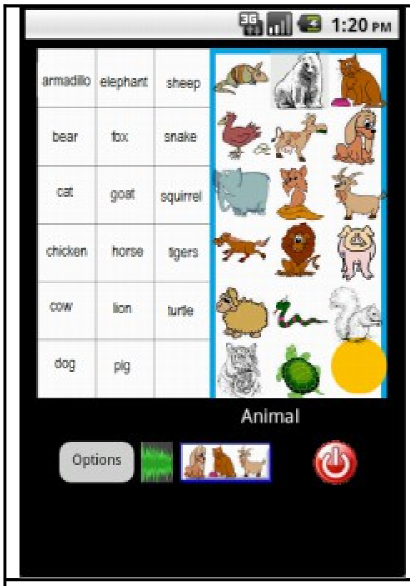
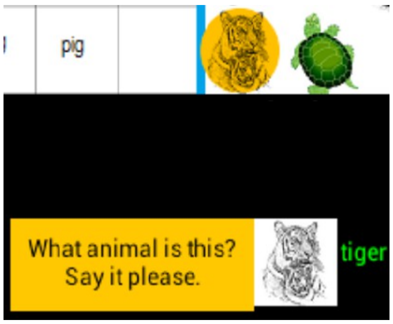
Animal Canvas - a children’s educational *App Inventor 2* game tutorial

Learn to build a game children can talk and listen to and learn how to recognize words. *Animal Canvas* is a simple AI2 app using lots of code and ImageSprites. The example app uses the Speech Recognizer, the TTS (TextToSpeech) and a grid on the Canvas to let children explore animal images, sounds, and the names of animals. The app is created on a single *App Inventor* screen.

 1:20 PM armadillo elephant sheep bear fox snake cat goat squirrel chicken horse tigers cow lion turtle dog pig Animal Options	 pig What animal is this? Say it please. tiger	
Animal Canvas main screen	*Animal Canvas* Question	

The animal images, sounds and words are managed with several lists keyed to each other with their item index numbers. The app is converted to another display language by substituting your language files into the lists (you need to provide the code). The app could become a bilingual app if you as a developer use techniques as described in this tutorial: **Polyglot … a multilingual tutorial for App Inventor 2** . *Animal Canvas* avoids hard coding fixed data, relying on variables and Lists instead of hardcoded values to identify images and text. The developer can modify the contents of several lists and change the playing language. Changing animals or adding more animals is more complicated. You need to understand how the app is programmed. Not happy with the images provided? Create new images of animals and image lists and replace the existing images and lists.

The *Animal Canvas* app includes “developer options.” These options are tools to help the developer understand how the app works. Eliminate the tools and associated code when you adjust the app to your purposes. Be careful. Create interim aia files while modifying the code in case you accidentally delete required blocks (so you are able to return gracefully to a point where the app last worked correctly prior to your modifications).*Animal Canvas* speaks English as designed. Change the TTS blocks in the Screen1.Initialize block to another language easily. You also must provide a List of animals in the appropriate language to substitute for their English names.

Game Design

The game is designed around a 36 block grid placed on a Canvas. A sprite’s position on the grid is monitored by the app. The game requires 17 animal menagerie images. The game could easily be a game based on birds, flags, traffic signs, images of important people...your imagination is your guide.

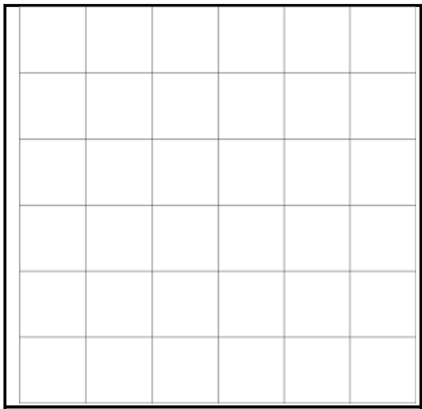
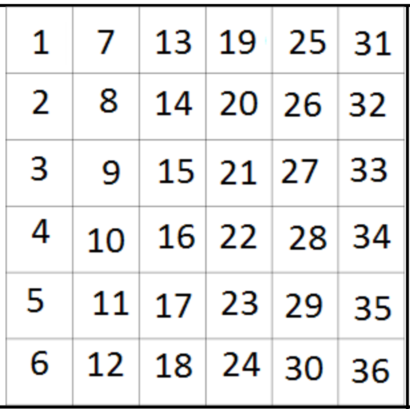
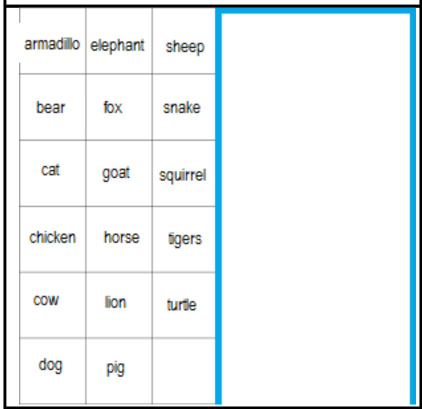
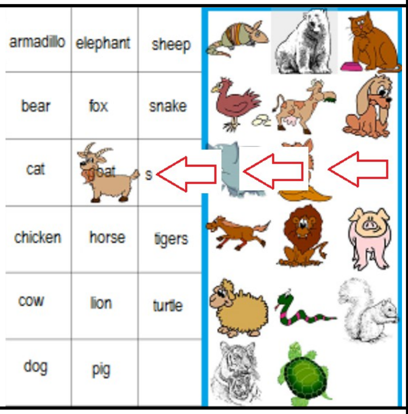
Lists manage the sounds, images and animal names.

Code an app to minimize reduce redundancy in code. I expect sometimes this goal is at the expense of clarity. While coding the example app, several original procedures were combined into one procedure. Reducing redundancy resultted in a net gain savings of a dozen blocks for over an hour of additional coding effort for no performance gain. The app uses two SpeechRecognizer controls; the functions could be coded with a single SR. I did not combine these into a single SR because the app performance was not enhanced and combining the functions resulted in difficult to comment/explain code. When you code, understand what a block of code does.

Animal Canvas is a children’s game/learning tool. There is no scoring. Let the kids have fun. Is the intended use of the app a competitive User audience? A countdown timer, a score system and a TinyDB to keep a single high score persistent might be appropriate in that instance. Keep scoring simple. A simple countdown or count up timer is shown **here**. Consider rewarding the minimal time to complete a task or the number of right answers versus incorrect answers as a goal. A children’s game is supposed to help a child learn and have fun.

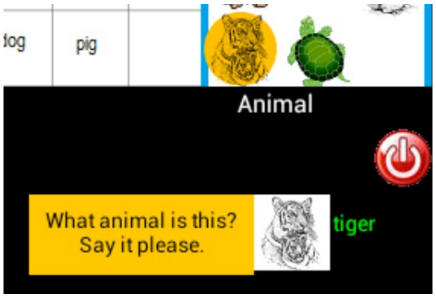
Playing Screen

The playing screen is constructed on a Canvas. The illustration below shows how how the CAnas is used for *Animal Canvas*. Your version of the app can have a few more animals (or other objects to play with).

	 1 7 13 19 25 31 2 8 14 20 26 32 3 9 15 21 27 33 4 10 16 22 28 34 5 11 17 23 29 35 6 12 18 24 30 36	The playing screen is made by creating a 300 x 300 pixel six by six grid using *Paint* or another image manipulation program on your PC. The grid is placed on a Canvas control and the grid snap routine knows where each numbered cell is located. You do not need to actually number the grid; you do need to be aware what cells are where.
Start with a grid image	*Label the grid with numbers*	
 armadillo elephant sheep bear fox snake cat goat squirrel chicken horse tigers cow lion turtle dog pig	 armadillo elephant sheep bear fox snake cat oat s chicken horse tigers cow lion turtle dog pig	This app uses 17 animals. Create a white space to the right for the animals to 'park.' You need 17 50x50 pixels placed in 17 ImageSprite.Picture objects. The app allows the player to drag the animal images to the 18 grid cells to the left. The player is rewarded when the animal and the text match. The play is described in "How to Play" below. Here the goat is dragged from its "parked" position.
Label the grid with animal names	*The playing grid*	

Lists and the ImageSprite Dragged, TouchUp and TouchDown event handlers control where the animals roam and what happens as they move to (are dragged to) the correct text cell. See the ImageSprite10 blocks in **The Blocks** section of the tutorial to see how the blocks are used in this example. The *xyToGridNumber* and *xyToButtonNumber* utilities keep track of the animal grid cells.

The Ask Animal and Ask Sound parts of the game



What animal is this? Say it please.

The image to the left shows the **Animal Ask**. Use the ListPicker to navigate to this option.   Animals randomly appear to the right of the orange button after touching the orange button. This continues until all the animals are identified as described below.

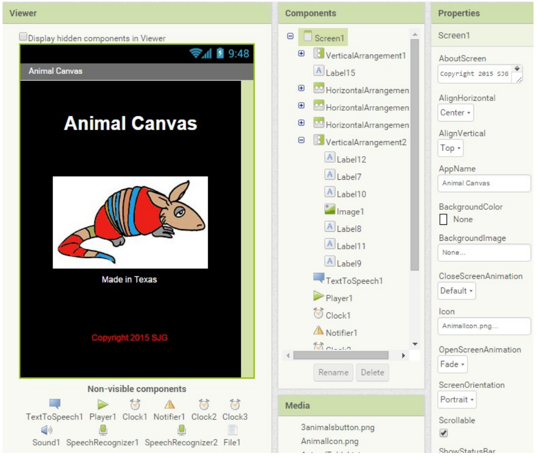
Click the orange button, an animal image appears along with the speech recognizer. The orange circle moves to the tiger on the playing screen for this example.   Say **tiger**; tiger is printed in green and the User is congratulated. Say something other than “tiger” or if the speech recognizer does not understand what is said and the spoken text appears in red text. Make a mistake choosing the animal or with pronunciation and a red text display appears. The User later gets an opportunity to identify the animal. Each time an animal is correctly identified, the animal is removed from the list of animals.   A wrong answer will not delete an animal. A User learns the images and sounds touching the animal images and dragging the images to the animal names.

