TUTORIEL QUIZZME

L'utilisateur répond à une série de question en cliquant sur un bouton pour soumettre sa réponse et sur un autre pour passer à la question suivante.

Louise Henninot - Anne-Cécile Patou - Julie Roquefort M2 IPM 2013-14

Sommaire

Compétences Développées :

- Associer une réponse à une question en particulier.
- Le séquencement par une liste utilisant une variable.
- Comportements conditionnels.
- La commutation d'une image pour montrer une image différente à chaque fois.

A l'issu de ce tutoriel, le concepteur pourra de façon autonome aller plus loin dans cette application en programmant par exemple :

- Un TakeQuizz MakeQuizz.

Intérêts pédagogiques de l'application :

- Créer des quizz sur un sujet vu en formation.

Dans le designer…

Objectif

- Aller plus loin dans les fonctionnalités du Designer et de l'Éditeur, introduire la notion de « gestionnaire d'événements » qui servira à l'utilisateur s'il désire créer des quizz plus poussés ou MakeQuizz - TakeQuizz

Pour démarrer la création de votre application QuizzMe, allez dans My Projects et cliquez sur New. Nommez votre projet : QuizzMe.

Cliquez sur OK, le Designer s'ouvre.

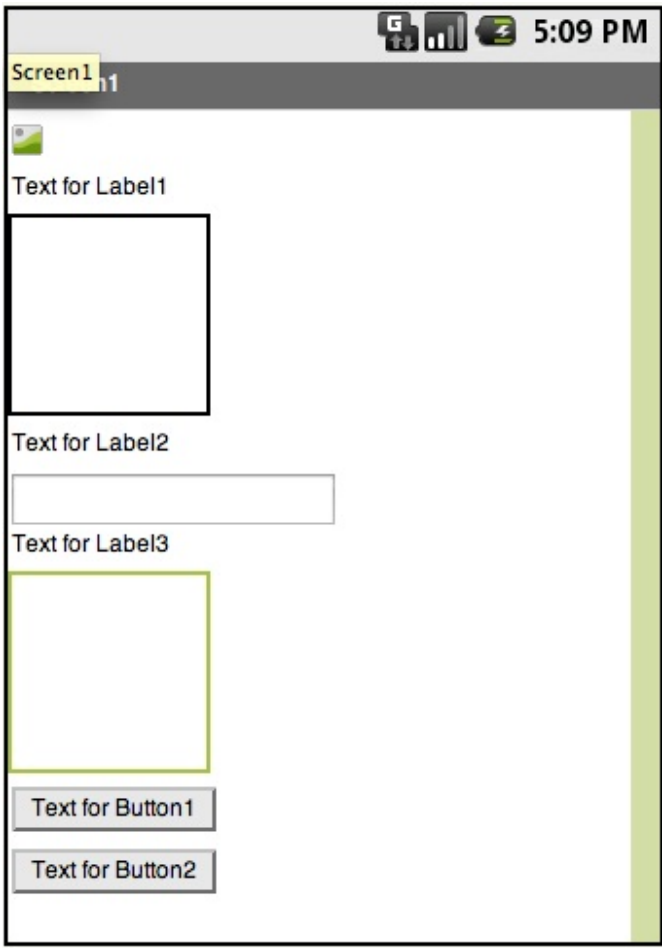
Mise en place des éléments

Il vous faut d'abord mettre en place dans le Screen les différents composants qui seront nécessaires à votre application. Vous les trouverez dans Palette. Vous devez également les renommer dans Components. Toutes les informations dont vous avez besoin sont dans ce tableau :

Type d'élément	Où le trouver dans la palette	Comment le renommer dans Components	Raison
Image	Basic	**Image1**	L'image de la question
Label	Basic	**Question**	Afficher la question en cours
Horizontal Arrangement	Screen Arrangement	**Horizontal Arrangement1**	Organiser les éléments Zone de Réponse et Réponse
Label	Basic	**ZoneDeReponse**	Étiquette de la réponse
TextBox	Basic	**Reponse**	Zone d'écriture de la réponse
Label	Basic	**Vrai_Faux**	Vrai/faux est affiché ici
Horizontal Arrangement	Screen Arrangement	**Horizontal Arrangement2**	Organiser les boutons Réponse et Suivant
Button	Basic	**BoutonReponse**	L'utilisateur clique pour soumettre sa réponse
Button	Basic	**BoutonSuivant**	L'utilisateur clique pour passer à la question suivante

Dans Media, cliquez sur Upload New et télécharger l'image Image1 et Image2 fournies avec ce tutoriel.

Votre Screen se présente comme cela :

