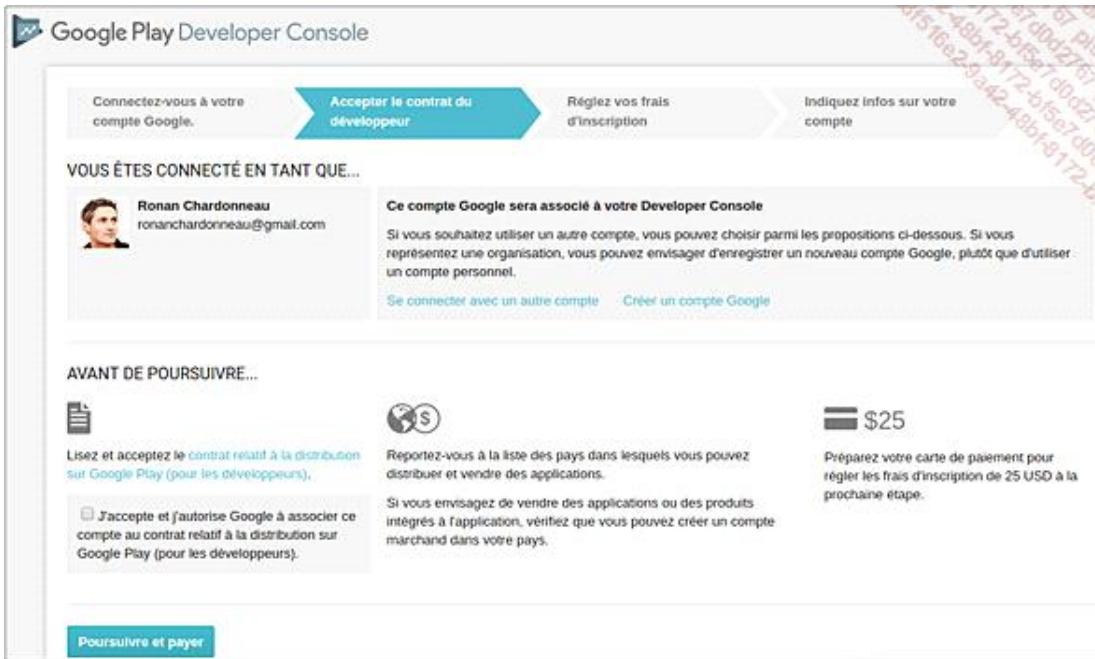
À la différence de la galerie d'App Inventor 2, celle du Play Store est payante.

- La première étape avant de diffuser votre application est de la créer en cliquant sur le bouton **Build** et en choisissant l'option **App (save .apk to my computer)** :



Contrairement à la galerie d'App Inventor 2, publier sur le Play Store, ne diffuse pas votre code source.

- Afin de publier cette application dans le Play Store de Google, rendez-vous à l'adresse suivante : play.google.com/apps/publish/signup



Le contrat relatif à la distribution des applications est disponible ici : <https://play.google.com/about/developer-distribution-agreement.html>.

Il est assez classique pour un contrat Google mais, si votre application est gratuite une grande partie de ce contrat ne vous concerne pas.

Cependant, nous nous devons de vous mettre en garde sur la partie relative aux indemnisations :

"13.1 Dans les limites prévues par la loi, **vous acceptez de défendre, d'indemniser et de dégager de toute responsabilité Google, ses affiliés et leurs administrateurs, dirigeants, salariés et mandataires respectifs, ainsi que leurs Opérateurs téléphoniques agréés, contre toute réclamation, action, poursuite, procédure, perte ou**

responsabilité, et contre tout dommage, coût et frais émanant de tiers (y compris des honoraires d'avocat raisonnables) résultant directement ou indirectement (a) du fait que vous avez utilisé la Developer Console et le Store sans vous conformer au présent Contrat, ou (b) du fait que votre Produit va à l'encontre de droits d'auteur, d'une marque déposée, d'un secret industriel, d'une présentation commerciale, d'un brevet ou d'un autre droit de propriété intellectuelle, quel qu'en soit le détenteur, ou bien du fait que votre Produit est diffamant, ou qu'il porte atteinte au droit à l'image ou à la vie privée d'une personne."

Ce qui signifie qu'en cas de faux pas de votre part, vous vous engagez à défendre Google à vos frais. Autrement dit, vérifiez bien que vous avez légalement le droit de faire ce que vous faites, que vous n'enfreignez aucun droit d'auteur...

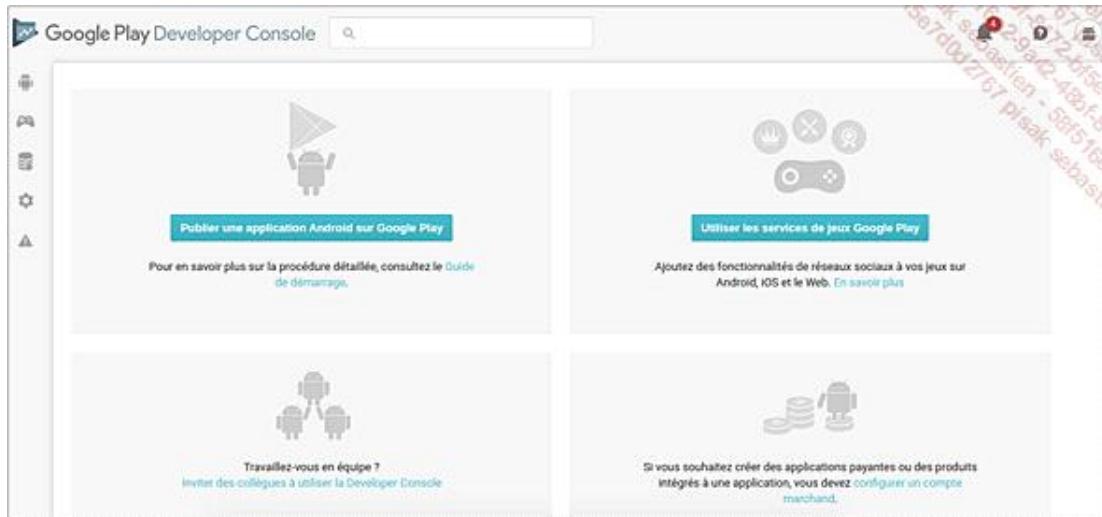
À noter que le contrat relatif à la distribution des applications fait également de nombreux renvois au centre d'informations réglementaire pour les développeurs : <https://play.google.com/intl/fr/about/developer-content-policy/>

Par exemple, voici la liste des contenus non autorisés à la date de rédaction de l'ouvrage : contenu à caractère sexuel explicite, mise en danger de mineurs, violence, intimidation et harcèlement, incitation à la haine, événements sensibles, jeux d'argent et de hasard, activités illégales, contenu généré par l'utilisateur.

Avant de pouvoir commencer à distribuer votre application sur le Play Store, il faudra vous acquitter de la somme de 25 \$. C'est d'une certaine façon une garantie que Google prend pour identifier que vous n'allez pas spammer le Play Store. Cela permet également à Google d'identifier qui est responsable du contenu publié.

Une fois cela fait, le Play Store vous demandera quelques informations personnelles telles que votre nom et prénom, votre numéro de téléphone...

Vous voici désormais dans la console des développeurs Google Play :



- Cliquez alors sur **Publier une application Android sur Google Play**, il vous sera alors demandé le nom de l'application que vous souhaitez publier.

The screenshot shows the Google Play Developer Console interface for a draft application named "Simple niveau". The left sidebar includes icons for Android, Game, and Tools, and lists sections like "Fichiers APK", "Fiche Google Play Store", "Catégorie de contenu", "Tarifs et disponibilité", "Produits intégrés à l'application", "Services et API", and "Conseils d'optimisation". The main area is titled "FICHAIERS APK" and contains three tabs: "PRODUCTION" (selected), "TESTS BÉTA", and "TESTS ALPHA". A note at the bottom states: "Les clés de licence sont désormais gérées individuellement pour chaque application. Si votre application utilise des services de gestion de licence (si elle est payante, ou si elle propose la facturation via l'application ou utilise des fichiers d'extension pour APK, par exemple), obtenez votre nouvelle clé de licence sur la page Services et API." A blue button at the bottom right says "Importer votre premier fichier APK en version production".

Vous avez la possibilité de la publier directement ou de passer par des bacs à sable ou "sandbox" (recommandé), lieux où vous pourrez tester si votre application est conforme avant de la référencer sur le Play Store.

Sachez qu'il y a un délai de quelques heures entre l'action effectuée lorsque vous appuyez sur le bouton et la publication réelle.

La position et la réputation de votre application dans le Play Store de Google dépend en partie du nombre de téléchargements, de désinstallations, d'avis laissés. Ainsi, il est fortement recommandé afin de ne pas faire des dégâts, de mettre en priorité votre application en mode tests bêta et tests alpha.

Introduction

Ce dernier chapitre a pour objectif de vous donner des pistes de réflexion pour aller au-delà d'App Inventor 2. Il comprend une partie relative à la monétisation et à Android Studio.

AdMob

Il existe différentes possibilités pour monétiser une application mobile. Nous avons choisi ici de vous présenter la plus accessible, celle d'AdMob.

AdMob est une entreprise publicitaire de Google diffusant de la publicité sur les terminaux mobiles. D'après une étude menée par l'entreprise britannique Juniper Research Ltd en 2015, seulement 1 % des applications mobiles sont payantes.

Inclure des publicités dans une application est un des modèles économiques pour rendre votre projet rentable, c'est justement ce qu'AdMob vous permet de faire.