The **Ask Sounds** is similar. The orange button in this case elicits an animal sound; correctly identify the sound by saying the animal name and get a reward. Well, the reward is only a visual and verbal indication the user provided the correct response. In your version of the app, the reward could be music or score points or…

The Designer Screen

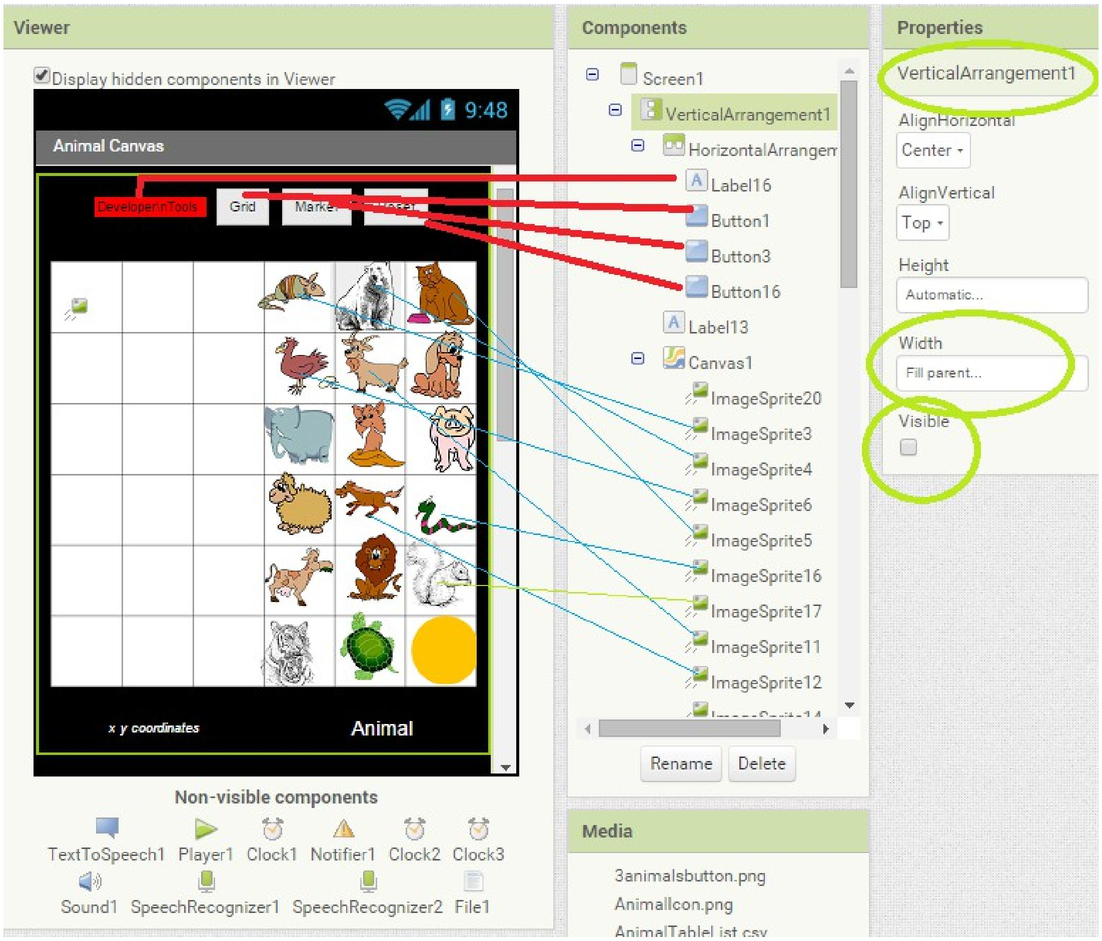
How the screen is laid out is very important in this app. Vertical layouts simulate screens. The main app virtual screen is VerticalArrangement1; the splash screen is VerticalArrangement2. Other layout controls display when the User uses the menu to ask questions about animals or animal sounds using the ListPicker menu.

The Splash Screen



VerticalArrangement2 is set to visible using the Designer.

The Main Screen

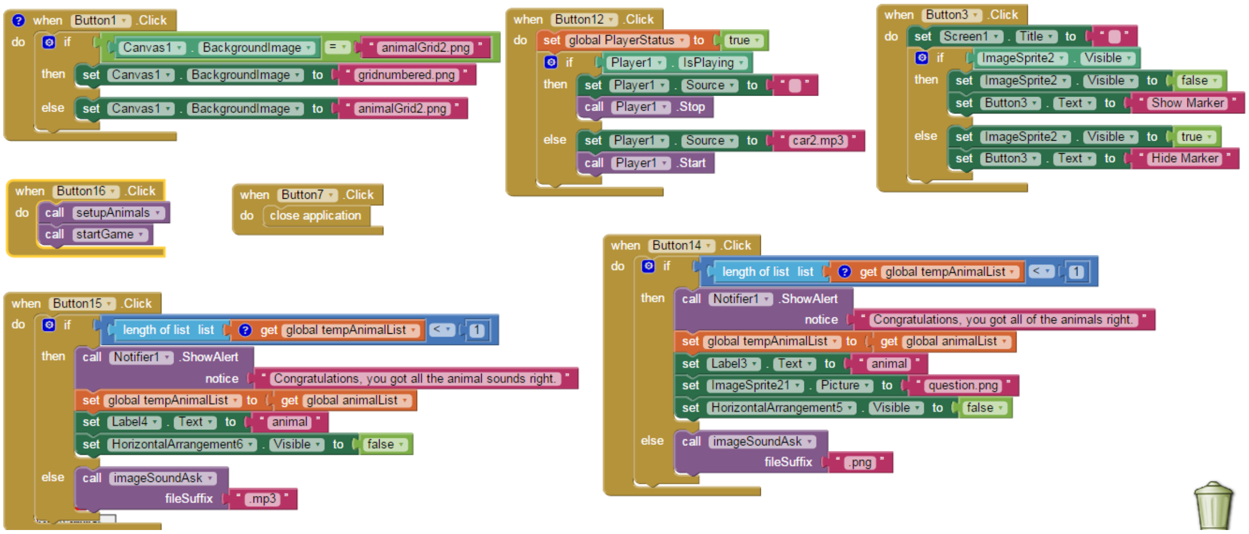


The Visible property of VerticalArrangement1 is set to **not visible** on the Designer (the Visible property is not checked).

The Blocks

This is an advanced tutorial. What the blocks do is explained for many but not all aspects of the game.

The Buttons



```
when Button1 .Click
do  if    Canvas1 . BackgroundImage = "animalGrid2.png"
    then  set Canvas1 . BackgroundImage to "gridnumbered.png"
    else  set Canvas1 . BackgroundImage to "animalGrid2.png"
```

```
when Button12 .Click
do  set global PlayerStatus to true
    if    Player1 . IsPlaying
    then  set Player1 . Source to " "
          call Player1 .Stop
    else  set Player1 . Source to "car2.mp3"
          call Player1 .Start
```

```
when Button3 .Click
do  set Screen1 . Title to " "
    if    ImageSprite2 . Visible
    then  set ImageSprite2 . Visible to false
          set Button3 . Text to "Show Marker"
    else  set ImageSprite2 . Visible to true
          set Button3 . Text to "Hide Marker"
```

```
when Button16 .Click
do  call setupAnimals
    call startGame
```

```
when Button7 .Click
do  close application
```

```
when Button15 .Click
do  if    length of list list  get global tempAnimalList < 1
    then  call Notifier1 .ShowAlert
                     notice "Congratulations, you got all the animal sounds right."
          set global tempAnimalList to  get global animalList
          set Label4 . Text to "animal"
          set HorizontalArrangement6 . Visible to false
    else  call imageSoundAsk
                  fileSuffix ".mp3"
```

```
when Button14 .Click
do  if    length of list list  get global tempAnimalList < 1
    then  call Notifier1 .ShowAlert
                     notice "Congratulations, you got all of the animals right."
          set global tempAnimalList to  get global animalList
          set Label3 . Text to "animal"
          set ImageSprite21 . Picture to "question.png"
          set HorizontalArrangement5 . Visible to false
    else  call imageSoundAsk
                  fileSuffix ".png"
```

Button1 contains code useful while developing to provide information to help understand how different parts of the code relate to the Canvas x,y coordinates and the grid cells. The code toggles between displaying a numbered grid and the animal grid on the playing screen.

Button12 has the code to stop and start the background music that is pre-loaded into the Android memory on program start.

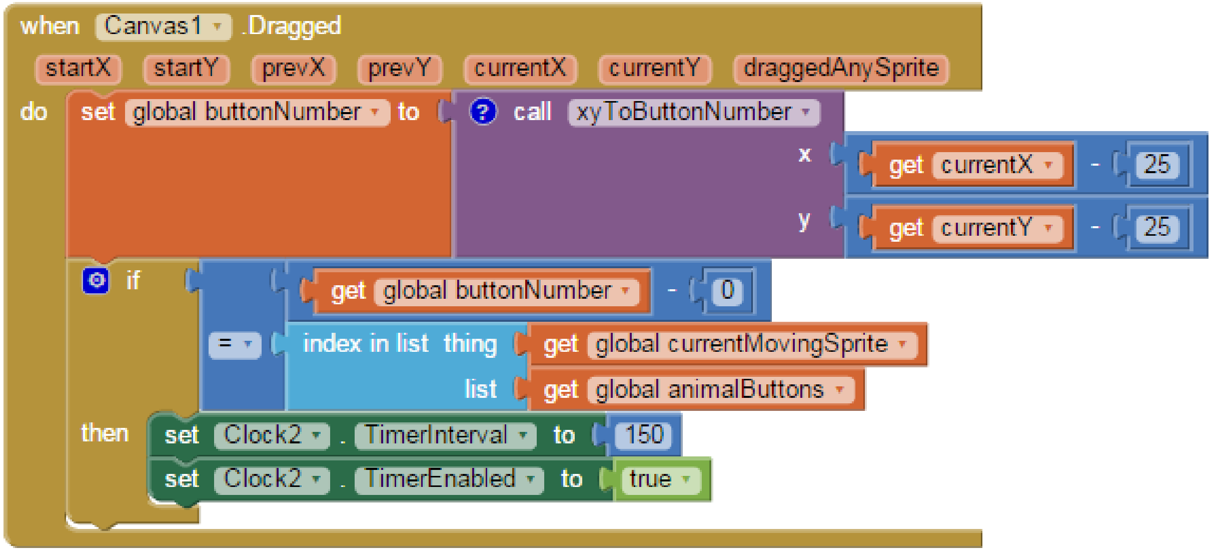
Button3 toggles the orange highlight marker visible to hidden.