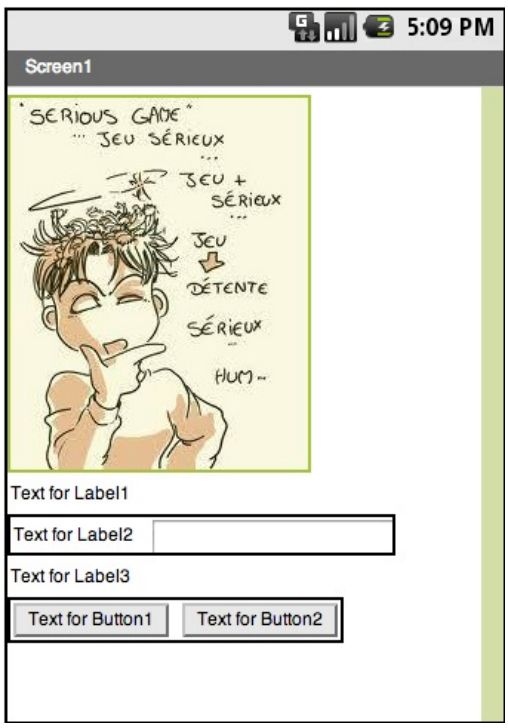


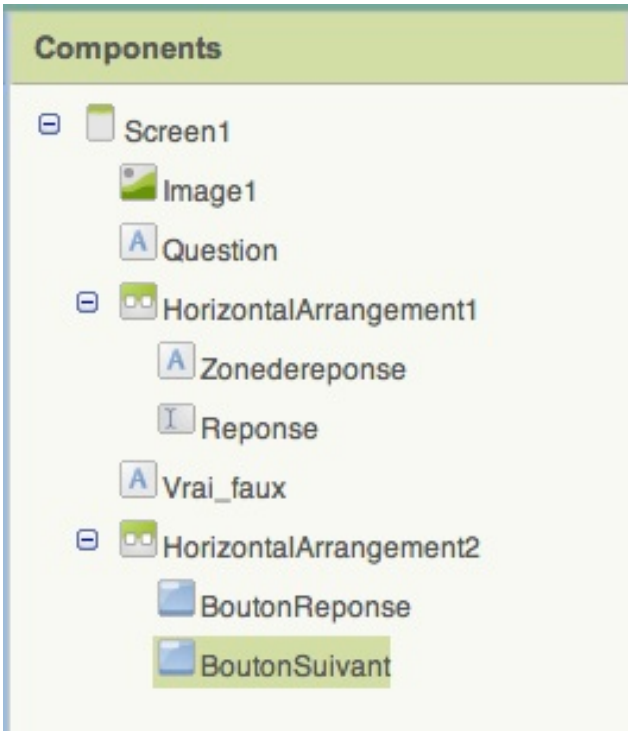
Agencer et attribuer des propriétés aux éléments

Vous allez maintenant agencer les éléments dans le Screen et attribuer les propriétés dans Properties comme ceci :

Éléments	Action
Image1	Définir cette image en y associant dans Properties l'image Image1 uploadée précédemment
Question	Changer le Text en « Question »
ZoneDeReponse	Changer le Text en « Écrivez la réponse ». Dans la partie Viewer, glisser cet élément dans HorizontalArrangement1.
Reponse	Changer Hint en « Veuillez entrer une réponse ». Dans la partie Viewer, glisser cet élément dans HorizontalArrangement1
BoutonReponse	Changer le Text en « Valider». Dans la partie Viewer, glisser cet élément dans HorizontalArrangement2.
BoutonSuivant	Changer le Text en « Suivant». Dans la partie Viewer, glisser cet élément dans HorizontalArrangement2.
VraiFaux	Changer le Text en « Vrai/Faux ».

Votre Screen et votre partie Components se présentent maintenant comme cela :





Vous pouvez passer dans la partie Éditeur.

Dans l'Éditeur…

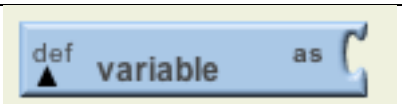
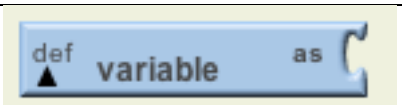
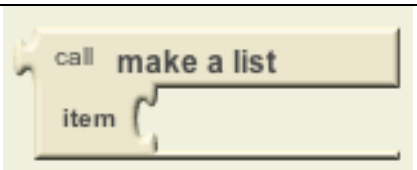
Objectifs

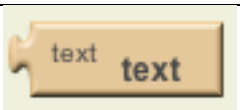
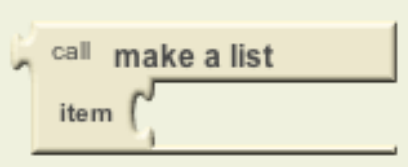
- Mettre en place les actions inhérentes à l'application.
- Définir les paramètres de comportement de mon application.

Pour créer une liste de questions et une liste de réponses

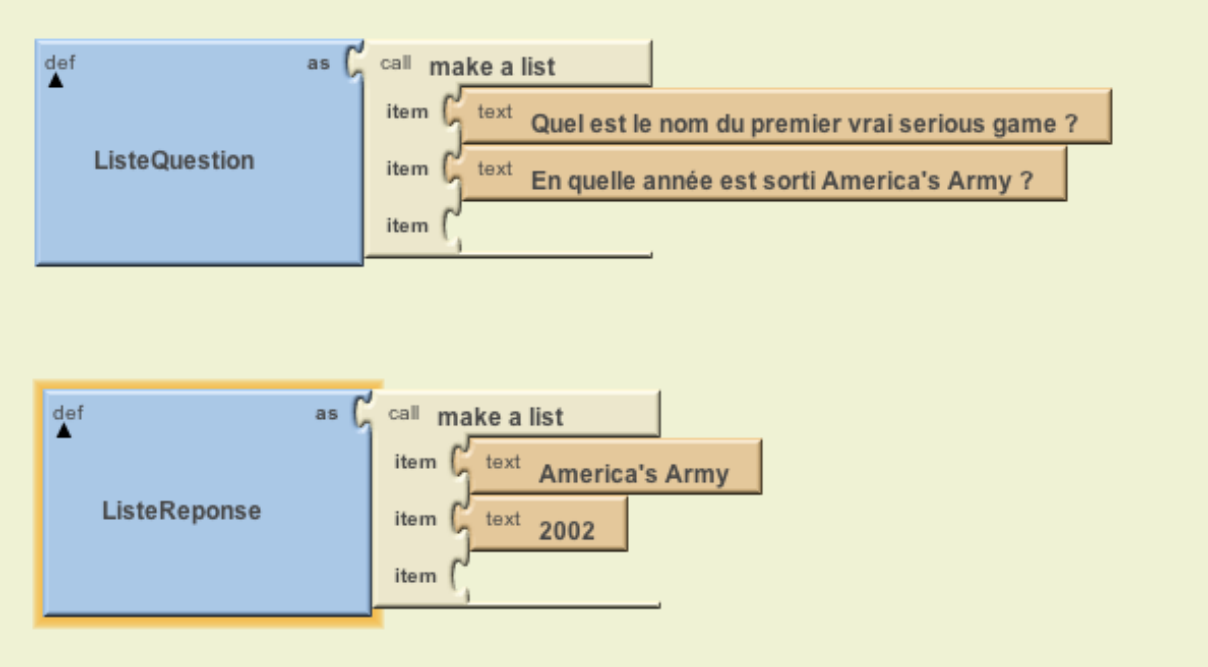
Dans le block d'édition, Vous allez avoir besoin de 2 listes de variables, une ListeQuestion pour contenir la liste des questions que vous souhaitez poser et une ListeReponse pour contenir les réponses correspondantes à chaque question.