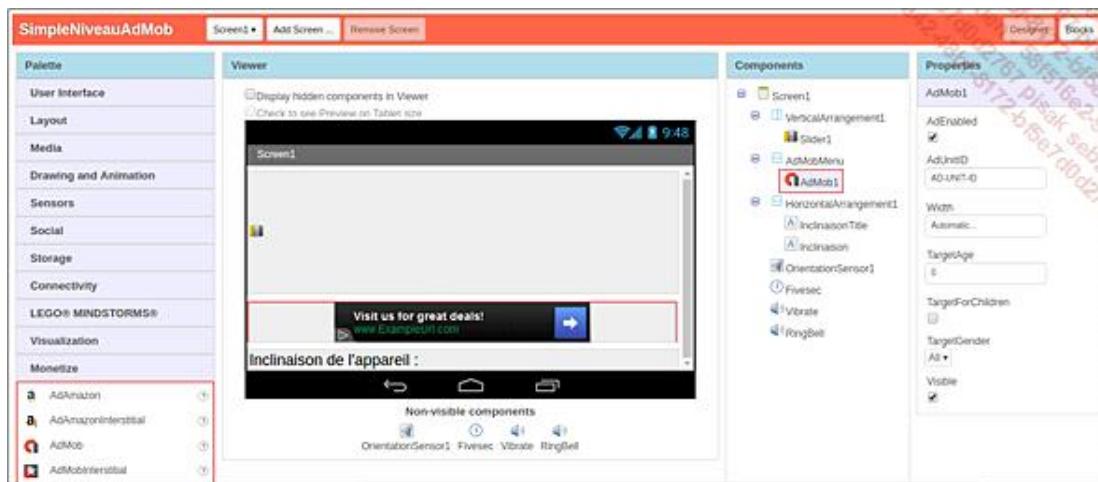
Guide de démarrage rapide d'AdMob : https://support.google.com/admob/answer/6168758?hl=fr&ref_topic=2740022

Présentation rapide d'AdMob : <https://www.youtube.com/watch?v=FanAk25RqNU>

App Inventor 2 est un service exceptionnel qui vous permettra de créer facilement une application fonctionnelle sans être développeur. En revanche, ce service ne vous permet pas encore de pouvoir facilement y intégrer AdMob. Si vous souhaitez intégrer AdMob à l'une de vos applications, il va probablement falloir que vous passiez sur une autre plateforme. Celle que nous avons choisi de présenter ici s'appelle Appy Builder : appybuilder.com. Il s'agit d'un fork payant d'App Inventor 2. L'un des avantages de cette plateforme est que si vous connaissez App Inventor 2, vous ne serez pas dépayssé.

Comment utiliser AdMob ?

- Afin d'utiliser AdMob, vous devez vous rendre sur le site Internet <https://apps.admob.com>. Pour ce faire, il vous faudra un compte Google.
- Une fois dans Appy Builder, vous devez simplement intégrer dans votre application un composant AdMob, ici nous avons choisi une bannière classique :



Lorsque vous démarrez AdMob, quelques questions vont vous être posées :

- Quel est le nom de votre application ?

Monétiser une nouvelle application

1 Sélectionnez une application

RECHERCHER UNE APPLICATION

AJOUTER UNE APPLICATION MANUELLEMENT

Nom de l'application
SimpleNiveau

Plate-forme ANDROID

AJOUTER UNE APPLICATION

ANNULER

2 Sélectionner le format de l'annonce et attribuer un nom au groupe d'annonces

- Le type de publicité que vous souhaitez avoir :

2 Sélectionner le format de l'annonce et attribuer un nom au groupe d'annonces

BANNIÈRE

INTERSTITIEL

INTERSTITIEL AVEC RÉCOMPENSE

NATIF

Type d'annonce
 Annonce textuelle
 Annonce illustrée

Le fait de limiter les types d'annonces peut entraîner une baisse des revenus.

Actualisation automatique
 Pas d'actualisation
 Fréquence d'actualisation : 60 secondes (30 à 120 secondes)

Style d'annonce textuelle STANDARD

Nom du bloc d'annonces SimpleNiveauBannierePrincipale

(i) Le type, la taille et l'emplacement de l'annonce sont définis au moment de l'intégration du code à l'aide du SDK Google Mobile Ads.

- Une fois cette opération de sélection effectuée, vous obtiendrez les éléments nécessaires pour configurer votre compte AdMob dans Appy Builder.

4 Prenez connaissance des instructions de mise en œuvre

1. Téléchargez le [SDK Google Mobile Ads](#).
2. Consultez le [guide relatif à l'intégration du SDK](#). Lorsque vous intégrez le code, vous devez spécifier le type, la taille et l'emplacement des annonces.
 - ID de l'application : ca-app-pub-152758755██████████32765
 - ID du bloc d'annonces : ca-app-pub-1527587██████████032361



[ENVOYER UN E-MAIL AVEC CES INSTRUCTIONS](#)

[CRÉER UN AUTRE BLOC D'ANNONCES](#)

OK

→ Au niveau d'Appy Builder, la configuration d'AdMob est assez simple :

The screenshot shows the Appy Builder interface with two main panels: 'Components' on the left and 'Properties' on the right.

Components Panel:

- Screen1
- VerticalArrangement1
 - Slider1
- AdMobMenu
 - AdMob1** (highlighted with a red box)
- HorizontalArrangement1
 - InclinaisonTitle
 - Inclinaison
- OrientationSensor1
- Fivesec
- Vibrate
- RingBell

Properties Panel:

AdMob1	
AdEnabled	<input checked="" type="checkbox"/>
AdUnitID	ca-app-pub-1527587552576
Width	Fill parent...
TargetAge	0
TargetForChildren	<input type="checkbox"/>
TargetGender	All ▾
Visible	<input checked="" type="checkbox"/>

Et le tour est joué ! Vous pourrez désormais obtenir quelques centimes d'euros pour chaque utilisateur interagissant avec cette bannière.

Android Studio

Dans cette section, nous allons succinctement vous présenter ce que vous pourriez faire avec Android Studio, le logiciel officiel de Google pour la création d'applications sur mesure. Vous pourrez ainsi identifier facilement les limites d'App Inventor 2.

L'objectif n'est pas de vous former de A à Z sur la création d'applications de manière professionnelle sur App Inventor 2 ; pour cela nous préférons vous orienter vers des MOOC que vous pourrez trouver sur Internet depuis un moteur de recherche. Sachez que dans tous les cas ces cours vous demanderont d'avoir des connaissances en Java.

L'exemple que nous prenons pour ce chapitre est la réalisation d'une application simple qui intégrera l'outil Google Analytics, chose qu'il n'est pas possible de faire dans les règles de l'art avec App Inventor 2 à l'heure où l'auteur rédige ces lignes.

Qu'est-ce qu'Android Studio ?

Développé par Google, Android Studio est un environnement de développement pour développer des applications Android : <https://developer.android.com/studio/index.html>

Différences entre Android Studio et App Inventor 2

Mais pourquoi s'intéresser à Android Studio lorsqu'on a déjà App Inventor 2 ?

- App Inventor 2 permet de créer des applications sans rédiger une seule ligne de code. Android Studio, c'est tout le contraire : si vous ne savez pas coder en Java, vous ne pourrez pas l'utiliser.
- Avec Android Studio, vous pouvez tout réaliser sur mesure. Vous avez également beaucoup plus de précision dans la manière de réaliser votre application.
- Pour résumer, App Inventor 2 est à destination d'un public débutant alors qu'Android Studio s'adresse à des utilisateurs expérimentés.
- Il faut compter entre une demi-journée à une journée pour la mise en place fonctionnelle d'Android Studio contre quelques secondes pour App Inventor 2.
- La vitesse d'exécution : App Inventor 2 est extrêmement rapide alors qu'Android Studio va solliciter énormément les ressources de votre machine, il vous faudra être patient.

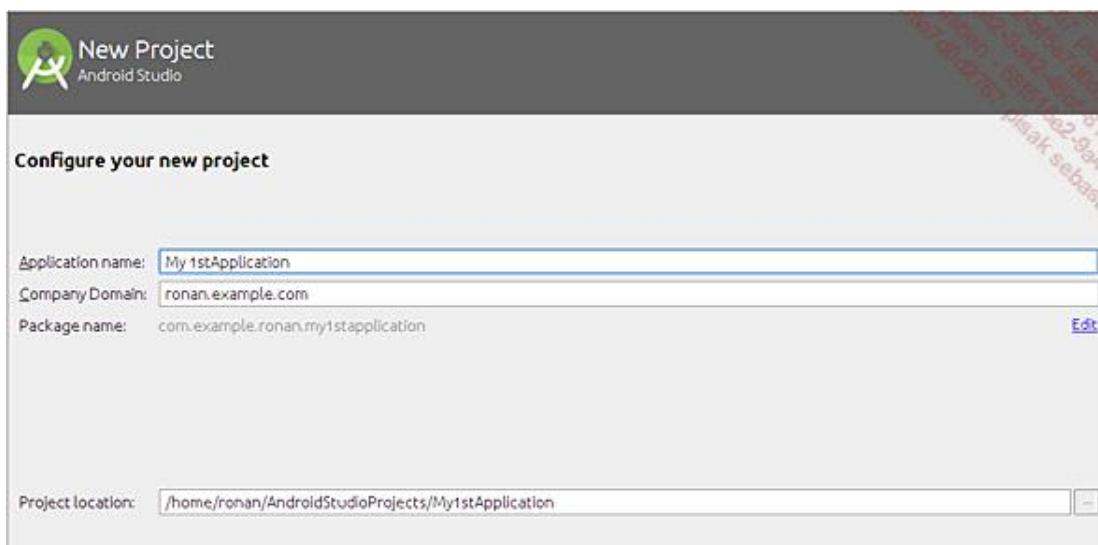
1. Installation d'Android Studio

L'installation d'Android Studio est assez similaire à l'installation des émulateurs d'App Inventor 2 et d'AIRLiveComplete. Ce qui vous posera probablement le plus de tracas, c'est la mise en place correcte du Java Developer Kit. Si vous y arrivez, vous ne devriez pas avoir de problème pour mettre en place Android Studio.