Button16 calls the procedures *setupAnimals* and *startGame*. What the procedures do is explained later.

Button7 code gracefully closes the app.

Buttons 14 and 15 control the Ask image and animal sound questions. Press the buttons to ask to identify an animal or a sound.

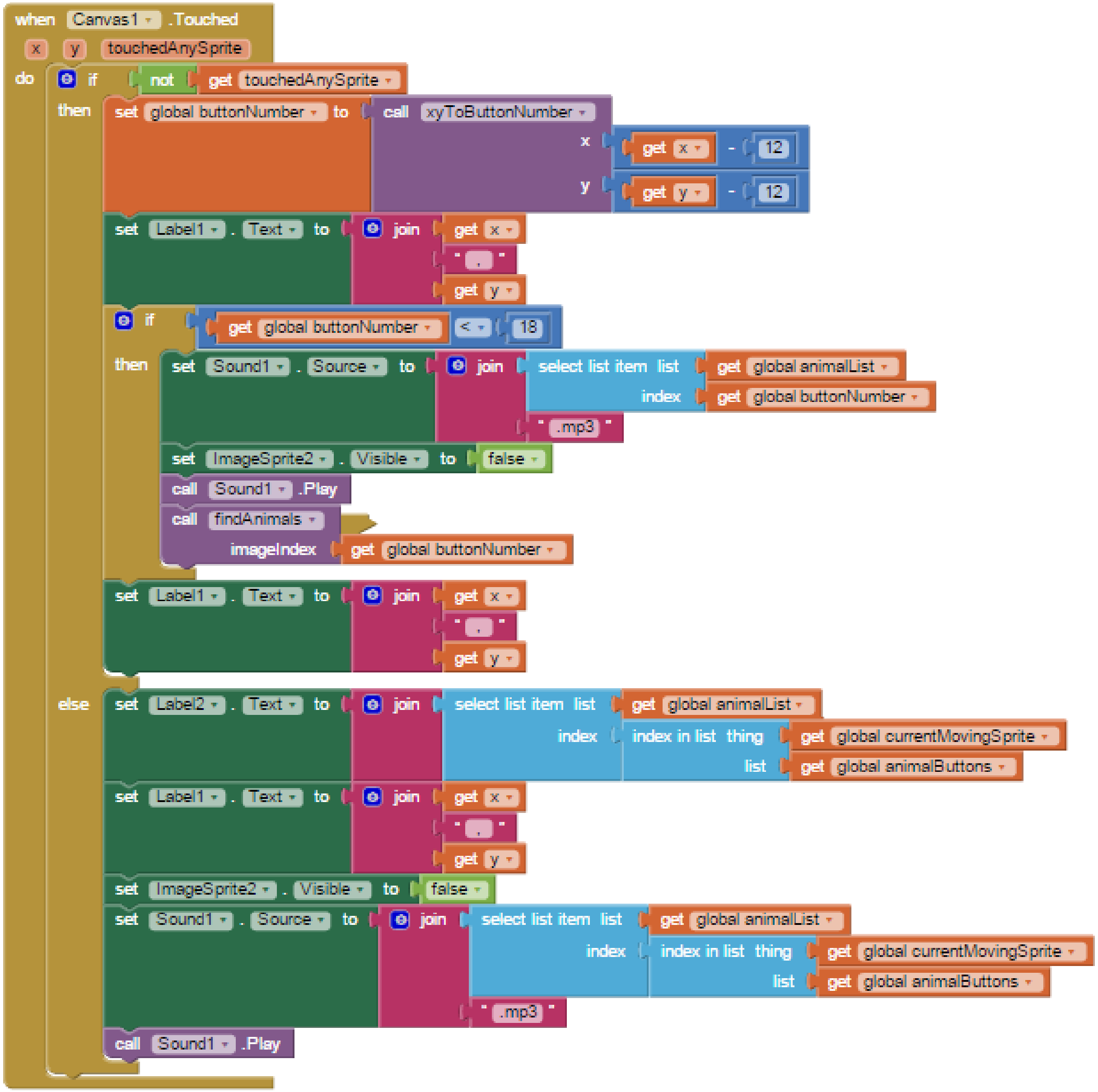
Canvas



```
when Canvas1 .Dragged
  startX  startY  prevX  prevY  currentX  currentY  draggedAnySprite
do  set global buttonNumber to  call xyToButtonNumber
                                     x  get currentX - 25
                                     y  get currentY - 25
    if      get global buttonNumber - 0
        =
            index in list  thing  get global currentMovingSprite
                           list   get global animalButtons
    then  set Clock2 . TimerInterval to 150
          set Clock2 . TimerEnabled to true
```

The Canvas Dragged event allows a user to drag an animal from the right side of the screen to the grid on the left. Part of the “game” is for a User to compare the animal name with the image by placing the image over the name of the animal. The code checks a User’s attempts to match the grid with the animal sprite; if the user moves the sprite to the correct block, the routine in Clock2 is invoked to provide the User a “reward” (“A Match”) and an indication of successful completion of animal identification.

The Canvas Touched event controls responses to touches. The code is aware of where the Canvas is touched. A touch might evoke an animal sound if an appropriate grid cell is touched.



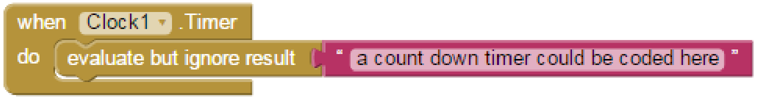
```
when Canvas1 .Touched
  x  y  touchedAnySprite
do  if  not  get touchedAnySprite
    then  set global buttonNumber to  call xyToButtonNumber
                                           x  get x - 12
                                           y  get y - 12
          set Label1 . Text to  join  get x
                                      " , "
                                      get y
          if  get global buttonNumber < 18
          then  set Sound1 . Source to  join  select list item  list   get global animalList
                                                                index  get global buttonNumber
                                              " .mp3 "
                set ImageSprite2 . Visible to false
                call Sound1 .Play
                call findAnimals
                     imageIndex  get global buttonNumber
          set Label1 . Text to  join  get x
                                      " , "
                                      get y
    else  set Label2 . Text to  join  select list item  list   get global animalList
                                                        index  index in list  thing  get global currentMovingSprite
                                                                              list   get global animalButtons
          set Label1 . Text to  join  get x
                                      " , "
                                      get y
          set ImageSprite2 . Visible to false
          set Sound1 . Source to  join  select list item  list   get global animalList
                                                          index  index in list  thing  get global currentMovingSprite
                                                                                list   get global animalButtons
                                        " .mp3 "
          call Sound1 .Play
```

The Clocks

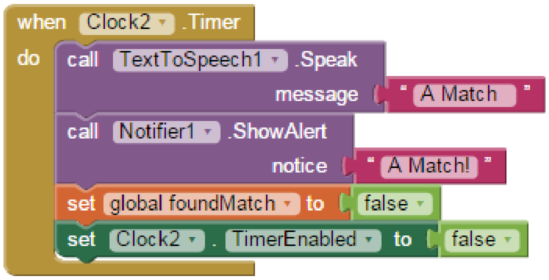
Clock1 is provided for future use. You supply the code.

Clock2 is called to provide a reward when it is determined an animal word and image match when a sprite is correctly position over a grid cell.

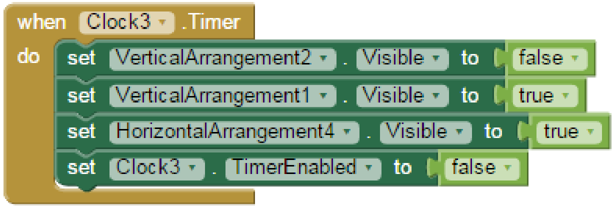
Clock3 invokes and closes the splash intro virtual screen (VerticalArrangement2). The Clock3.Interval is set in the Designer to 8000 ms to allow the introduction music to play and then show the main screen (VerticalArrangement1). The virtual screen displays for eight seconds.



```
when Clock1 .Timer
do  evaluate but ignore result  " a count down timer could be coded here "
```

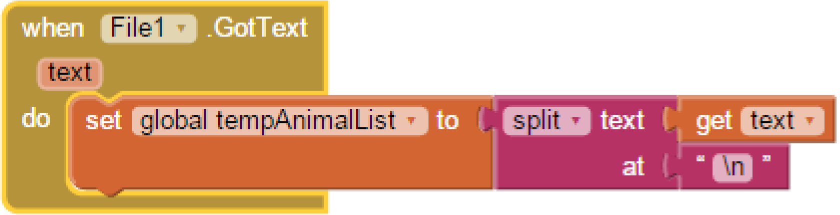


```
when Clock2 .Timer
do  call TextToSpeech1 .Speak
            message  " A Match "
    call Notifier1 .ShowAlert
            notice  " A Match! "
    set global foundMatch to  false
    set Clock2 . TimerEnabled to  false
```