Pour cela vous avez besoin des blocks suivants :

Type de block	Où le trouver dans l'Éditeur	Raison
	Definitions	Définir les variables de la ListeQuestion (le renommer une fois glissé dans la partie de travail)
	Definitions	Définir les variables de la ListeReponse (le renommer une fois glissé dans la partie de travail)
	Lists	Interface pour insérer les items des la ListeQuestion

	Text	La question posée
	Lists	Interface pour insérer les items des la ListeReponse
	Text	La réponse de la question posée

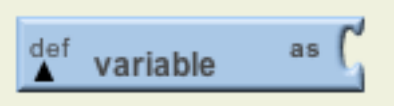
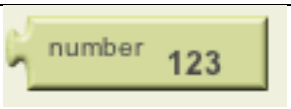
Puis installez les Blocks afin de créer ces deux puzzles :



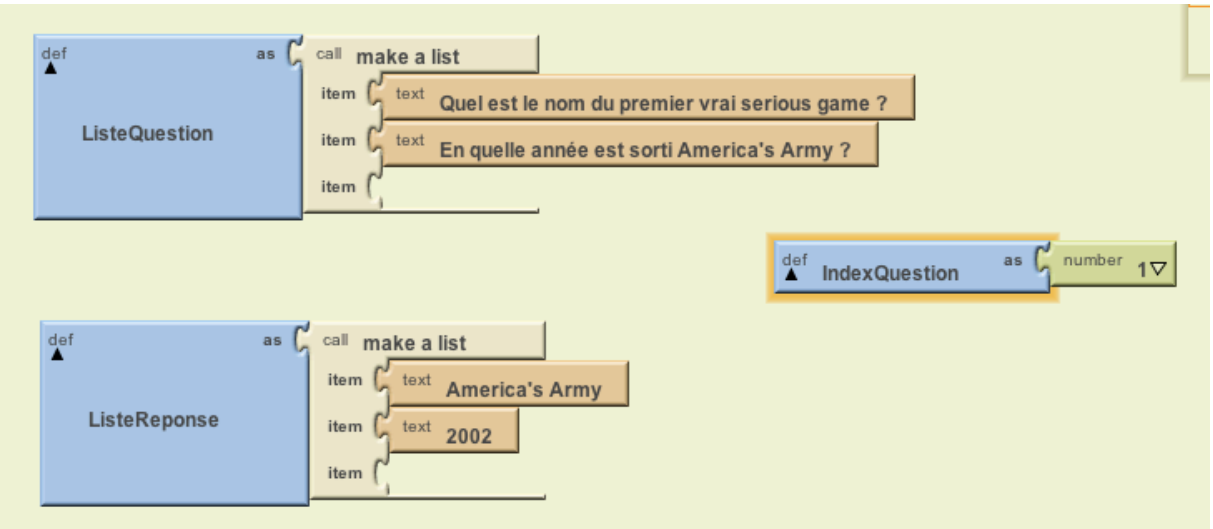
Définir l'index de variables

Chaque fois que l'utilisateur clique sur le bouton Suivant pour changer de question, l'application doit se souvenir quelle question est affiliée à quelle réponse.
Dans la programmation, pour se rappeler de cela, vous devez définir une nouvelle variable.
Dans ce cas, l'application doit se rappeler le numéro de la question actuelle c'est à dire l'indice de la ListeQuestion.

Pour créer la variable « IndexQuestion », vous aurez besoin des Blocks suivants

Type de block	Où le trouver dans l'Éditeur	Raison
	Definitions	Définir la variable IndexQuestion (la renommer une fois glissé dans la partie de travail)
	Math	Attribuer la valeur 1 à IndexQuestion

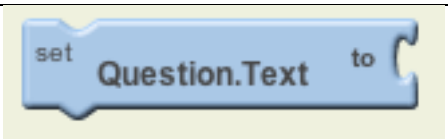
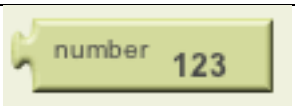
Puis installez les Blocks afin de créer ces deux puzzles :



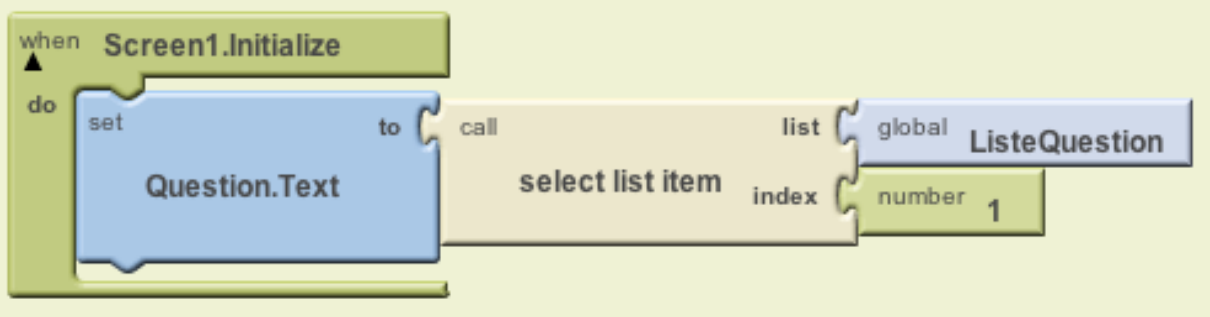
Afficher la première question :

Pour commencer, vous ignorez les réponses et travaillez juste sur le comportement séquentiel des questions.
Le comportement désiré est le suivant: quand l'application démarre la première question apparaît dans l'étiquette nommée Question.
Quand l'utilisateur clique sur le bouton Suivant, la deuxième question devrait apparaître.
Quand l'utilisateur clique de nouveau, la troisième devrait apparaître.
Quand la dernière question est atteinte, la première question réapparaît.