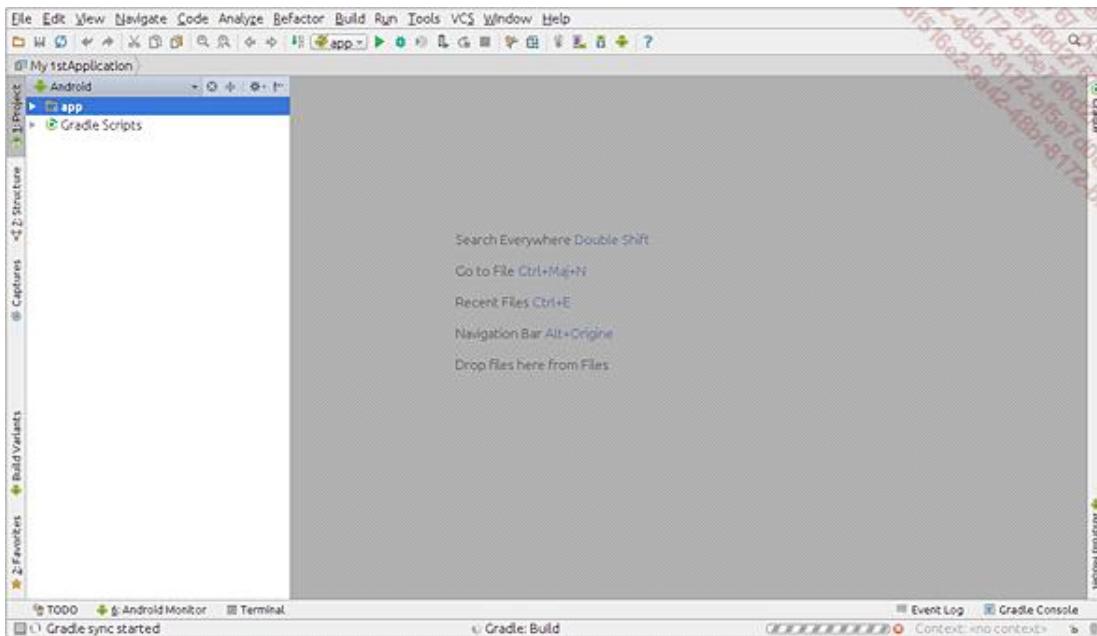
Ce ne sont pas les tutoriels qui manquent sur Internet concernant l'installation d'Android Studio. Pour cet ouvrage, nous avons simplement téléchargé le logiciel officiel. Vous trouverez ce logiciel sur le site officiel : <https://developer.android.com/studio/index.html> ; nous l'avons décompressé dans le répertoire de notre choix, nous nous sommes rendus en ligne de commande dans le répertoire où les fichiers ont été décompressés puis nous avons exécuté le fichier studio.sh avec la ligne de commande ci-dessous (dans notre cas, ce fichier se trouvait dans le répertoire /bin/) :

```
./studio.sh
```

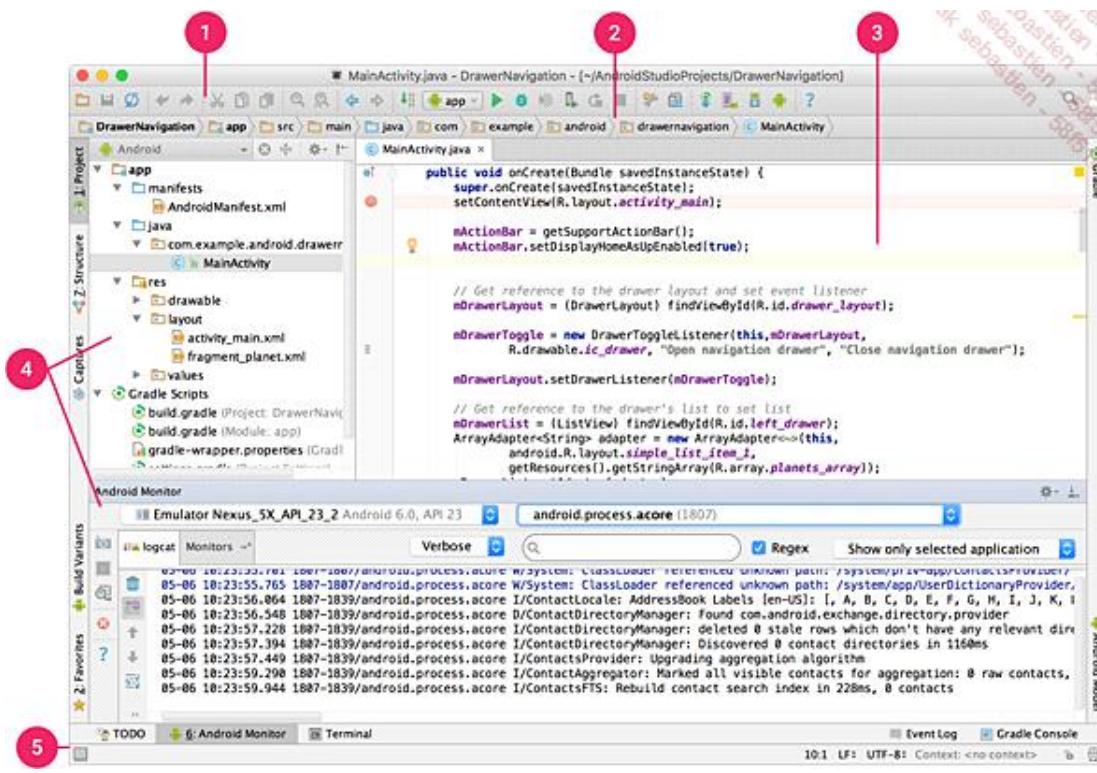
Nous avons ensuite suivi les instructions d'installation à l'écran, et 30 minutes plus tard, Android Studio était installé.



Nous voici désormais dans l'interface d'Android Studio :



Pour les besoins du livre, nous avons repris la présentation de l'interface officielle ci-dessous (source : <https://developer.android.com/studio/intro/index.html>) :



La partie 1 regroupe les outils qui vous permettront de copier-coller des bouts de code, annuler vos actions, déclencher l'émulateur...

La partie 2 vous présente les différents fichiers que vous avez ouverts au sein d'Android Studio et que vous pouvez éditer.

La partie 3 vous permet de modifier le code du fichier sur lequel vous êtes en train de travailler.

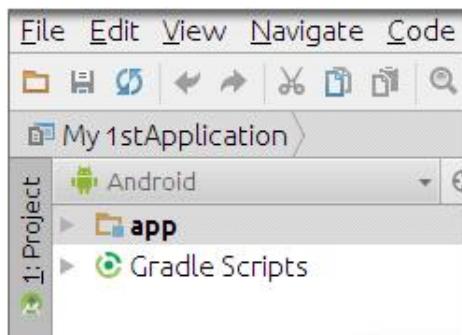
La partie 4 vous permet d'accéder à des fonctions spécifiques telles que la gestion de projets, la recherche. Vous pouvez vous en passer dans un premier temps.

La partie 5 vous indique où vous en êtes dans votre projet et vous affichera au besoin des messages d'avertissement.

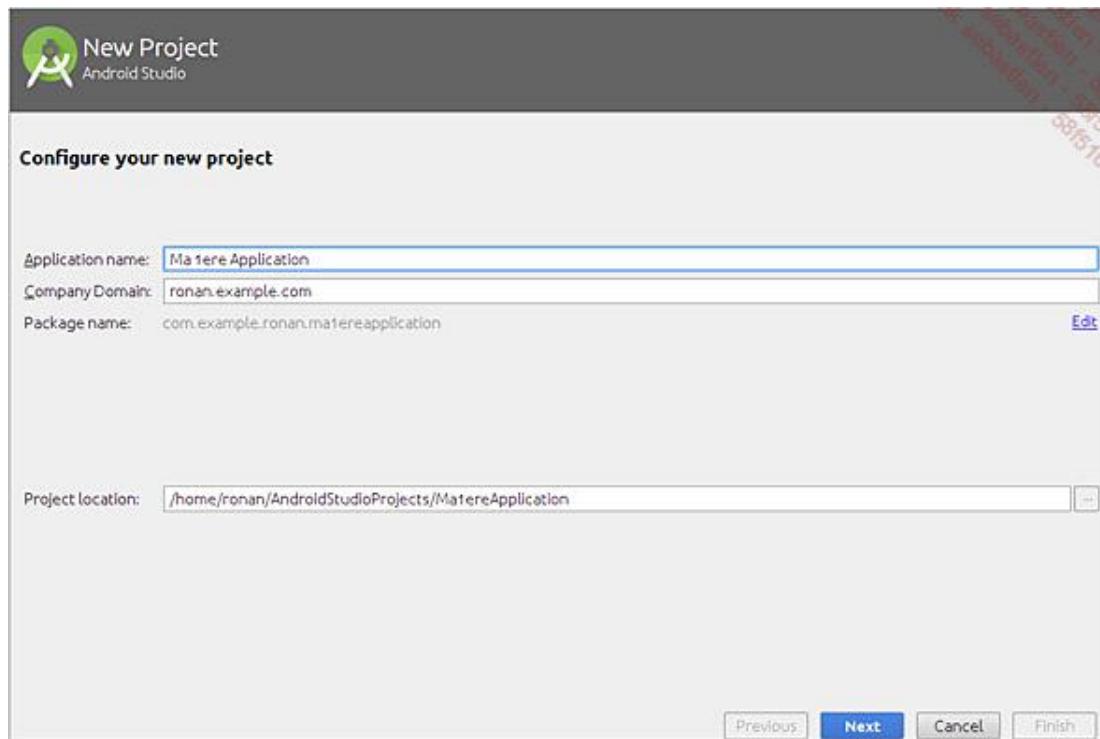
Maintenant que nous avons eu une vue très succincte d'Android Studio, nous pouvons nous lancer dans la création de notre première application.

2. Création d'une application toute simple avec Android Studio

- Afin de créer votre première application sur Android Studio, cliquez en haut à gauche sur **File** et sur **New** et sur **New project** :



- Entrez alors simplement le nom que vous souhaitez donner à votre application et ne changez pas les autres informations qui vous sont proposées (le but est de créer une première application facilement, sans se préoccuper des détails qui vont au-delà de l'objectif fixé par cet ouvrage) :



- Cliquez sur **Next**, vous devez ensuite indiquer l'objectif de votre application :

**Select the form factors your app will run on**

Different platforms may require separate SDKs

 Phone and Tablet

Minimum SDK API 10: Android 2.3.3 (Gingerbread)

Lower API levels target more devices, but have fewer features available.