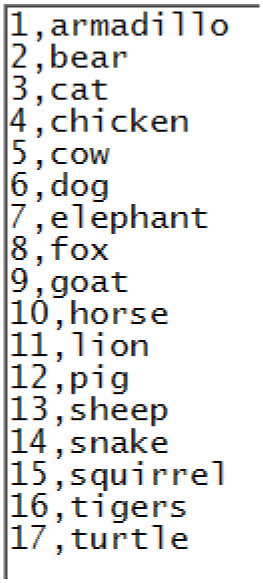
```
when Clock3 .Timer
do  set VerticalArrangement2 . Visible to  false
    set VerticalArrangement1 . Visible to  true
    set HorizontalArrangement4 . Visible to  true
    set Clock3 . TimerEnabled to  false
```

File1



```
when File1 .GotText
  text
do  set global tempAnimalList to  split  text  get text
                                         at  " \n "
```

The list of animals the app knows (is programmed with) is stored in a csv text file. AnimalTableList.csv is the file that names the 17 animals. Maintaining the list of animals in a csv allows the developer to “change” animals in the app or provide the names in a different language rather easily. The GotText tells the app what to do after the csv file is read in the Screen1.Initialize event handler. The split block loads the AnimalTableList.csv stored in Media (resources).



```
1,armadillo
2,bear
3,cat
4,chicken
5,cow
6,dog
7,elephant
8,fox
9,goat
10,horse
11,lion
12,pig
13,sheep
14,snake
15,squirrel
16,tigers
17,turtle
```

To change the language of play, provide a similar csv file substituting the animal names in an appropriate language; retain the numbers. You also have to set the TTS language appropriately as shown below in **Resources - Sound**.   Selectable multiple animal lists may be used to add more creatures to the game. Change the animals in the list using the language you want the app to work in or provide a secondary list to build a bilingual app Code should to be provided to select which animal list to use in game play (in either the ListPicker or somewhere else) if you build the app bilingual. Save the language choice in a TinyDB so the app will remember the primary language of play and is able to load the names at app start.

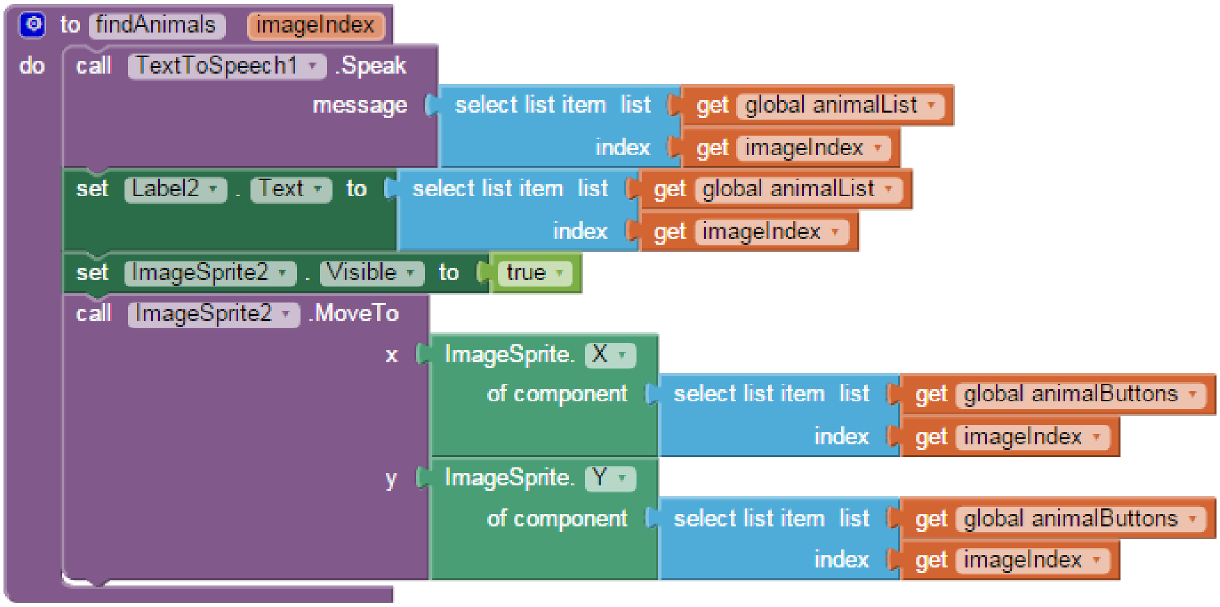
Later, the Screen1.Initial code is described. It says that ListPicker1.ElementsFromString   is composed of the following text:
`armadillo,bear,cat,chicken,cow,dog,elephant,fox,goat,horse,lion,pig,sheep,snake,squirrel,tigers,turtle`   and

global animalList is
`armadillo,bear,cat,chicken,cow,dog,elephant,fox,goat,horse,lion,pig,sheep,snake,squirrel,tigers,turtle.`

Both the lists could be populated using the AnimalTableList csv. Notice, with the exception of the csv being a List of Lists, providing an index number, the animal elements are identical. Coding to use the single csv instead of a combination of fixed data is a project for you. A single list will make future changes and additions to the animal displays easier, however, providing the separate lists might be easier for most new AI2 coders to understand.

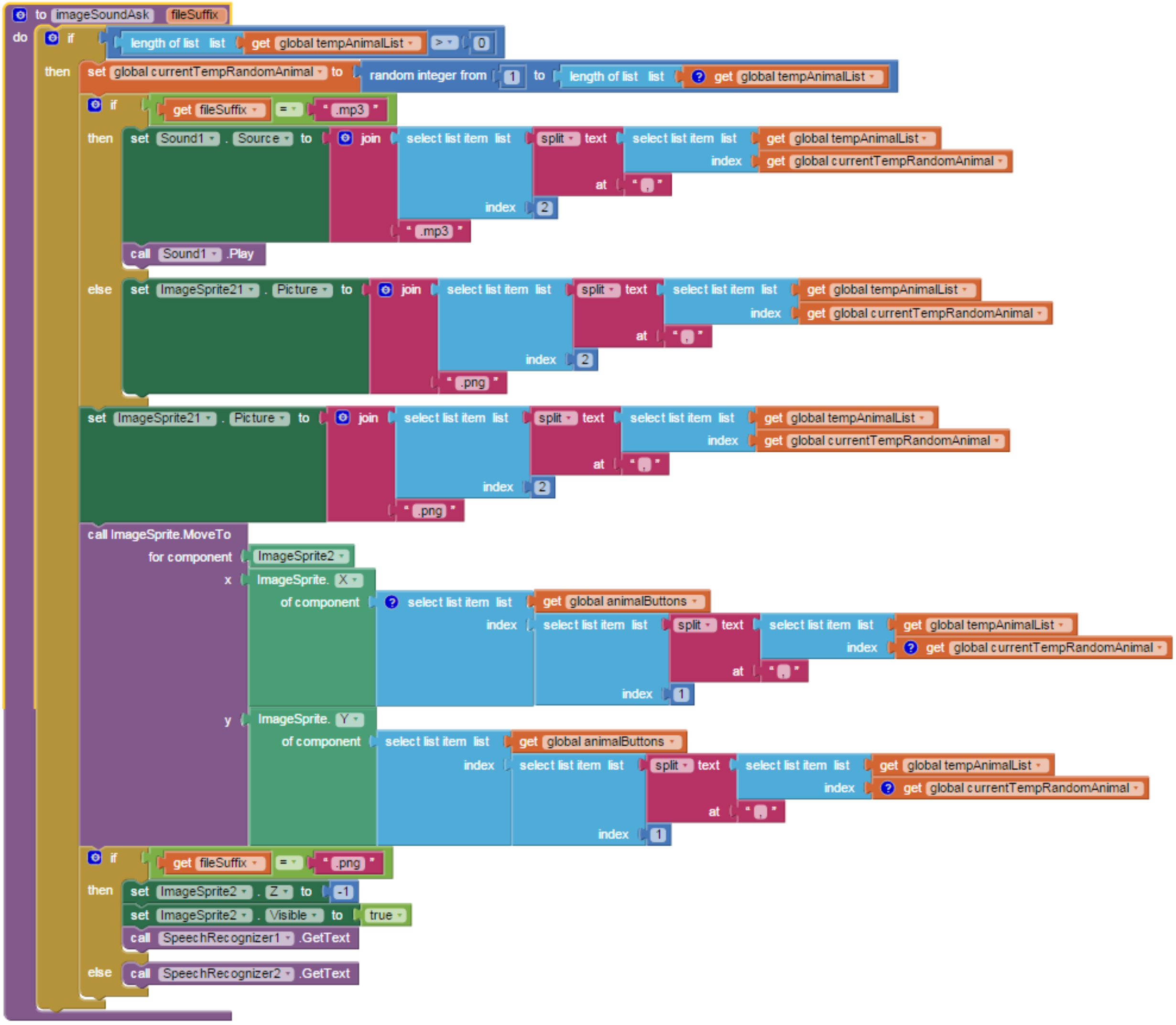
Procedures

The ***findAnimals*** Procedure uses the *Any component* blocks to “move” the orange marker sprite (imagesprite2) to the appropriate Canvas coordinates.