Pour faire cela, vous avez besoin des Blocks suivants :

Type de block	Où le trouver dans l'Éditeur	Raison
	Screen1	Quand l'application démarre, ce gestionnaire d'événement est déclanché
	Question	Il est nécessaire d'afficher la première question dans l'élément Question
	Lists	Il est nécessaire de sélectionner la première question depuis Question
	My Definitions	La liste dans laquelle aller chercher l'élément
	Math	Sélectionner la première question en utilisant l'index 1

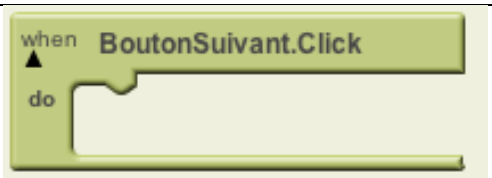
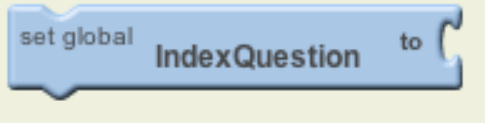
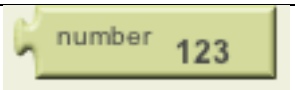
Puis installez les Blocks afin de créer ces deux puzzles :



Itération des questions :

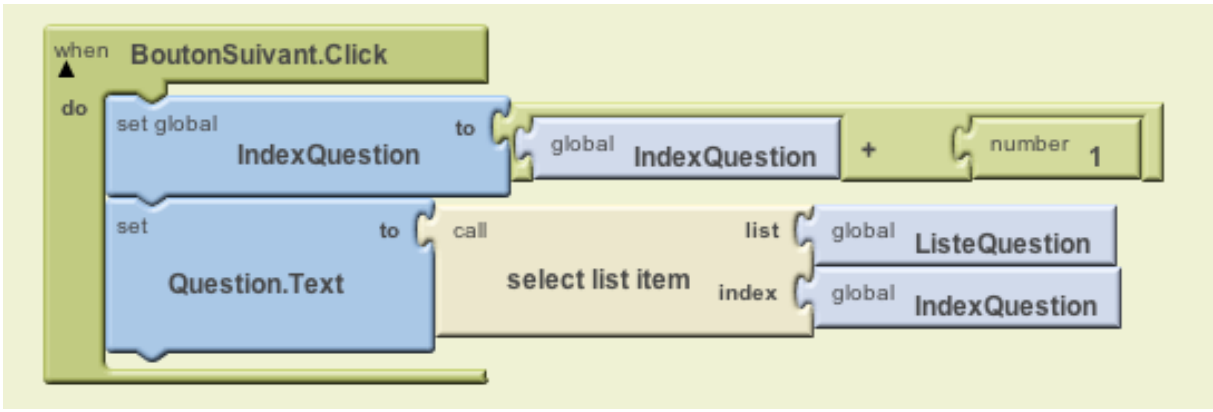
Vous allez maintenant programmer le comportement du bouton Suivant.
Vous avez déjà défini l'IndexQuestion pour rappeler la question sur laquelle l'utilisateur est. Quand le bouton Suivant est cliqué, l'application doit incrémenter cette variable, par exemple, passer de 1 à 2 ou de 2 à 3, etc et utiliser ensuite la valeur résultante pour choisir la nouvelle question.

Pour ce comportement, vous avez besoin des Blocks suivants :

Type de block	Où le trouver dans l'Éditeur	Raison
	BoutonSuivant	Quand l'utilisateur clique sur Suivant, le gestionnaire d'événement est déclenché.
	My Definitions	La première question doit apparaître dans Question
	Math	Incrémenter une valeur à IndexQuestion
	My Definitions	La nouvelle valeur est égale à la précédente valeur +1
	Math	Pour le +1
	Question	La nouvelle question doit s'afficher ici
	Lists	La première question est sélectionnée depuis la ListeQuestion
	My Definitions	Piocher dans la liste l'item appelée

	My Definitions	*Piocher* dans l'index de la liste d'items appelée

Puis installez les Blocks afin de créer ces deux rangées :



La première rangé de blocks incrémente la variable IndexQuestion. Si IndexQuestion équivaut à 1, il est changé en 2. S'il vaut 2, il est changé en 3, etc.
Une fois que la variable IndexQuestion a été changée, l'application l'utilise pour choisir la nouvelle question.

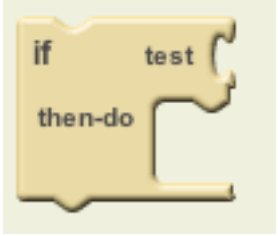
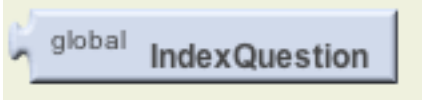
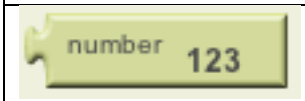
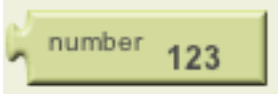
Les Blocks sont exécutés de la droite vers la gauche. L'application évalue d'abord le paramètre d'index de « Select list item », qui est la variable IndexQuestion.
Le numéro est stocké dans IndexQuestion et est utilisé comme l'index quand « select list item » est exécuté.

Quand le bouton Suivant est cliqué pour la première fois, les Blocks d'incrémentation passeront IndexQuestion de 1 à 2, donc l'application choisira le deuxième article de ListQuestion, « En quelle année est sorti America's Army ? ».

Le problème avec l'application est qu'elle incrémente toujours la variable IndexQuestion quand le bouton Suivant est cliqué. Quand IndexQuestion équivaut déjà à 2 et que l'utilisateur clique sur le bouton Suivant, l'application change IndexQuestion de 2 en 3.
Puisqu'il n'y a seulement que 2 questions dans la variable ListeQuestion, l'application ne comprend plus.