By targeting API 10 and later, your app will run on approximately **100,0%** of the devices that are active on the Google Play Store.[Help me choose](#) Wear

Minimum SDK API 21: Android 5.0 (Lollipop)

 TV

Minimum SDK API 21: Android 5.0 (Lollipop)

 Android Auto Glass

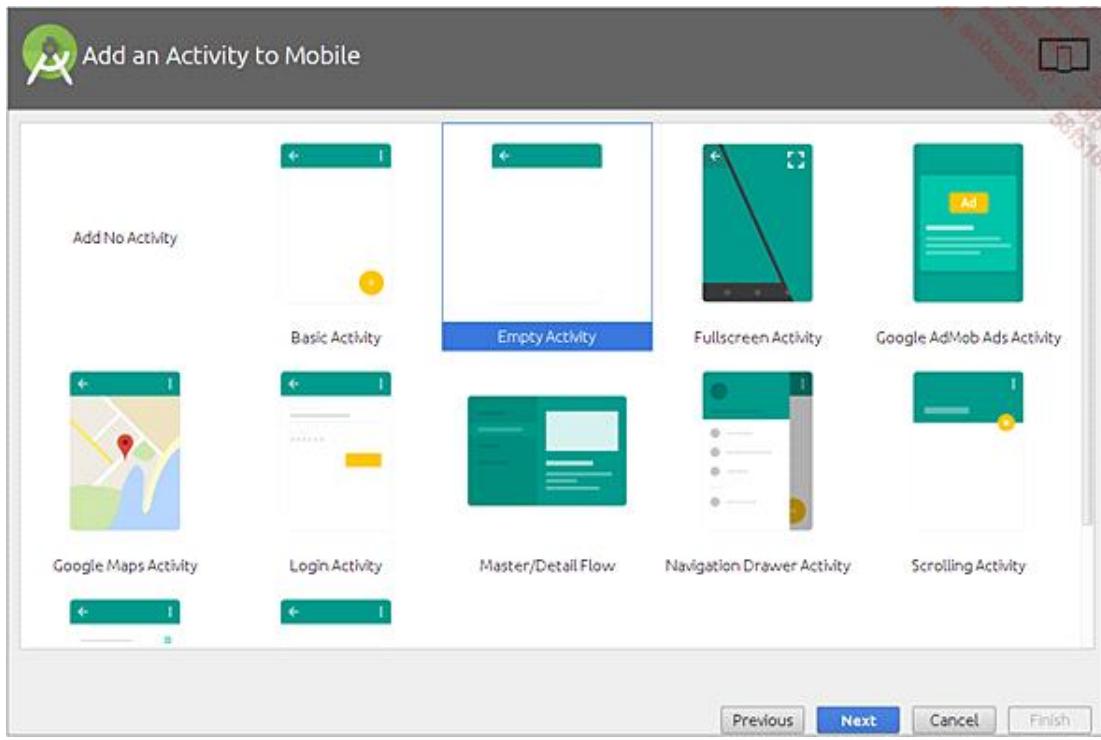
Minimum SDK Glass Development Kit Preview (API 19)

[Previous](#)[Next](#)[Cancel](#)[Finish](#)

Dans notre exemple, nous souhaitons réaliser une application pour les téléphones et tablettes. Nous allons donc laisser la case **Phone and Tablet** cochée. Vous remarquerez sur cet écran un menu déroulant associé. Il s'agit de la version d'Android que vous allez utiliser : plus celle-ci est élevée et plus elle proposera de nouvelles fonctionnalités mais cela signifie aussi qu'elle sera supportée uniquement sur les tous derniers téléphones. Ainsi, dans notre cas de figure, nous avons pris le parti de choisir une vieille version, soit Android 2.3.3 (pour information, le téléphone que nous utilisons à la date de l'écriture de cet ouvrage, acheté en 2016, est en version 5.1.1) qui permettra à notre application d'être supportée sur tous les appareils Android.

→ Cliquez sur **Next**.

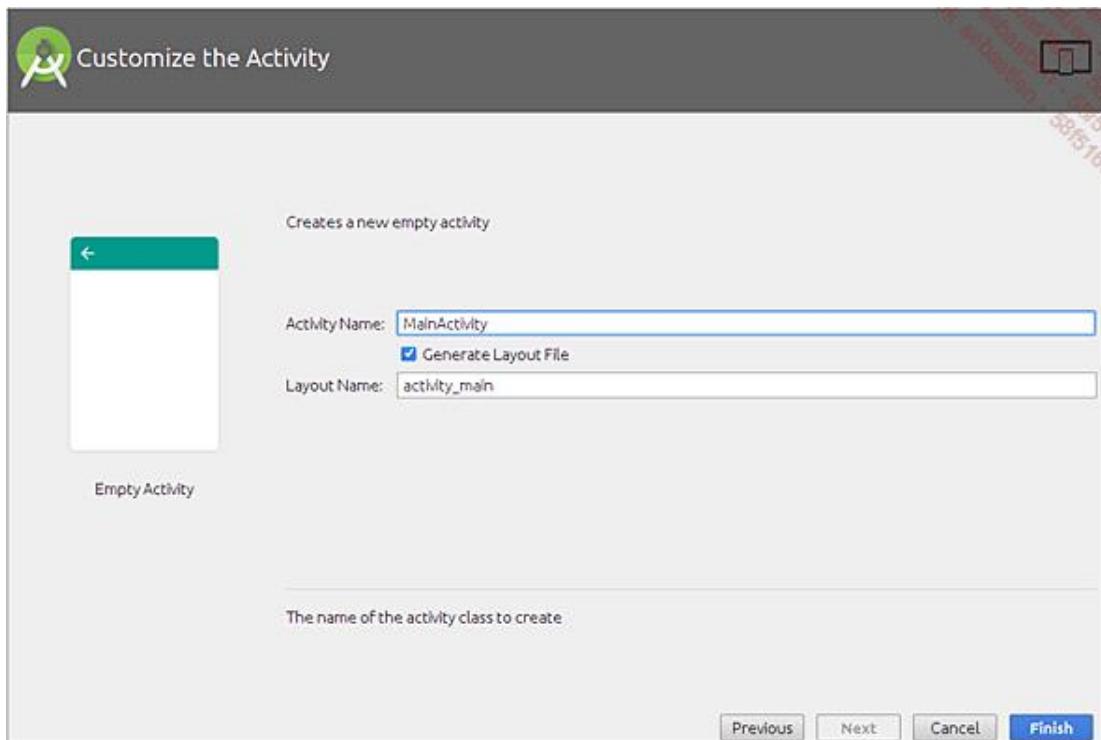
Vous arriverez alors sur l'écran suivant :



Qu'est-ce qu'une activity ?

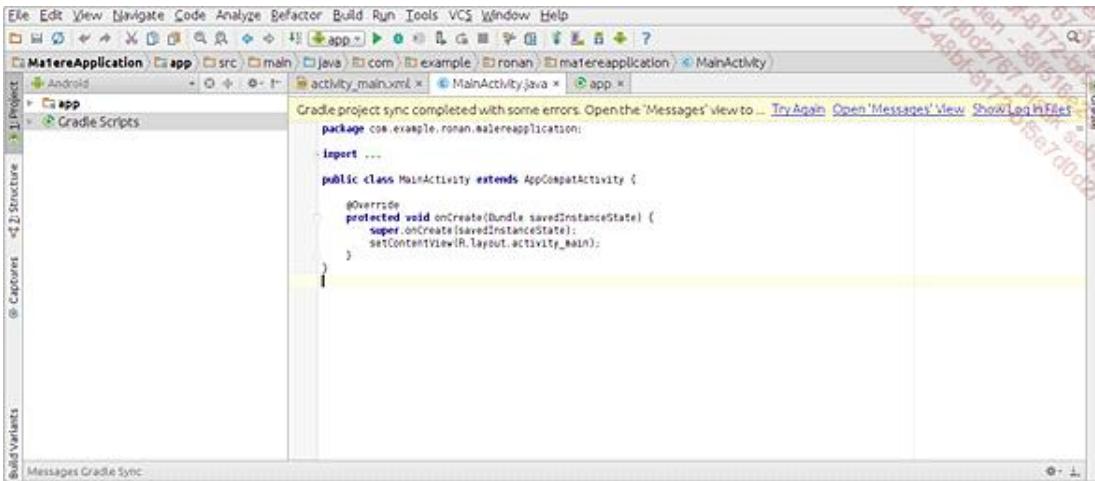
Pour faire très simple, une activity fait référence à l'affichage des fenêtres au sein d'une application. Dans notre cas de figure, nous souhaitons quelque chose de très basique.

- Sélectionnez **Empty Activity** indiquée en bleu et cliquez sur **Next**.

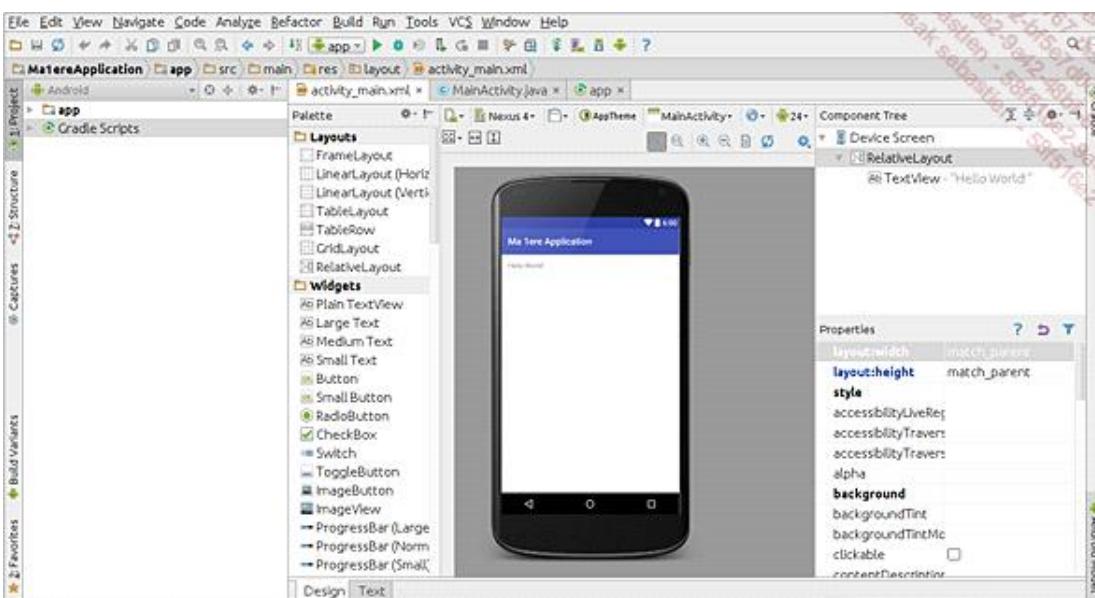


- Comme nous ne souhaitons pas la personnaliser, cliquez directement sur **Finish**.