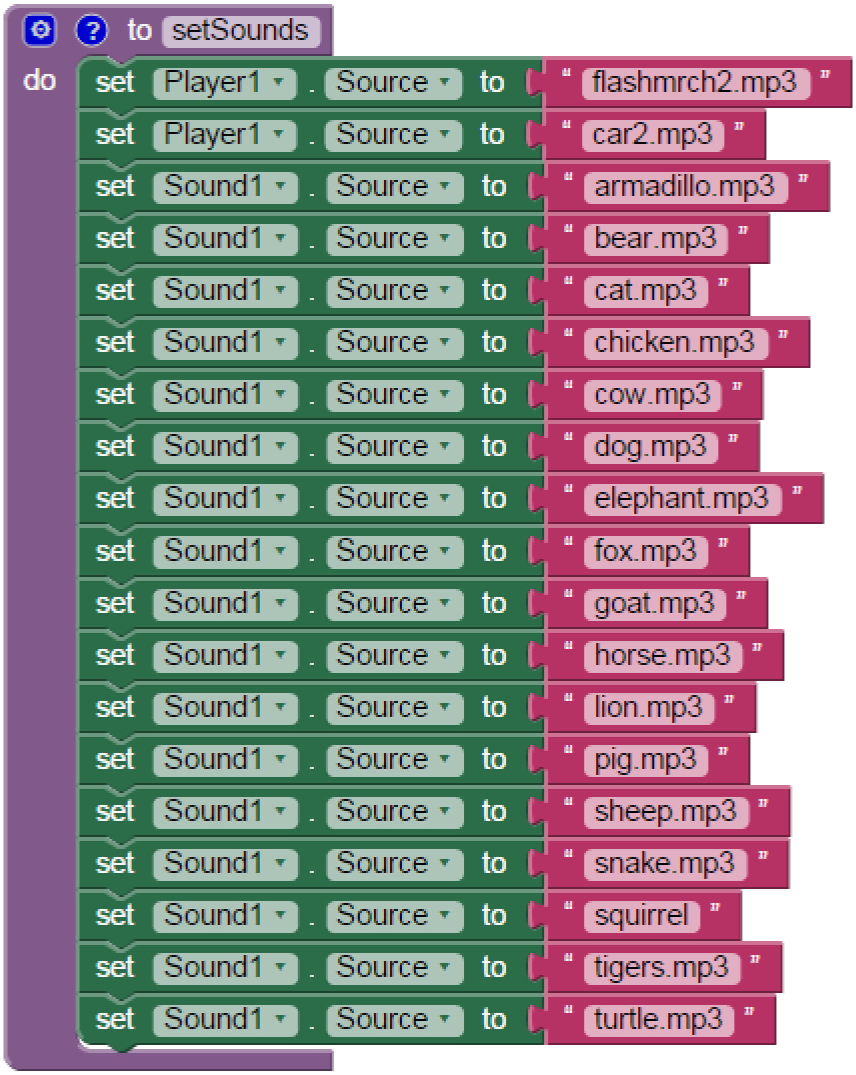
```
to findAnimals imageIndex
do  call TextToSpeech1 .Speak
                 message  select list item  list  get global animalList
                                           index  get imageIndex
    set Label2 . Text to  select list item  list  get global animalList
                                           index  get imageIndex
    set ImageSprite2 . Visible to  true
    call ImageSprite2 .MoveTo
                      x  ImageSprite. X
                            of component  select list item  list  get global animalButtons
                                                           index  get imageIndex
                      y  ImageSprite. Y
                            of component  select list item  list  get global animalButtons
                                                           index  get imageIndex
```

The ***imageSoundAsk*** Procedure selects an animal from the tempAnimalList randomly . The code knows whether the Ask animal sound or Ask animal picture buttons are used.



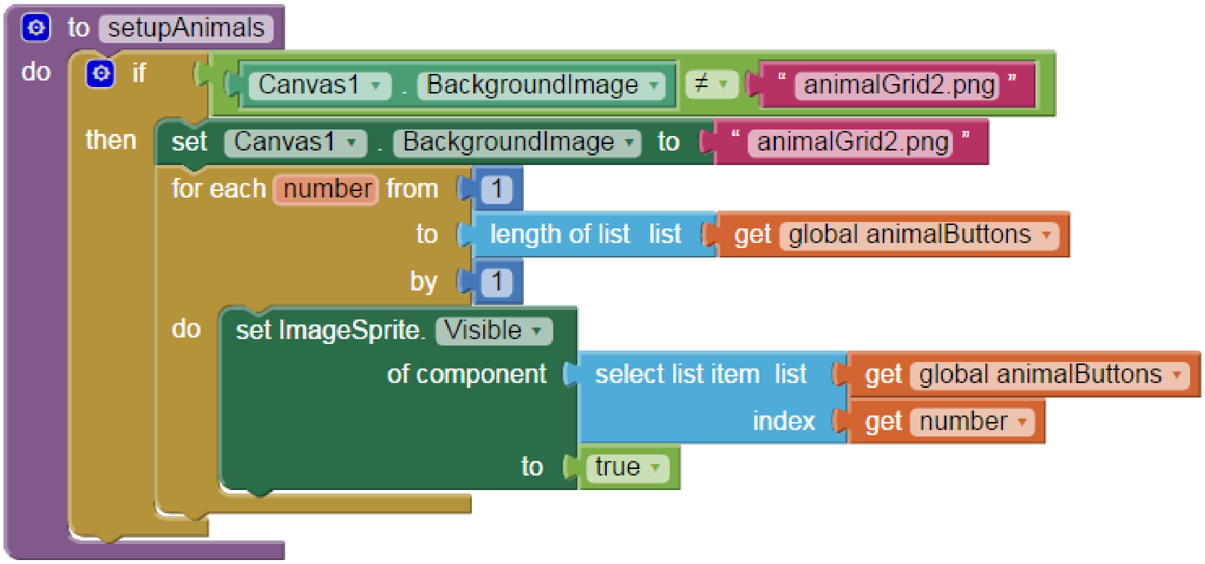
```
to imageSoundAsk fileSuffix
do  if  length of list list  get global tempAnimalList  >  0
    then  set global currentTempRandomAnimal to  random integer from 1 to  length of list list  get global tempAnimalList
          if  get fileSuffix  =  " .mp3 "
          then  set Sound1 . Source to  join  select list item list  split text  select list item list  get global tempAnimalList
                                                                                  index  get global currentTempRandomAnimal
                                                                    at  " , "
                                                   index  2
                                             " .mp3 "
                call Sound1 .Play
          else  set ImageSprite21 . Picture to  join  select list item list  split text  select list item list  get global tempAnimalList
                                                                                         index  get global currentTempRandomAnimal
                                                                           at  " , "
                                                          index  2
                                                    " .png "
          set ImageSprite21 . Picture to  join  select list item list  split text  select list item list  get global tempAnimalList
                                                                                   index  get global currentTempRandomAnimal
                                                                     at  " , "
                                                    index  2
                                              " .png "
          call ImageSprite.MoveTo
               for component  ImageSprite2
                           x  ImageSprite. X
                                 of component  select list item list  get global animalButtons
                                                                index  select list item list  split text  select list item list  get global tempAnimalList
                                                                                                                        index  get global currentTempRandomAnimal
                                                                                                          at  " , "
                                                                                         index  1
                           y  ImageSprite. Y
                                 of component  select list item list  get global animalButtons
                                                                index  select list item list  split text  select list item list  get global tempAnimalList
                                                                                                                        index  get global currentTempRandomAnimal
                                                                                                          at  " , "
                                                                                         index  1
          if  get fileSuffix  =  " .png "
          then  set ImageSprite2 . Z to  -1
                set ImageSprite2 . Visible to  true
                call SpeechRecognizer1 .GetText
          else  call SpeechRecognizer2 .GetText
```

The **setSounds** Procedure is a List of the mp3 animal sound files. The procedure is called in the Screen1.Initialize block to pre-load the sound files. Pre-loading means the sound files are ready to play when called for within the app.



```
to setSounds
do  set Player1 . Source to "flashmrch2.mp3"
    set Player1 . Source to "car2.mp3"
    set Sound1 . Source to "armadillo.mp3"
    set Sound1 . Source to "bear.mp3"
    set Sound1 . Source to "cat.mp3"
    set Sound1 . Source to "chicken.mp3"
    set Sound1 . Source to "cow.mp3"
    set Sound1 . Source to "dog.mp3"
    set Sound1 . Source to "elephant.mp3"
    set Sound1 . Source to "fox.mp3"
    set Sound1 . Source to "goat.mp3"
    set Sound1 . Source to "horse.mp3"
    set Sound1 . Source to "lion.mp3"
    set Sound1 . Source to "pig.mp3"
    set Sound1 . Source to "sheep.mp3"
    set Sound1 . Source to "snake.mp3"
    set Sound1 . Source to "squirrel"
    set Sound1 . Source to "tigers.mp3"
    set Sound1 . Source to "turtle.mp3"
```

The ***setupAnimals*** Procedure places the animals on the play screen in a neat arrangement. You might provide code to shuffle the sprite locations to always have a different arrangement of the start positions for the images. You have to write additional code.



```
to setupAnimals
do  if    Canvas1 . BackgroundImage ≠ " animalGrid2.png "
    then  set Canvas1 . BackgroundImage to " animalGrid2.png "
          for each number from 1
                            to length of list list get global animalButtons
                            by 1
          do  set ImageSprite. Visible
                  of component select list item list get global animalButtons
                                                index get number
                            to true
```

Speech Recognizers

Two speech recognizers are used. Separate code blocks help to avoid confusion.

SpeechRecognizer1.AfterGettingText handles responses to the animal pictures questions.

The text responses understood from the speech recognizer are downcased. Responses to the SpeechRecognizer sometimes result in capitalization of the first letter spoken, other times capitalization does not occur. Down casing all responses ensures that the app responds to both “Elephant” and “elephant” for example and renders them both as “elephant” which is the required correct response recognized by the SR.