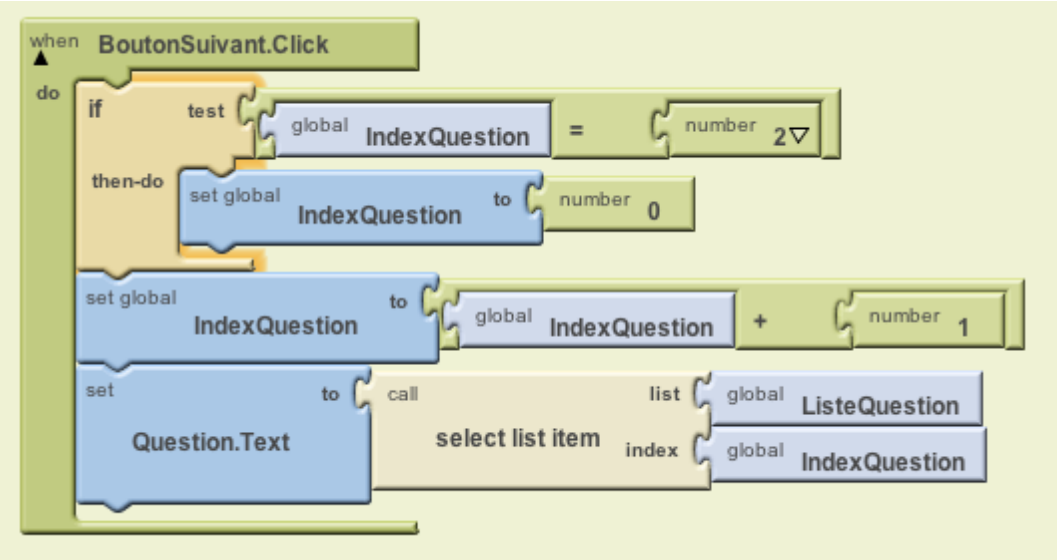
Ce que l'application doit faire est : poser une question + vérifier une condition quand le bouton Suivant est cliqué et exécuter différents blocks dépendants de la réponse.
Le problème se pose de la façon suivante : "est-ce que la variable IndexQuestion est déjà à 2 ?"
Si la réponse est oui, vous devez faire revenir IndexQuestion à 0 de façon à ce que l'utilisateur soit renvoyé à la première question.

Pour faire cela vous avez besoin des Blocks suivants :

Type de block	Où le trouver dans l'Éditeur	Raison
	Control	Demander si l'utilisateur est à la dernière question
	Math	Tester si IndexQuestion équivaut à 2
	My Definitions	
	Math	2 est le nombre d'items dans la liste
	My Definitions	Revenir à 0 et donc à la première question
	Math	Revenir à 0 parce que le block suivant va incrémenter à 1

Vous allez donc rajouter cette commande à l'application via le block NextButton.click que nous avons mis en place au tout début de ce tutoriel.

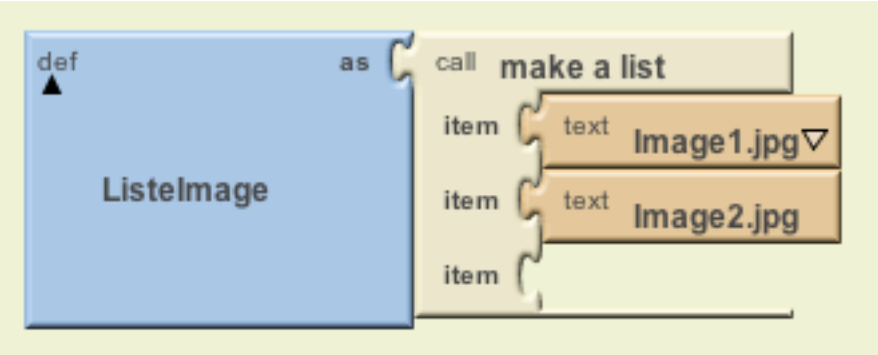
Le puzzle ressemble désormais à cela :



Faire changer l'image à chaque question

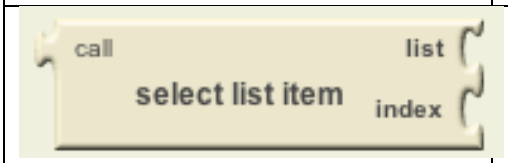
L'application actuelle montre la même image, peu importe la question.
Vous pouvez changer cela en spécifiant une image se rapportant à chaque question dès que le bouton Suivant est cliqué.
Précédemment vous avez uploadé 2 images via l'interface Medias du Designer.
Maintenant, vous allez créer une troisième liste, ListeImage, avec les noms des fichiers image.
Modifiez le bouton Suivant afin de faire changer l'image chaque fois que le bouton sera actionné.

D'abord, créez une ListeImage et spécifiez les noms des fichiers image.
Assurez que les noms soient exactement les mêmes que celui des fichiers qui ont été uploadés.
Voici ce à quoi les blocks doivent ressembler :