Vous devriez alors obtenir l'écran suivant :

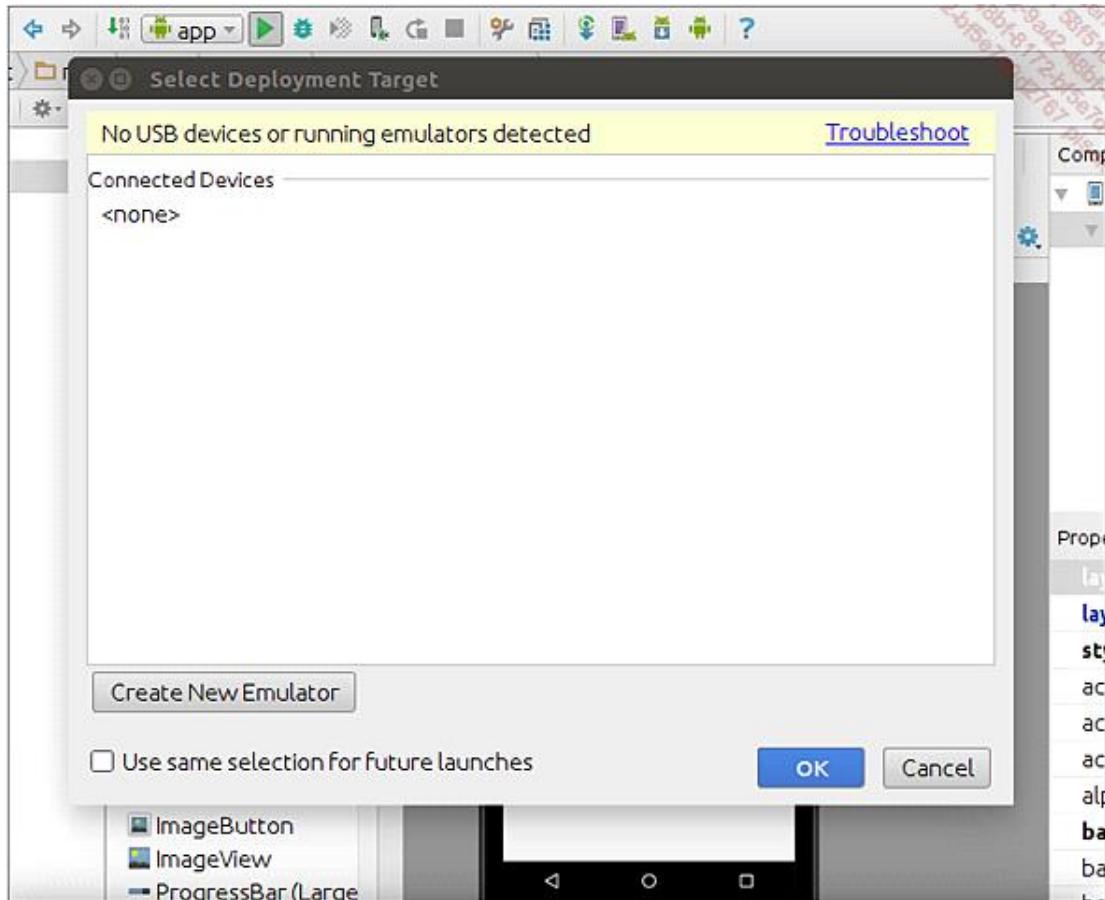


- Une fois que vous accédez à cet écran, cliquez sur l'onglet **activity_main.xml**. Vous devriez alors afficher l'écran suivant :

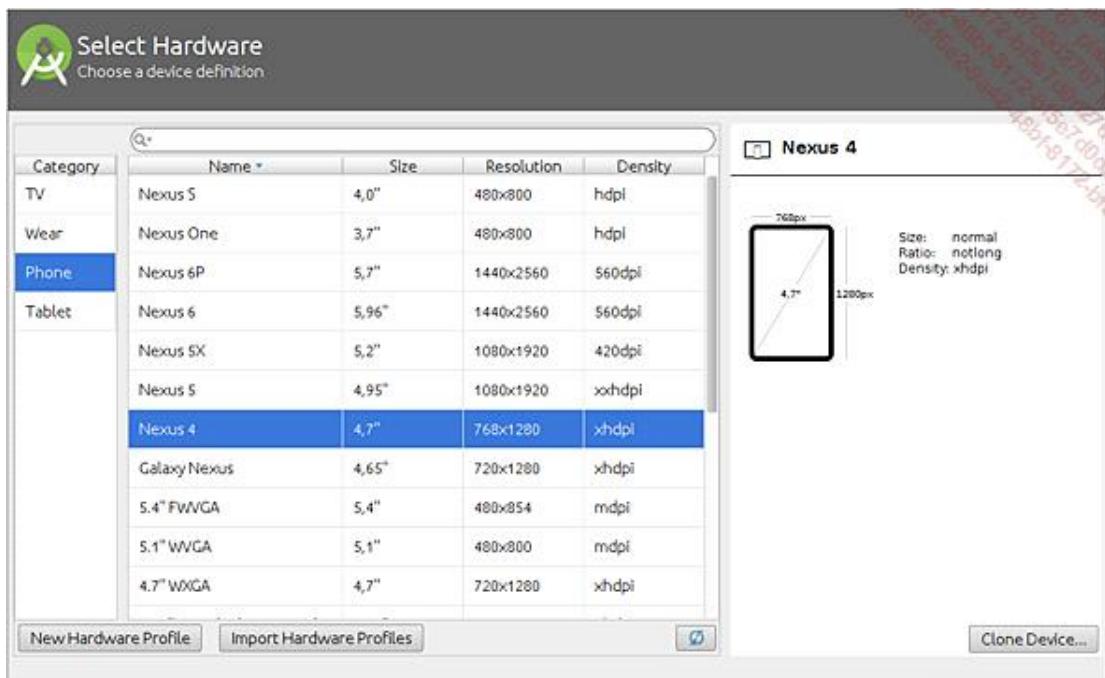


Cet espace va vous permettre de décider ce que vous souhaitez inclure à l'intérieur de votre application mobile. C'est également ici que vous pourrez avoir un aperçu du développement de cette dernière.

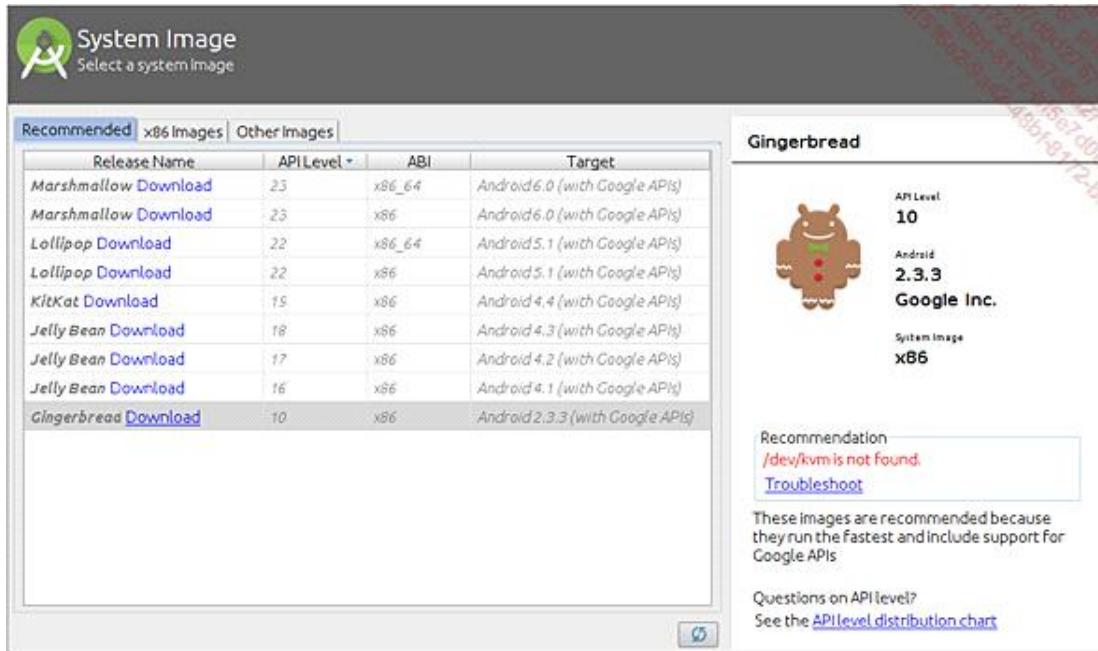
- Vous pouvez choisir le modèle précis d'appareil sur lequel vous souhaitez voir votre application être développée, ici le Nexus 4.
- Vous pouvez également lancer l'émulateur en cliquant sur la petite flèche verte :



→ Sélectionnez l'appareil de votre choix pour émuler l'application :