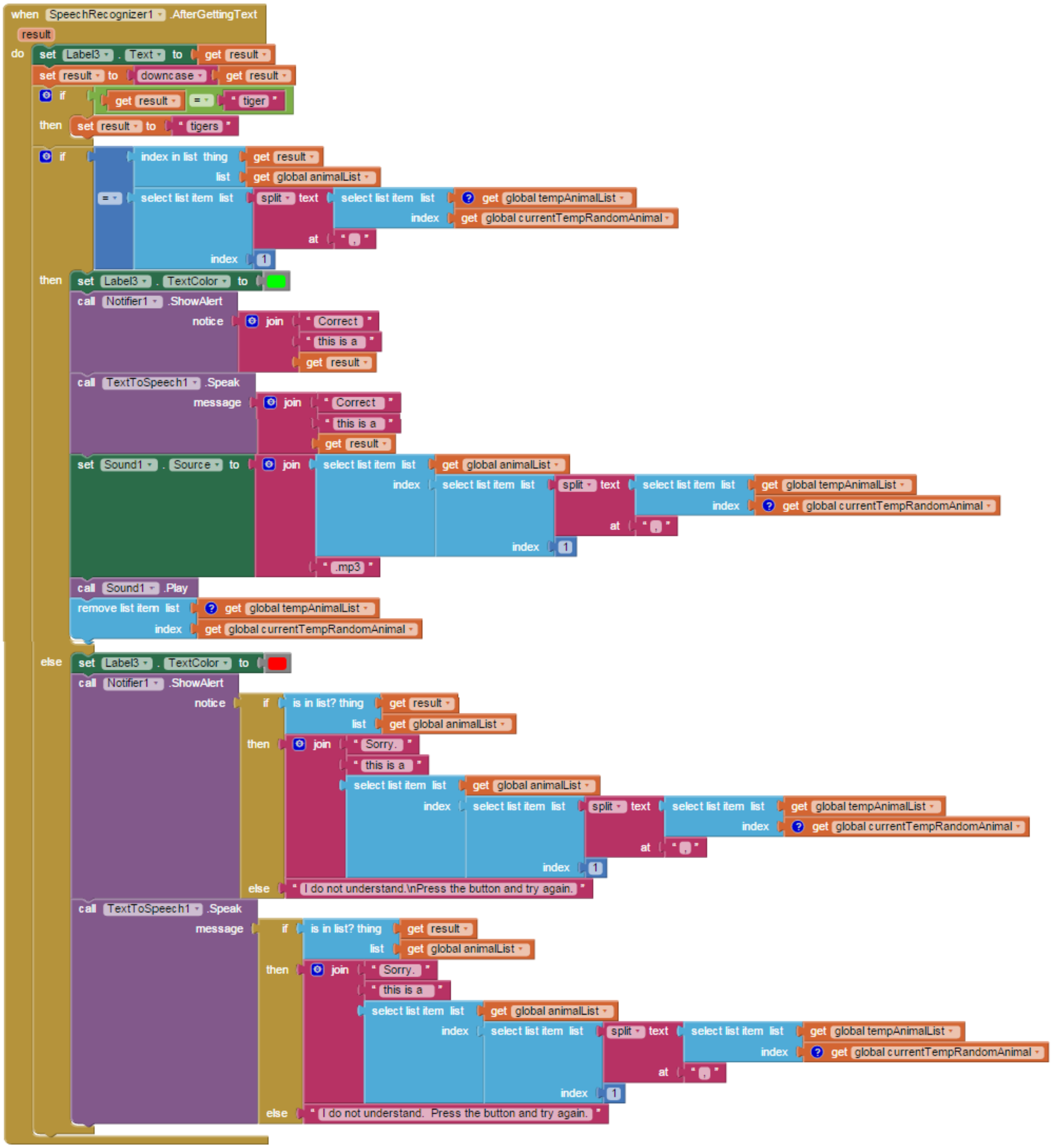
Animal Canvas uses an image depicting both a large and small tiger.   Is the proper response “tiger” or “tigers?” Code is provided to advise the SR to respond in the same way to both “tiger” and to the plural “tiger” with an If..then block. The plural versus singular problem was not envisioned when selecting the animal images. The image could be changed or code provided to allow flexibility in SR understanding could be added.

Ambiguity might be minimized in your app by avoiding images that have more than one individual represented. Coding similar to what resolved this issue can provide flexibility in the SR “understanding” a User’s response to difficult to pronounce words if necessary. You may want to provide an alternative to common names for animals; for example, you as a programmer might allow the SR to recognize both “Billy goat” and “goat” as **goat.** Without additional code to filter responses, children’s answers (and adult’s too) might result in the SR not accepting a response as valid. Try to anticipate those situations; the SR, by itself, is not very smart.

If the spoken word is a word present in the animalList, the app understands it should respond with a “Correct” message. If correct, the user is rewarded with a notifier message, a spoken message and the sound of the animal celebrating.

If a User verbal response is correct, the animal is removed from the temporary animalList. The app does not ask about that animal again.

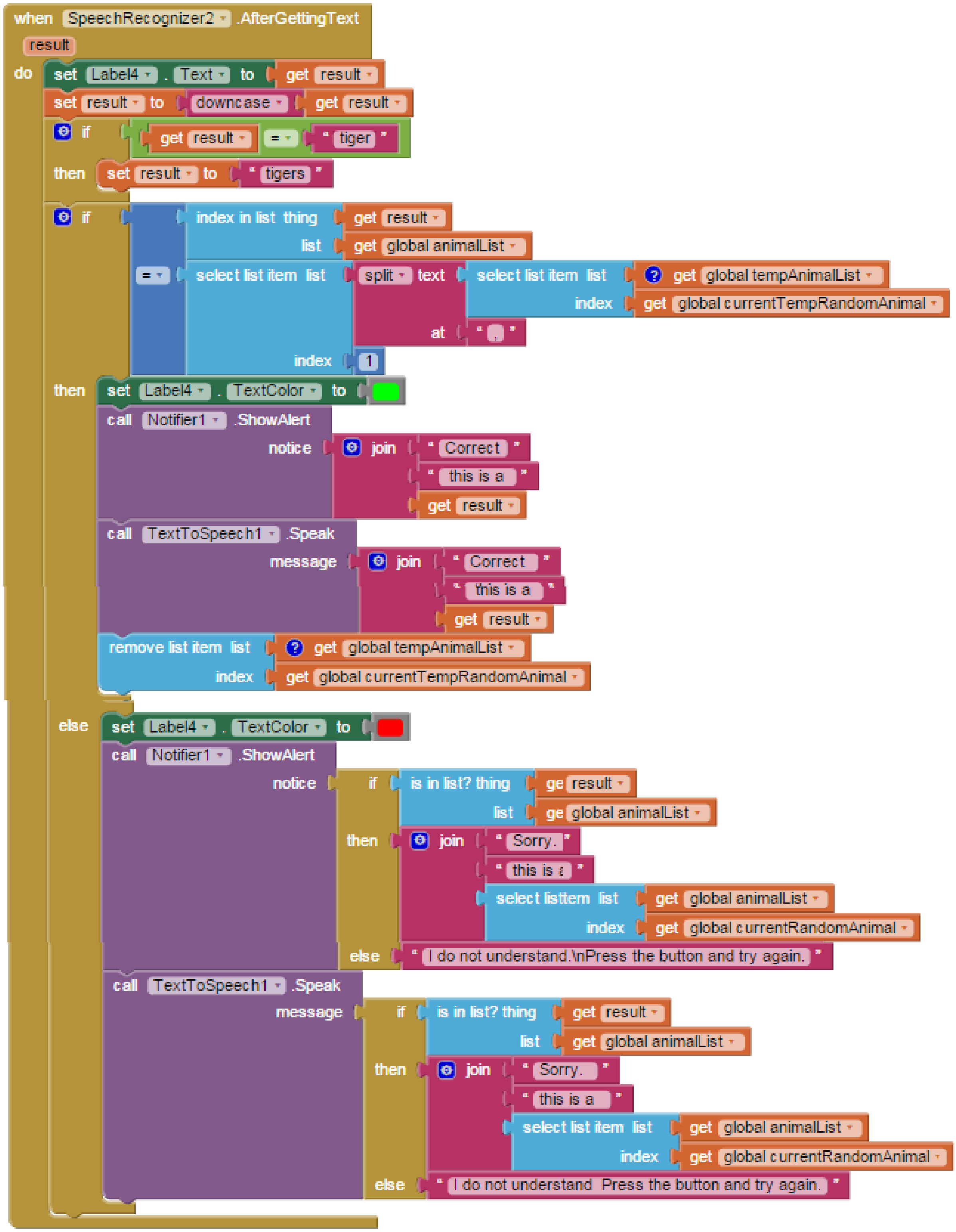
If the animal identification is not correct but the name of an animal spoken is also in the list, a “sorry” response is elicited. If there is no animal in the list that matches, the app indicates it does not understand what the user said.



```
when SpeechRecognizer1 .AfterGettingText
  result
do  set Label3 . Text to  get result
    set result to  downcase  get result
    if    get result = " tiger "
    then  set result to " tigers "
    if    =  index in list thing  get result
                           list   get global animalList
             select list item list  split text  select list item list   get global tempAnimalList
                                                                  index  get global currentTempRandomAnimal
                                          at  " , "
                              index  1
    then  set Label3 . TextColor to
          call Notifier1 .ShowAlert
                          notice  join  " Correct "
                                        " this is a "
                                        get result
          call TextToSpeech1 .Speak
                          message  join  " Correct "
                                         " this is a "
                                         get result
          set Sound1 . Source to  join  select list item list  get global animalList
                                                         index  select list item list  split text  select list item list   get global tempAnimalList
                                                                                                                     index  get global currentTempRandomAnimal
                                                                                             at  " , "
                                                                                 index  1
                                        " .mp3 "
          call Sound1 .Play
          remove list item list   get global tempAnimalList
                           index  get global currentTempRandomAnimal
    else  set Label3 . TextColor to
          call Notifier1 .ShowAlert
                          notice  if    is in list? thing  get result
                                                   list    get global animalList
                                  then  join  " Sorry. "
                                              " this is a "
                                              select list item list  get global animalList
                                                               index  select list item list  split text  select list item list   get global tempAnimalList
                                                                                                                           index  get global currentTempRandomAnimal
                                                                                                   at  " , "
                                                                                       index  1
                                  else  " I do not understand.\nPress the button and try again. "
          call TextToSpeech1 .Speak
                          message  if    is in list? thing  get result
                                                    list    get global animalList
                                   then  join  " Sorry. "
                                               " this is a "
                                               select list item list  get global animalList
                                                                index  select list item list  split text  select list item list   get global tempAnimalList
                                                                                                                            index  get global currentTempRandomAnimal
                                                                                                    at  " , "
                                                                                        index  1
                                   else  " I do not understand.  Press the button and try again. "
```

SpeechRecognizer2.AfterGettingText handles responses to animal sounds.