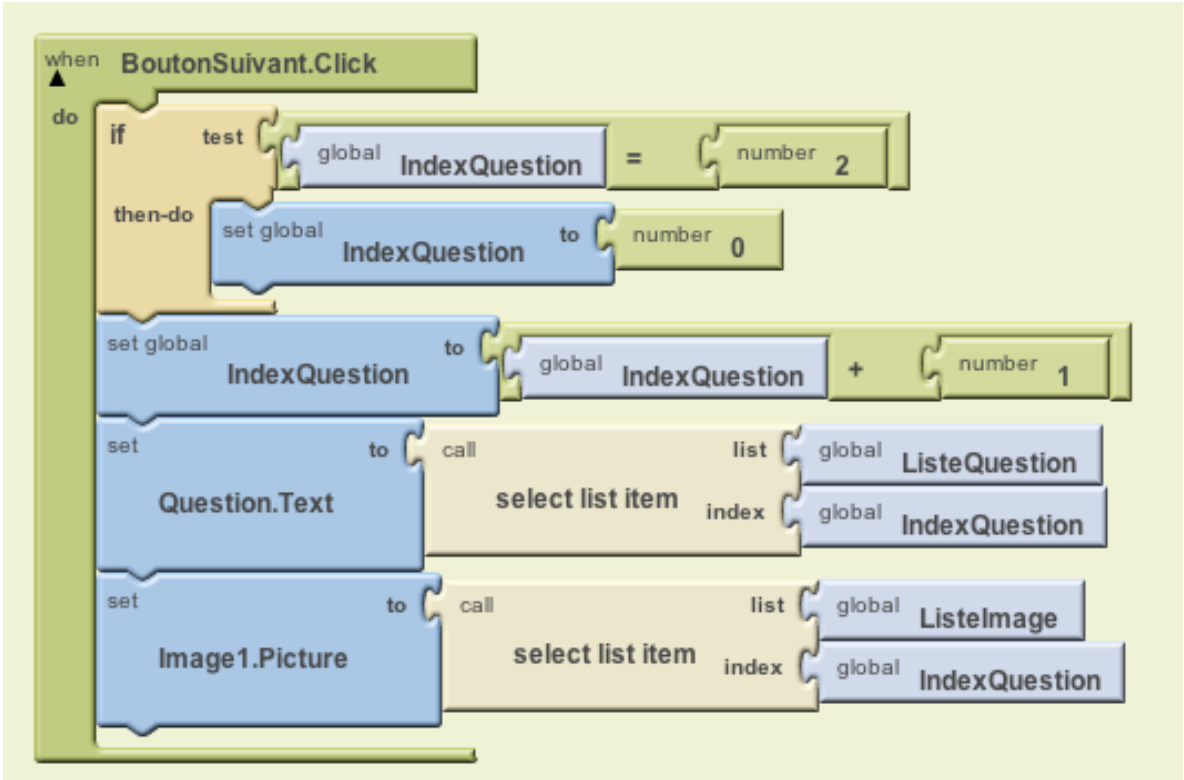


Ensuite, vous avez besoin de modifier le bouton Suivant afin que l'image dépendant de la question soit modifiée à chaque fois que l'utilisateur clique ce bouton.

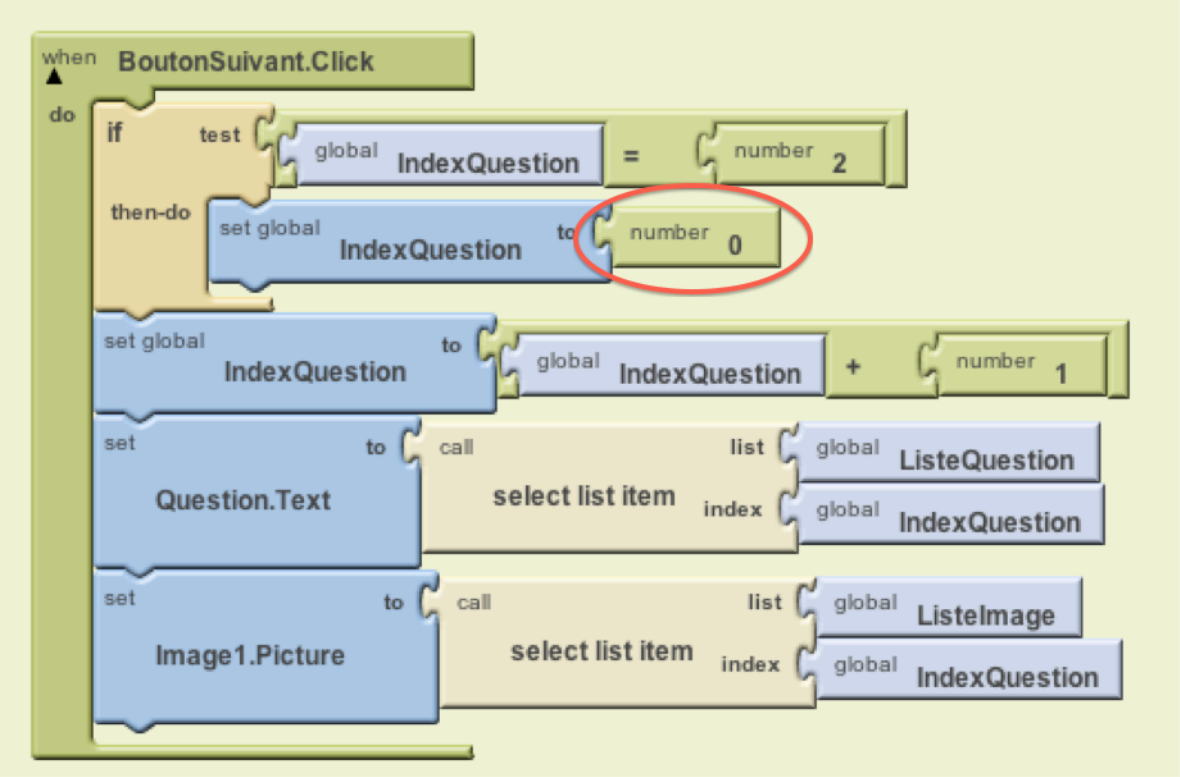
Utilisez les blocks suivants :

Type de block	Où le trouver dans l'Éditeur	Raison
	Image1	Définir le changement d'image
	Lists	Sélectionner l'image correspondant à la question en cours
	My Definitions	Sélectionner un nom de fichier de cette liste
	My Definitions	Sélectionner l'IndexQuestion item

Votre puzzle BoutonSuivant ressemble à cela désormais :



Afin de passer à l'étape suivante, vous allez expliquer à votre application comment afficher la bonne évaluation de réponse, comme nous allons le faire juste après. Pour cela, remplacez dans votre grand block BoutonSuivant.click le block number 0