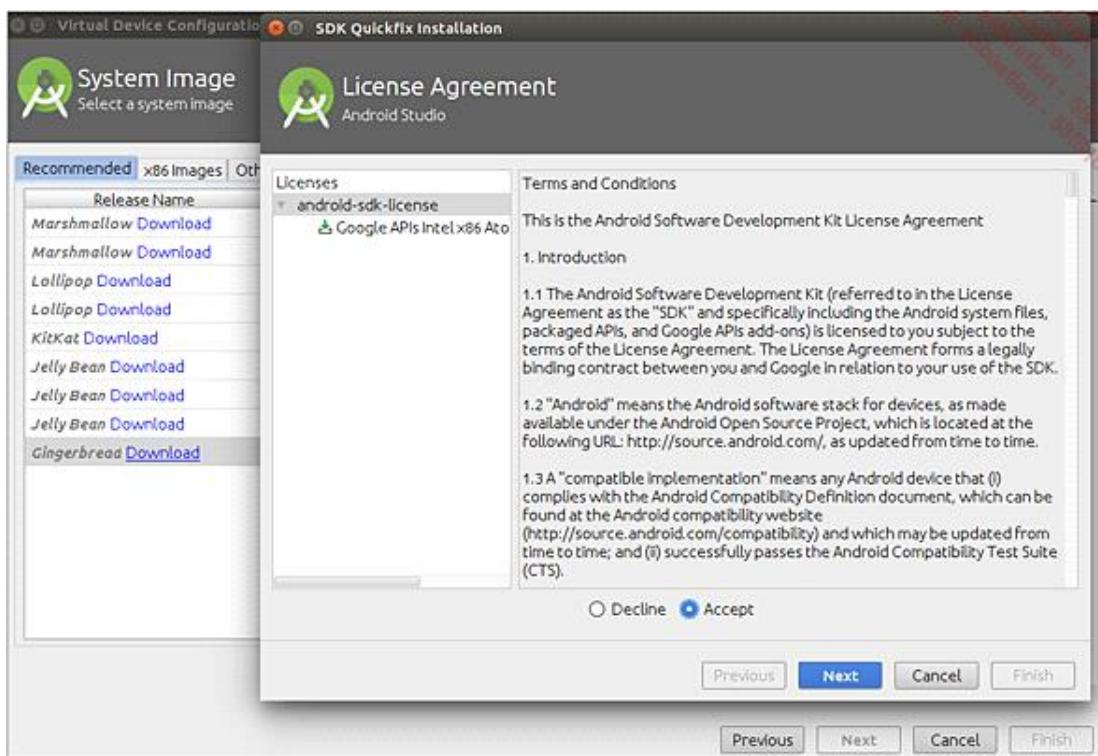


→ Choisissez ensuite le système d'exploitation de l'appareil à émuler, nous vous conseillons également ici de prendre une ancienne version du système d'exploitation :

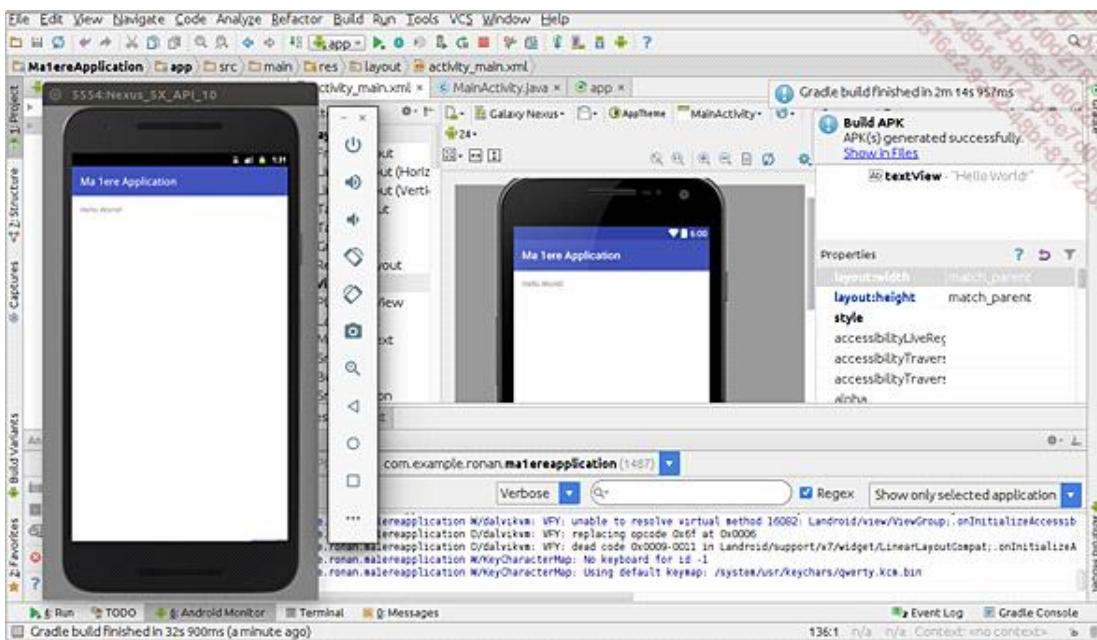


➤ À noter que si, comme l'écran ci-dessus, vous obtenez le message **/dev/kvm is not found**, cela signifie que votre ordinateur ne permet pas l'exécution de machine virtuelle pour le moment. Pour activer cette fonctionnalité, il faudra vous rendre dans le BIOS de votre ordinateur et l'activer.

→ Une fois la version sélectionnée, cliquez sur **Download**. Comptez une dizaine de minutes pour le téléchargement et l'installation du système d'exploitation. Cliquez ensuite sur **Next**, puis sur le bouton **Finish** au prochain écran :

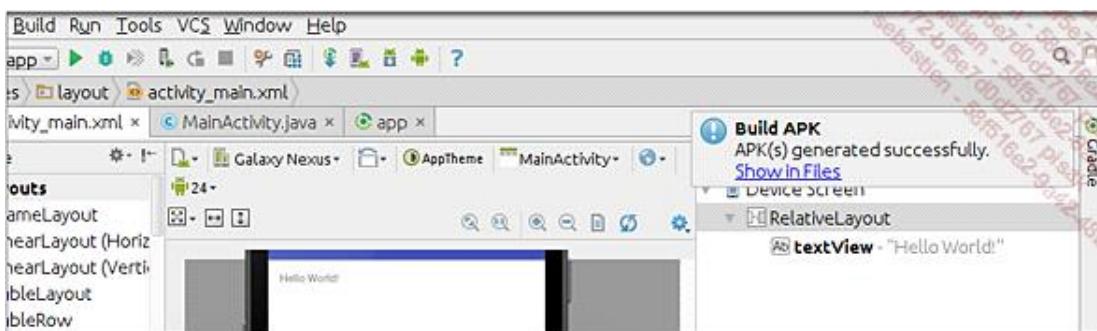


Une fois cela fait, vous devriez pouvoir accéder directement à l'émulateur de votre application :

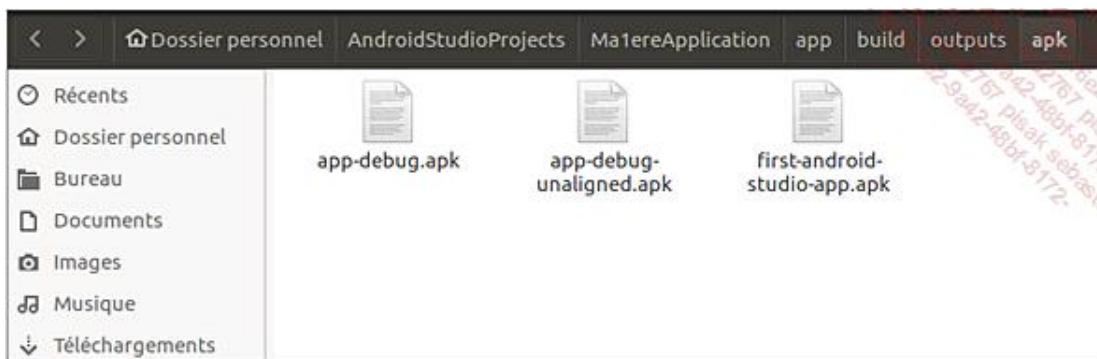


- Pour créer directement cette application et l'installer sur votre propre téléphone, à la manière d'App Inventor 2, cliquez tout en haut de l'écran sur **Build** puis sur **Build APK**.

À noter que, par défaut, Android Studio la mettra dans un répertoire spécifique et non dans dossier Download :



Dans notre exemple, il s'agit de :

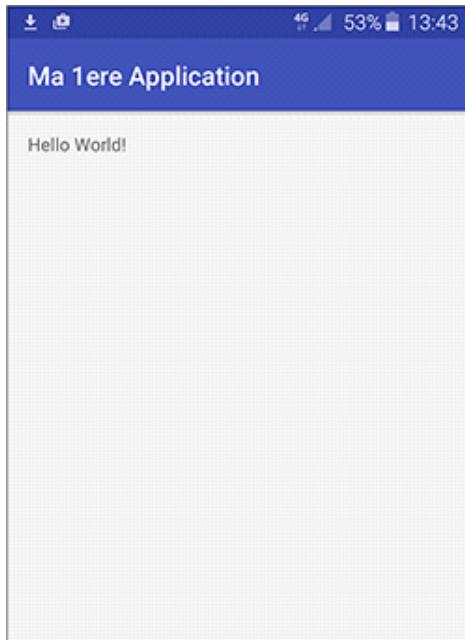


À noter que votre fichier portera le nom de **app-debug.apk**, cela signifie que votre application n'est pas encore signée, mais cela ne vous empêchera pas de l'installer directement sur votre téléphone.

- Pour ce faire, ouvrez simplement le fichier .apk depuis votre téléphone en le connectant à votre ordinateur ou en vous l'envoyant par e-mail. Le système d'installation est similaire à celui d'App Inventor 2 c'est-à-

dire qu'il vous faudra accepter d'installer une application d'une source inconnue.

Vous possédez désormais votre première application Android créée depuis Android Studio, félicitations !



Un SDK ou un kit de développement, est un ensemble d'outils permettant de développer pour une cible particulière. Par exemple, pour développer pour une console de jeu vidéo, il est possible d'utiliser un SDK spécifique pour développer des applications pour cette console. Le SDK Android est donc un ensemble d'outils que met à disposition Google afin de vous permettre de développer des applications pour Android.