The code block follows the same pattern as SpeechRecognizer1 with some subtle differences.



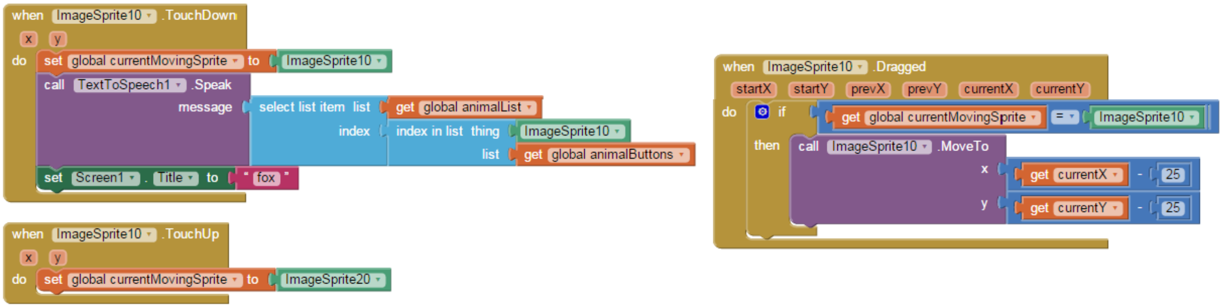
```
when SpeechRecognizer2 .AfterGettingText
  result
do  set Label4 . Text to  get result
    set result to  downcase  get result
    if  get result = " tiger "
    then  set result to " tigers "
    if  =   index in list  thing  get result
                           list   get global animalList
            select list item  list   split  text  select list item  list   get global tempAnimalList
                                                                    index  get global currentTempRandomAnimal
                                            at    " , "
                              index  1
    then  set Label4 . TextColor to 🟩
          call Notifier1 .ShowAlert
                         notice   join  " Correct "
                                        " this is a "
                                        get result
          call TextToSpeech1 .Speak
                         message  join  " Correct "
                                        " this is a "
                                        get result
          remove list item  list   get global tempAnimalList
                            index  get global currentTempRandomAnimal
    else  set Label4 . TextColor to 🟥
          call Notifier1 .ShowAlert
                         notice   if    is in list? thing  ge result
                                                    list   ge global animalList
                                  then  join  " Sorry. "
                                              " this is a "
                                              select listtem  list   get global animalList
                                                              index  get global currentRandomAnimal
                                  else  " I do not understand.\nPress the button and try again. "
          call TextToSpeech1 .Speak
                         message  if    is in list? thing  get result
                                                    list   get global animalList
                                  then  join  " Sorry. "
                                              " this is a "
                                              select list item  list   get global animalList
                                                                index  get global currentRandomAnimal
                                  else  " I do not understand  Press the button and try again. "
```

Image Sprites ...all essentially the same but be careful

There are three event handlers used in conjunction with ImageSprite 3 to 19 ... all 17 of the animal ImageSprites event handlers are coded almost identically. Note, there is no number 1 sprite.   ImageSprite2 is the orange circle marker, it does not have any code in an event handler. ImageSprite20 is placed on the Canvas but there is no image assigned to it intentionally.

Below is the code for the three ImageSprite10 event handlers used in the app. The code controls the fox. The TouchDown event handler lets the app know which sprite is moving and provides the code that has the TTS say the animal name of the image touched. There is code to post the word fox to the title screen; (however the title screen is disabled. The code is used as a "comment" to identify the animal associated with the sprite.

The Dragged event handler tells the app to allow the sprite to move and to drop the sprite at the current x and y location on the Canvas. TouchUp is used to tell the app that <u>none</u> of the animal sprites is currently moving. ImageSprite20 is a sprite with no image and is a placeholder and is assigned as the "currentMovingSprite."



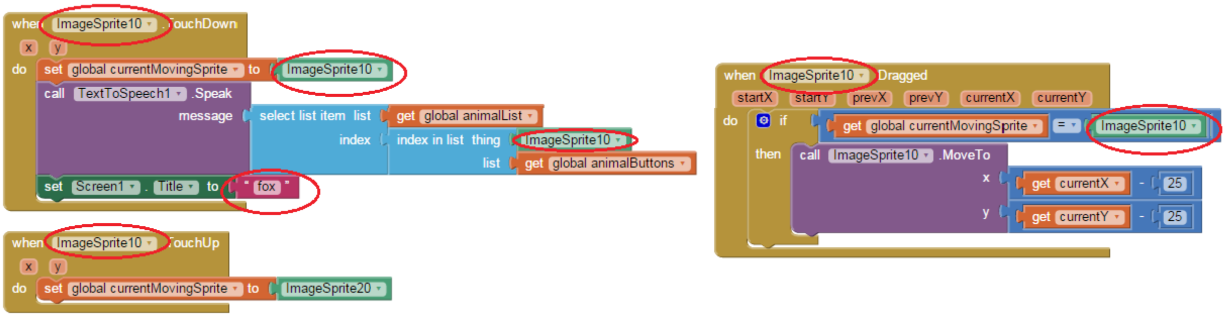
```
when ImageSprite10 .TouchDown
  x  y
do  set global currentMovingSprite to ImageSprite10
    call TextToSpeech1 .Speak
         message  select list item  list  get global animalList
                                   index  index in list  thing  ImageSprite10
                                                         list   get global animalButtons
    set Screen1 . Title to " fox "

when ImageSprite10 .TouchUp
  x  y
do  set global currentMovingSprite to ImageSprite20

when ImageSprite10 .Dragged
  startX  startY  prevX  prevY  currentX  currentY
do  if   get global currentMovingSprite  =  ImageSprite10
    then call ImageSprite10 .MoveTo
              x  get currentX  -  25
              y  get currentY  -  25
```

The above described ImageSprite10; you are not finished. Sixteen more sets of three event handlers must also be created. A set is necessary for each of the remaining animal spridtes.

The code must be modified in seven places (circled in red) shown below when you make event handlers for all the required sprites. Smile, the template aia does most of this for you.

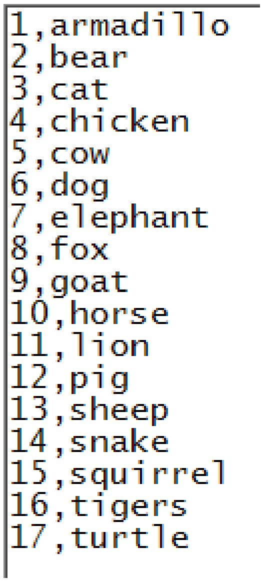


```
when ImageSprite10 .TouchDown
  x  y
do  set global currentMovingSprite to ImageSprite10
    call TextToSpeech1 .Speak
         message  select list item  list  get global animalList
                                   index  index in list  thing  ImageSprite10
                                                         list   get global animalButtons
    set Screen1 . Title to " fox "

when ImageSprite10 .TouchUp
  x  y
do  set global currentMovingSprite to ImageSprite20

when ImageSprite10 .Dragged
  startX  startY  prevX  prevY  currentX  currentY
do  if   get global currentMovingSprite  =  ImageSprite10
    then call ImageSprite10 .MoveTo
              x  get currentX  -  25
              y  get currentY  -  25
```

A block is required for each of the other sixteen animals wherever ImageSprite10 is indicated in the images,. Where the TouchDown block indicates "fox," change the animal name

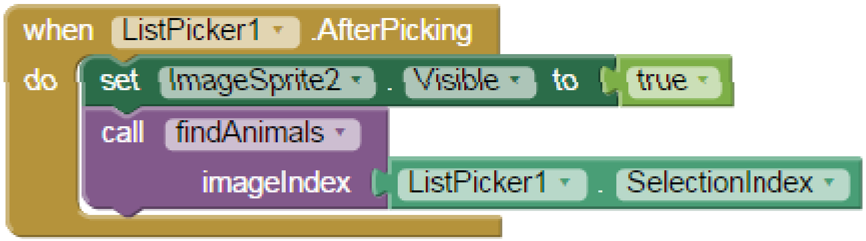


```
1,armadillo
2,bear
3,cat
4,chicken
5,cow
6,dog
7,elephant
8,fox
9,goat
10,horse
11,lion
12,pig
13,sheep
14,snake
15,squirrel
16,tigers
17,turtle
```

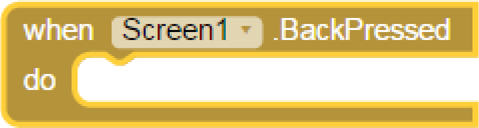
appropriately using the above list as a guide. Add 2 to the number shown on the list to get the appropriate sprite ID; for example, tigers is 16+2 or ImageSprite18. You could rename ImageSprite3 to ImageSprite1 etc. or provide more descriptive names (TigersImageSprite perhaps) to tidy up the coding. If you rename, do not get confused or you will mess up a lot of code. I chose not to rename, being comfortable with the default object labels..

Stuff

AfterPicking uses the findAnimals procedure in conjunction with the ListPicker SelectionIndex to find an animal using the menu of animals.



```
when ListPicker1 .AfterPicking
do  set ImageSprite2 . Visible to true
    call findAnimals
         imageIndex ListPicker1 . SelectionIndex
```

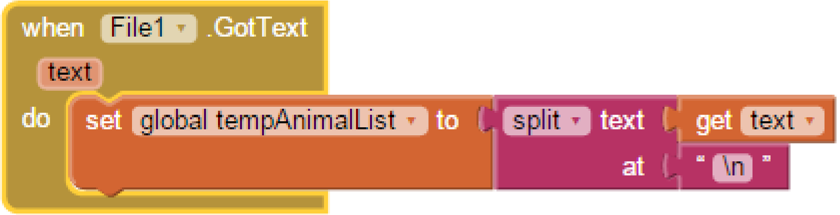


```
when Screen1 .BackPressed
do
```

An empty BackPressed prevents a User from inadvertently closing the app when they press the Android refresh virtual key.