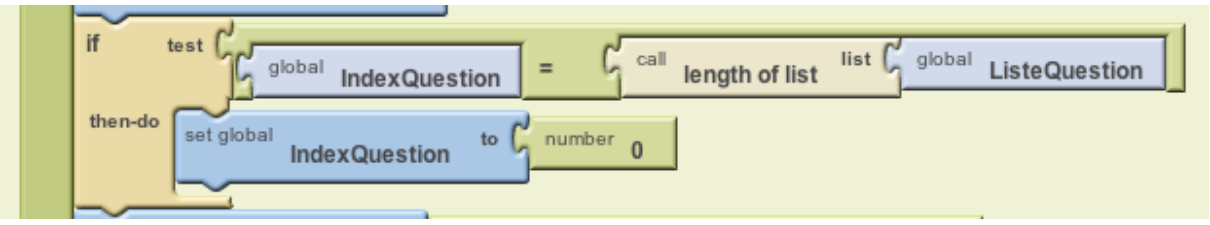
par ces 2 blocks :

que vous trouvez dans l'onglet List

et par le block 

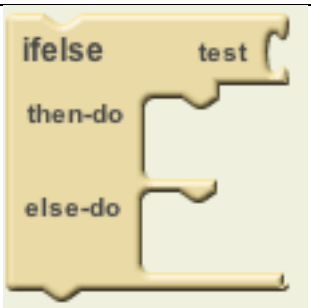
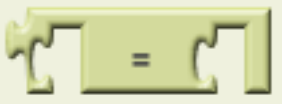
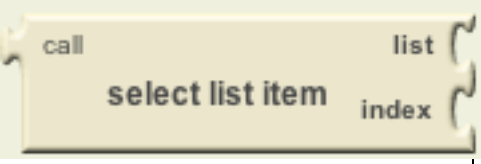
.

Cela donne l'extension suivante:



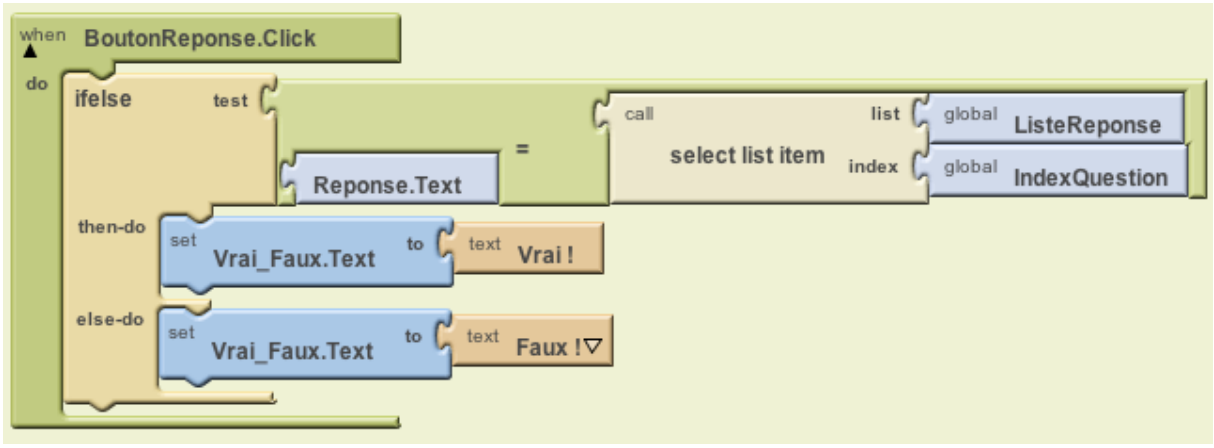
Évaluer la réponse

Vous devez ajouter des blocks qui indiqueront si l'utilisateur a répondu correctement ou non à la question posée.
L'utilisateur entre la réponse dans l'élément Reponse et clique ensuite sur le bouton Suivant.
L'application doit comparer l'entrée de l'utilisateur avec la réponse à la question en cours, utilisant un Block d' « if else » pour vérifier. Le Vrai/Faux devrait être modifié pour dire si la réponse est vraiment correcte.
Vous avez besoin des Blocks suivants pour obtenir ce comportement :

Type de block	Où le trouver dans l'Éditeur	Raison
	BoutonReponse	Le comportement est crée quand l'utilisateur clique sur le bouton Réponse
	Control	Si la réponse est correct : faire cela. Si la réponse est incorrect : faire ceci
	Math	Demander si la réponse est correcte
	Reponse	La réponse de l'utilisateur est dans cette zone de texte
	Lists	Sélectionner la réponse actuelle depuis ListeReponse
	My Definitions	La liste dans laquelle venir piocher

	My Definitions	Le numéro de la question et réponse actuelle
	Vrai_Faux	Reporter la réponse ici
 « Vrai »	Text	Si la réponse est vraie
	Vrai_Faux	Reporter la réponse ici
 « Faux »	Text	Si la réponse est fausse

Voici votre puzzle:



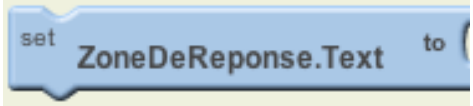
Le block « if else » lit : « Est-ce que la réponse de l'utilisateur est égal à l' item IndexQuestion de la ListeReponse ? ». Si l'IndexQuestion équivaut à 1, l'application va comparer la réponse de l'utilisateur avec le premier item de la ListeReponse.

Si l'IndexQuestion équivaut à 2, l'application va comparer la réponse de l'utilisateur avec le deuxième item de la ListeReponse et ainsi de suite.