3. Mise en place de Google Analytics sur Android Studio

Pourquoi s'intéresser à Google Analytics ?

Google Analytics est l'outil d'analyse de données le plus populaire au monde. C'est également l'un des services les plus utilisés par les gestionnaires de sites internet non développeurs. Avec App Inventor 2, il n'existe malheureusement pas de techniques simples et surtout fiables pour installer Google Analytics dans vos applications. Résultat, de nombreux profils sur le marché du travail identifient la mise en place de Google Analytics et le suivi des applications pour mobile comme une activité ne les concernant pas, alors qu'il s'agit pourtant de projets pour lesquels ils devraient être impliqués.

Pour ce faire, nous allons suivre la documentation officielle de Google :
<https://developers.google.com/analytics/devguides/collection/android/v4/>

Étape 1

- Pour démarrer ce tutoriel, vous aurez besoin de créer un nouveau projet dans Android Studio : **File - New - New Project**.
- Indiquez alors le nom de votre choix pour l'application et sélectionnez **empty activity**.

La prochaine étape consiste à modifier le fichier `AndroidManifest.xml`, ce fichier permet d'indiquer les composants qu'il y a à l'intérieur de votre application ainsi que les permissions pour que les autres applications de votre téléphone puissent utiliser les composants de votre application ; pour faire simple il s'agit d'un fichier de liaison.

En principe votre fichier AndroidManifest.xml ressemble à :

The screenshot shows the Android Studio interface. The left sidebar displays the project structure under 'app': 'manifests' contains 'AndroidManifest.xml'; 'java' contains 'MainActivity'; 'res' is empty; 'Gradle Scripts' contains 'build.gradle'. The right panel shows the XML code of 'AndroidManifest.xml'. The code includes the manifest declaration with package 'com.example.ronan.googleanalyticssuccessversion', an application section with activity 'MainActivity' (set as launcher), and intent filters for MAIN and LAUNCHER categories.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.ronan.googleanalyticssuccessversion">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="GoogleAnalyticsSuccessVersion"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

→ Vous devez le modifier afin qu'il inclue les lignes suivantes :

```
<uses-permission android:name="android.permission.INTERNET" />

<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
```

juste avant le début de la balise <application>, ce qui vous donnera :

The screenshot shows the modified 'AndroidManifest.xml' code in the Android Studio editor. The two new lines have been added just before the start of the 'application' tag:

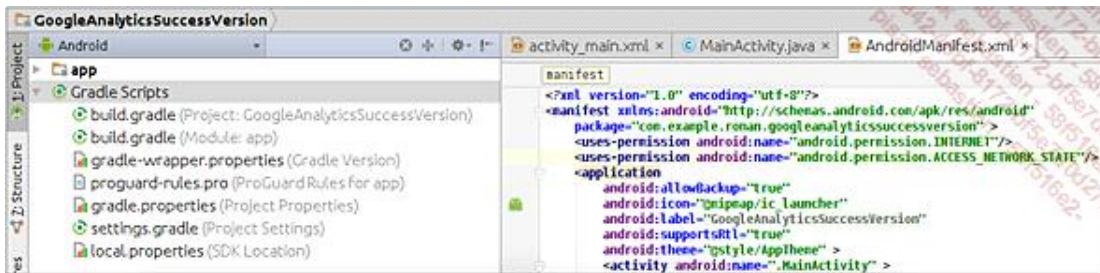
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.ronan.googleanalyticssuccessversion">
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="GoogleAnalyticsSuccessVersion"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Comme vous pouvez le lire, ces deux lignes permettent d'autoriser l'application à utiliser Internet et à accéder à l'état du réseau.

Étape 2

Une fois cette première étape terminée, vous devez inclure ce que l'on appelle le plug-in Google Services qui permet d'inclure des fonctionnalités essentielles à l'utilisation des services de Google. Pour cela, il faut effectuer deux modifications au niveau des fichiers build.gradle :



- La première va s'effectuer dans le fichier build.gradle du projet (ici Project : GoogleAnalyticsSuccessVersion), il vous faut alors ajouter la ligne de code suivante :

```
classpath 'com.google.gms:google-services:3.0.0'
```

Ce qui donne :

```
// Top-level build file where you can add configuration options common to all sub-projects/modules.

buildscript {
    repositories {
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:2.1.3'
        classpath 'com.google.gms:google-services:3.0.0'
        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}

allprojects {
    repositories {
        jcenter()
    }
}

task clean(type: Delete) {
    delete rootProject.buildDir
}
```

- La deuxième dans le fichier build.gradle de votre application ici (Module:app), vous devez ajouter la ligne suivante à la fin du fichier :

```
apply plugin: 'com.google.gms.google-services'
```

Ce qui donne :

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 24
    buildToolsVersion "24.0.2"

    defaultConfig {
        applicationId "com.example.ronan.googleanalyticssuccessversion"
        minSdkVersion 10
        targetSdkVersion 24
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'
    compile 'com.android.support:appcompat-v7:24.2.1'
}
apply plugin: 'com.google.gms.google-services'
```

Vous devez ensuite ajouter la ligne suivante :

```
compile 'com.google.android.gms:play-services-analytics:9.2.0'
```

de la manière suivante :

```
apply plugin: 'com.android.application'

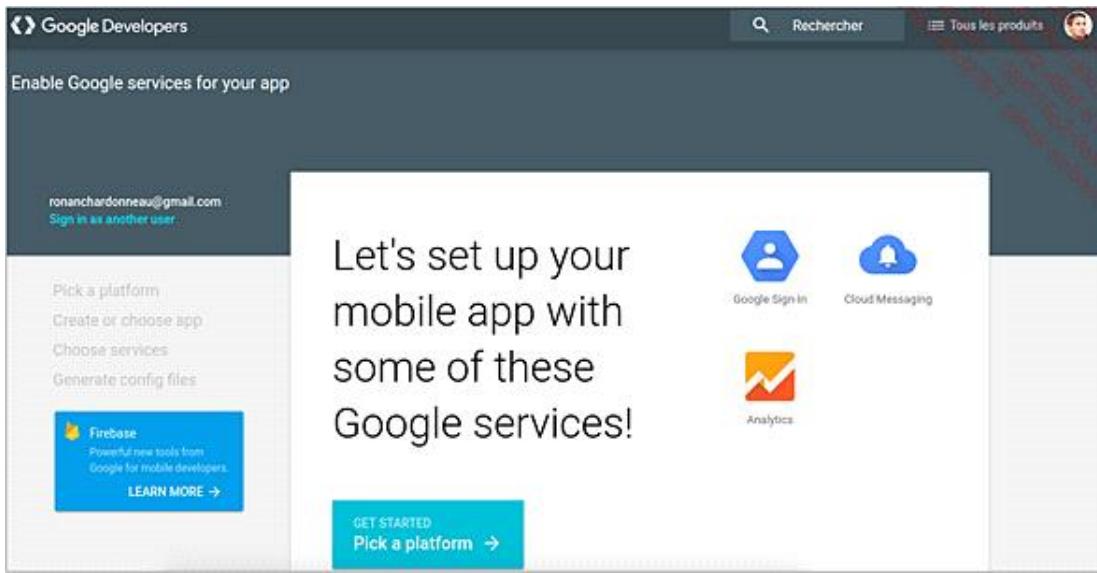
android {
    compileSdkVersion 24
    buildToolsVersion "24.0.2"

    defaultConfig {
        applicationId "com.example.ronan.googleanalyticssuccessversion"
        minSdkVersion 10
        targetSdkVersion 24
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'
    compile 'com.android.support:appcompat-v7:24.2.1'
    compile 'com.google.android.gms:play-services-analytics:9.2.0'
}
apply plugin: 'com.google.gms.google-services'
```

Étape 3

- Maintenant que les autorisations sont données et le plug-in mis en place, vous allez pouvoir télécharger le fichier de configuration Google Analytics en vous rendant à l'URL suivante :
<https://developers.google.com/mobile/add>



- Choisissez à l'étape suivante **Android App**, puis indiquez le nom d'un projet que vous avez créé. Pour créer un projet rendez-vous à l'URL suivante : <https://console.developers.google.com>

Il faudra également renseigner le nom du package, par exemple :
com.googleanalyticssuccessversion.android

Create or choose an app

App name: **GoogleAnalyticsSuccess** ✓ Services will be added to your existing project in the [Google Developers Console](#).

Android package name: **com.googleanalyticssuccessversion.android**

Share your [Google Mobile Developer Services](#) data with Google to help improve Google's products and services. This includes sharing with Google technical support, account specialists, and anonymous data for benchmarking. If you disable this option, data can still flow to other Google products that are explicitly added.

Your country/region: France

CONTINUE TO Choose and configure services →

- Cliquez alors sur **CONTINUE TO Choose and configure services** et sélectionnez la propriété Google Analytics que vous avez créée en amont :



Google Sign-In



Analytics



Cloud Messaging

Analytics

Measure everything about your mobile app from first discovery to in-app purchases.

[LEARN MORE](#)



Google Analytics Account (Required)

00 - My First App Tracking

Analytics Property (Required)

00 - My First App Tracking

ENABLE ANALYTICS SERVICE

- Cliquez alors sur **ENABLE ANALYTICS SERVICE** puis, dans le nouvel écran qui va apparaître, sur **CONTINUE TO Generate configuration**. Cliquez ensuite sur **Download google-services.json** :

Download and install configuration

Download google-services.json
for com.googleanalyticssuccessversion.android

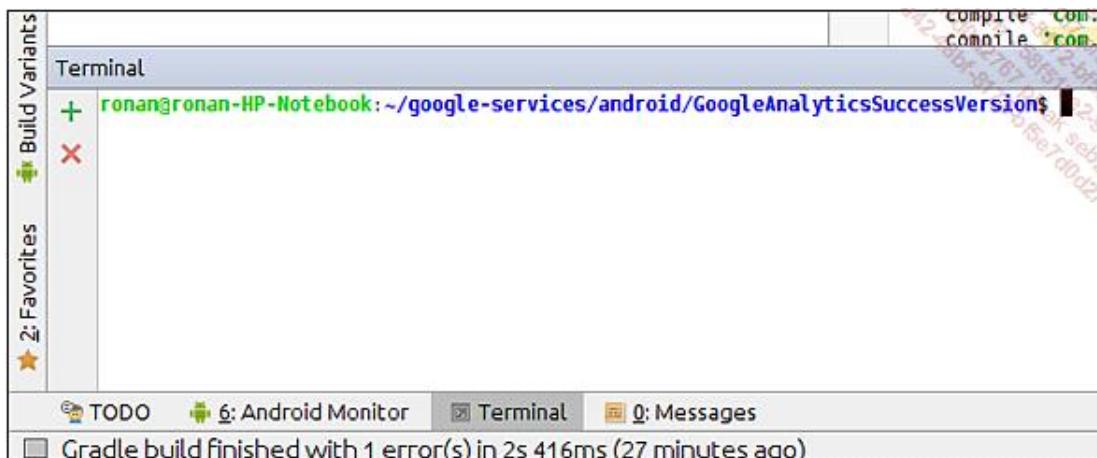
The file contains configuration details, such as keys and identifiers, for the services you just enabled. After downloading, copy the google-services.json to the **app/ or mobile/ module directory** in your Android project.

Implement your new services

Your tracking ID:
UA-83962542-1

Documentation

Une fois ce fichier téléchargé, il faut l'inclure à votre projet Android Studio, pour ce faire, vous avez à votre disposition le terminal d'Android Studio :



- Vous trouverez celui-ci en cliquant en bas à gauche sur l'onglet **Terminal**.
- Une fois dans celui-ci, l'objectif va être de déplacer votre fichier de configuration dans le répertoire /app.

Si vous êtes sur MAC ou Linux, vous aurez une ligne de commande de type :

```
mv path-to-download/google-services.json app/
```

Si vous êtes sur Windows :

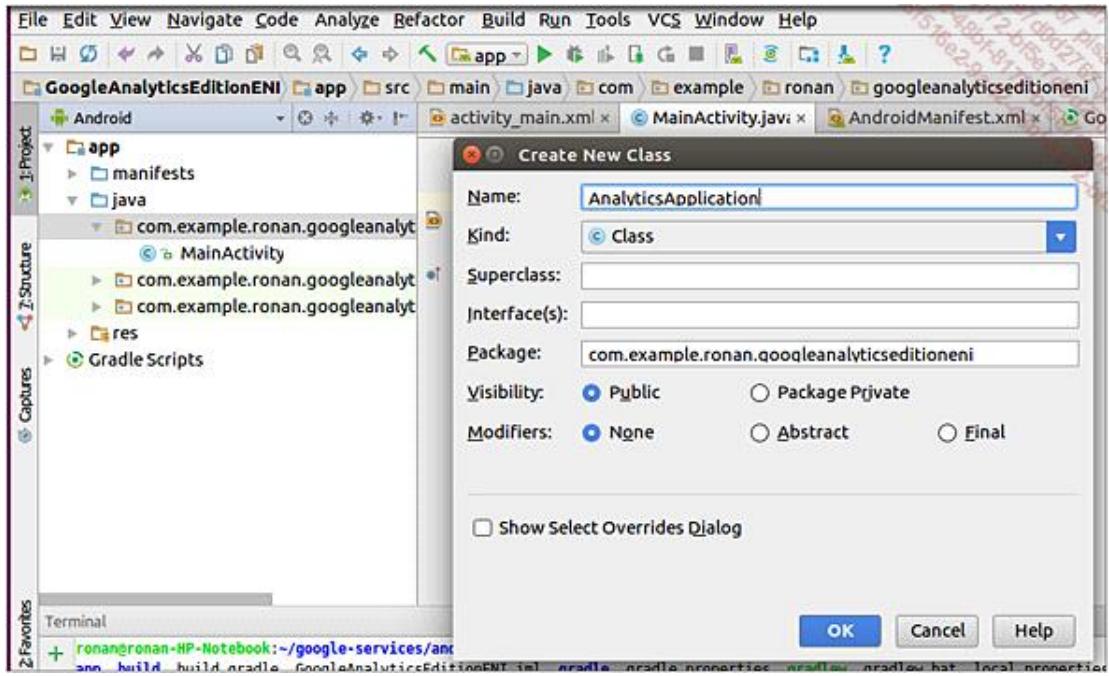
```
move path-to-download/google-services.json app/
```

À noter que path-to-download dépend de l'endroit où se situe le fichier sur votre ordinateur, dans notre cas de figure, sous Linux, l'instruction à entrer a été :

```
sudo mv /home/ronan/Bureau/Downloads/google-services.json /app
```

Une fois le fichier google-services.json dans le répertoire /app, il vous suffit de mettre dans votre application le tracking de Google Analytics. Dans notre exemple, nous souhaitons simplement faire remonter le fait que l'écran d'accueil de notre application est chargé.

- Vous allez, pour ce faire, vous rendre dans votre répertoire java où se situe le fichier MainActivity.java, créer un fichier .java (clic droit puis **New** et **Java class**) et donnez le nom **AnalyticsApplication** à votre classe :



→ Une fois ce fichier créé, copiez-collez celui de Google à l'URL suivante :

<https://developers.google.com/analytics/devguides/collection/android/v4/>, soit :

```

/*
 * Copyright Google Inc. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *     http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package com.example.ronan.googleanalyticseditioneni;

import android.app.Application;

import com.google.android.gms.analytics.GoogleAnalytics;
import com.google.android.gms.analytics.Tracker;

/**
 * This is a subclass of {@link Application} used to provide shared objects for this app, such as
 * the {@link Tracker}.
 */
public class AnalyticsApplication extends Application {
    private Tracker mTracker;

    /**
     * Gets the default {@link Tracker} for this {@link Application}.
     * @return tracker
     */
    synchronized public Tracker getDefaultTracker() {
        if (mTracker == null) {
            GoogleAnalytics analytics = GoogleAnalytics.getInstance(this);
            // To enable debug logging use: adb shell setprop log.tag.GAv4 DEBUG
            mTracker = analytics.newTracker(R.xml.global_tracker);
        }
    }
}

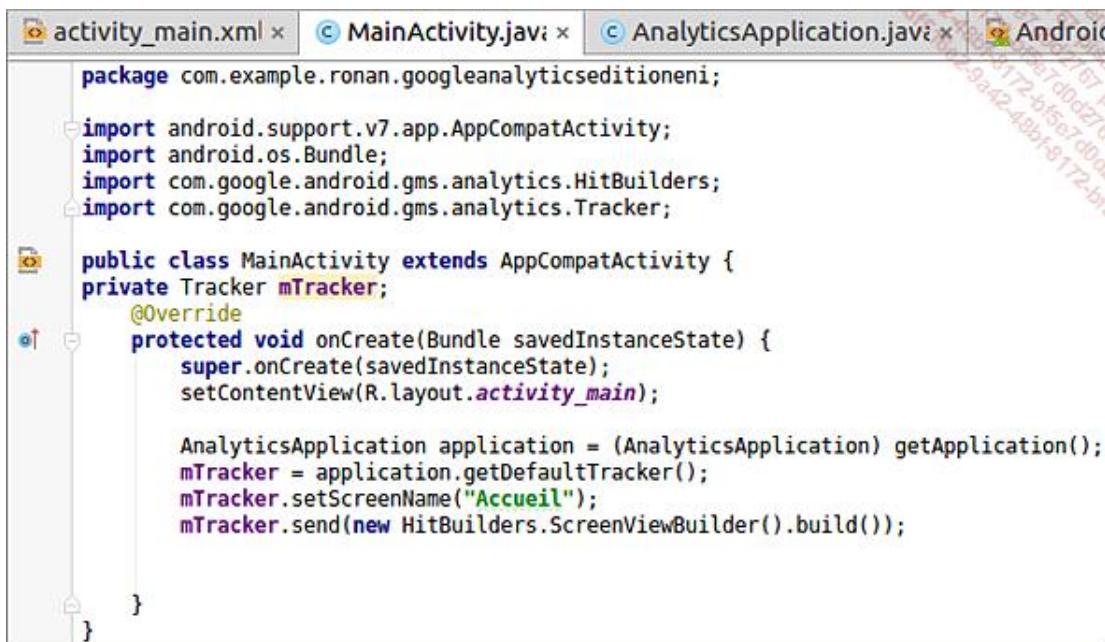
```

Attention : n'oubliez pas de changer le nom du package 'package com.google.samples.quickstart.analytics;' en fonction de votre application.

Étape 4

Modification du fichier MainActivity.java

- Il faut maintenant modifier le fichier MainActivity.java de la manière suivante :



```
activity_main.xml x MainActivity.java x AnalyticsApplication.java x Android
package com.example.ronan.googleanalyticseditioneni;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import com.google.android.gms.analytics.HitBuilders;
import com.google.android.gms.analytics.Tracker;

public class MainActivity extends AppCompatActivity {
    private Tracker mTracker;

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

        AnalyticsApplication application = (AnalyticsApplication) getApplication();
        mTracker = application.getDefaultTracker();
        mTracker.setScreenName("Accueil");
        mTracker.send(new HitBuilders.ScreenViewBuilder().build());
    }
}
```

- Une fois cela fait, enregistrez votre projet et cliquez dans le menu **Build** sur **Build APK**. Vous pouvez alors transférer l'application d'Android Studio sur votre smartphone. Une fois l'application lancée, vous devriez voir la donnée suivante remonter dans Google Analytics :



Comme vous aurez pu le constater dans ce tutoriel, l'utilisation d'Android Studio est bien plus compliquée que celle d'App Inventor 2 et s'adresse à des utilisateurs connaissant le langage de programmation JAVA.

Keosu

Si l'utilisation de Google Android Studio vous semble trop complexe, sachez que d'autres plateformes voient le jour. C'est le cas de Keosu, une plateforme de CMS open source hybride (Android, iOS, Windows Phone) <https://keosu.com/>.