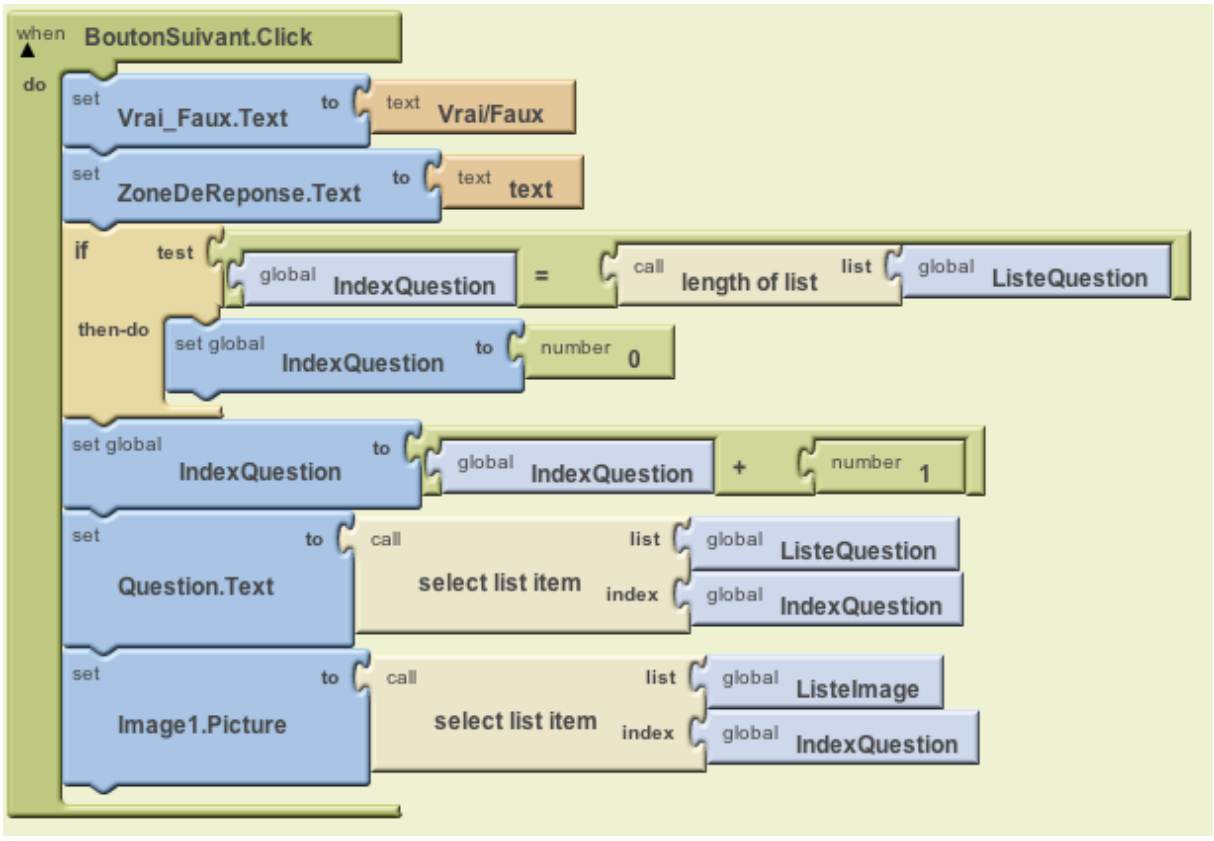
Si le résultat est positif, le « then-do » block est exécuté est le VraiFaux s'affiche en « Vrai ! ».

Si le résultat est négatif, le « then-do » block est exécuté est le VraiFaux s'affiche en « Faux ! ».

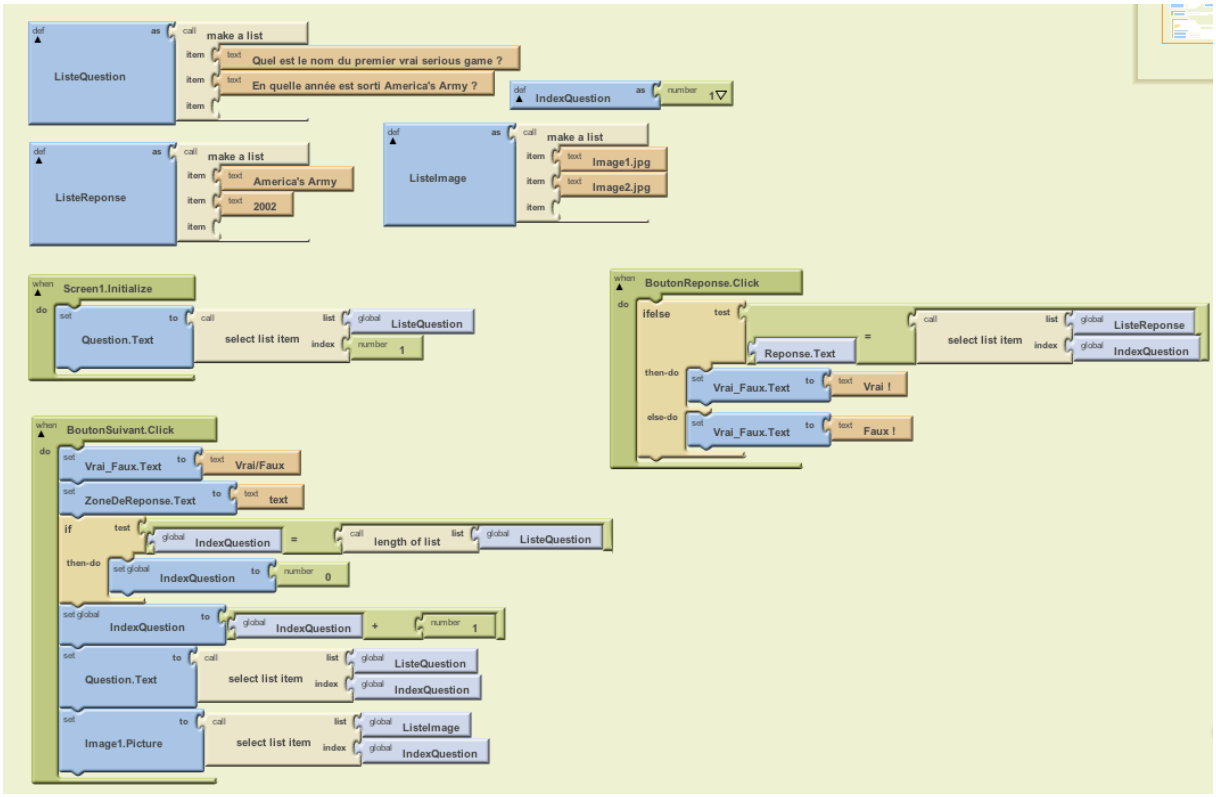
Pour masquer le Vrai/Faux et le Reponse, vous devez mettre les blocks suivants dans le grand block « when BoutonSuivant.click » :

Type de block	Où le trouver dans l'Éditeur	Raison
	Vrai_Faux	L'étiquette est masquée
	Text	Quand le bouton Suivant est cliqué, la critique de l'ancienne réponse est effacée
	Reponse	La réponse de l'utilisateur sur le précédente question
	Text	Quand le bouton Suivant est cliqué, l'ancienne réponse est effacée.

Voici mon résultat :



L’architecture de votre application est complète, cela donne un résultat global suivant :



Vous pouvez tester votre application dans l’Emulateur.