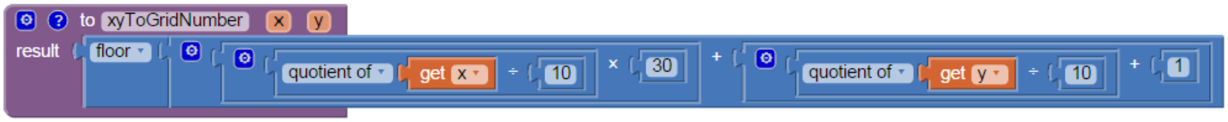
File1

File1.GetText retrieves the csv file AnimalTableList.csv . This csv is loaded at the start of the app and makes a List called tempAnimalList to use to keep track of the animal questions asked using the split block.

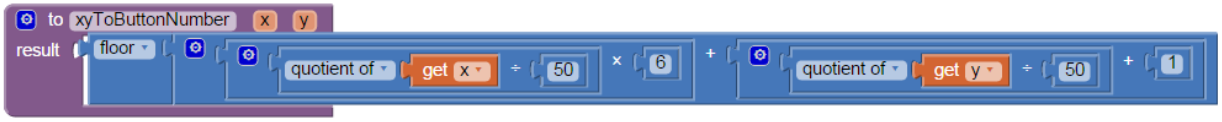


```
when File1 .GotText
  text
do set global tempAnimalList to split text get text
                                      at " \n "
```

The xyToGridNumber and xyToButtonNumber procedures are utilities to keep track of the animal grid cells. These are the grid snap routines.



```
to xyToGridNumber x y
result floor quotient of get x ÷ 10 × 30 + quotient of get y ÷ 10 + 1
```



```
to xyToButtonNumber x y
result floor quotient of get x ÷ 50 × 6 + quotient of get y ÷ 50 + 1
```

Screen1.Initialize

TextToSpeech settings are adjusted here. If you want AnimalCanvas to use non-English animal names, you must supply a modified animalList and change the Language and perhaps the country entry. See the section below on Resources. SpeechRate is slowed slightly from the default 1.

File1.ReadFrom loads the AnimalTableList csv file (see the File1.GotText block above) into the tempAnimalList. The tempAnimalList is the List from which random images or sounds are provided for the orange Ask buttons.

ListPicker1.ElementsFromString   is `armadillo,bear,cat,chicken,cow,dog,elephant,fox,goat,horse,lion,pig,sheep,snake,squirrel,tigers,turtle`

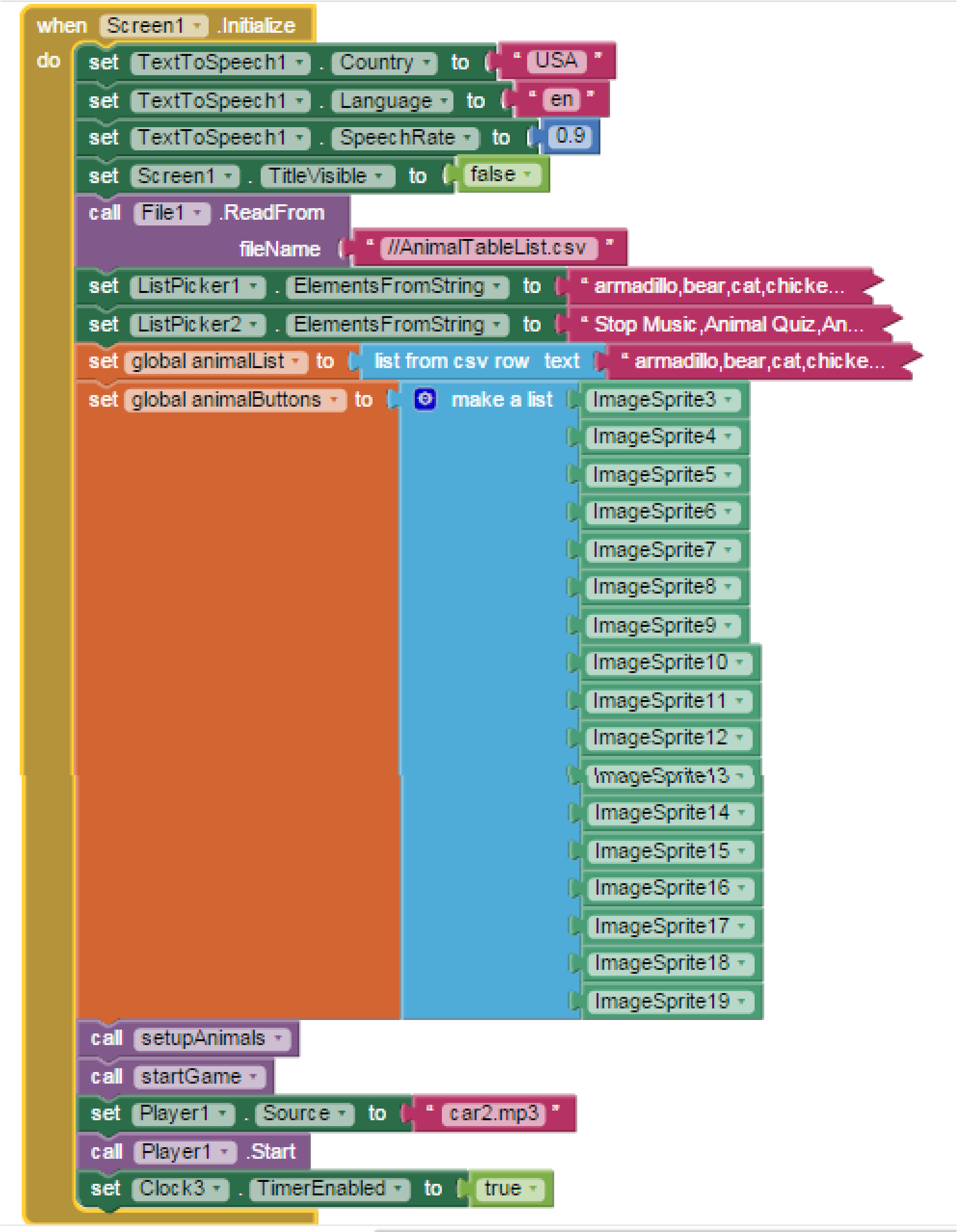
LIstPicker2.ElementsFromString is `Stop Music,Animal Quiz,Animal Sound Quiz,Hide / Reset Quizes,Toggle Developer Tools`

global animalList is `armadillo,bear,cat,chicken,cow,dog,elephant,fox,goat,horse,lion,pig,sheep,snake,squirrel,tigers,turtle`

TextToSpeech is set to USA and language en. Speech rate is set to 0.9 (1.0 is the default) to make the TTS speech more understandable.

The animalButtons list is a list of the images used in the sprites.

Earlier, it was mentioned the csv in the File1.ReadFrom response could also be used to populate the animalList and the ListPicker1 Elements. You may wish to change the code in the Screen1.Initialize event handler to populate animalList and the ListPicker1 ElementsFromString using the csv.



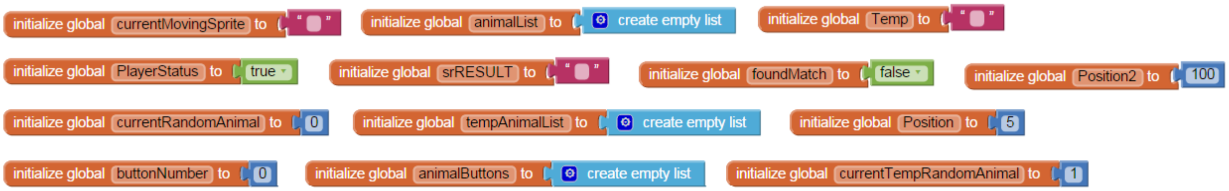
```
when Screen1 .Initialize
do  set TextToSpeech1 . Country to " USA "
    set TextToSpeech1 . Language to " en "
    set TextToSpeech1 . SpeechRate to 0.9
    set Screen1 . TitleVisible to false
    call File1 .ReadFrom
                fileName " //AnimalTableList.csv "
    set ListPicker1 . ElementsFromString to " armadillo,bear,cat,chicke...
    set ListPicker2 . ElementsFromString to " Stop Music,Animal Quiz,An...
    set global animalList to list from csv row text " armadillo,bear,cat,chicke...
    set global animalButtons to make a list ImageSprite3
                                            ImageSprite4
                                            ImageSprite5
                                            ImageSprite6
                                            ImageSprite7
                                            ImageSprite8
                                            ImageSprite9
                                            ImageSprite10
                                            ImageSprite11
                                            ImageSprite12
                                            ImageSprite13
                                            ImageSprite14
                                            ImageSprite15
                                            ImageSprite16
                                            ImageSprite17
                                            ImageSprite18
                                            ImageSprite19
    call setupAnimals
    call startGame
    set Player1 . Source to " car2.mp3 "
    call Player1 .Start
    set Clock3 . TimerEnabled to true
```

Read about the setupAnimals and startGame code blocks in the section on Procedures.

Set the Player1.Source to your background music and start playing (Player1.Start). Simultaneously set the splash screen timer (using the code in Clock3) to true to enable the eight second delay in displaying the Main screen.

Variables

Do not be afraid to use variables. The following are required.



```
initialize global currentMovingSprite to " "
initialize global animalList to create empty list
initialize global Temp to " "
initialize global PlayerStatus to true
initialize global srRESULT to " "
initialize global foundMatch to false
initialize global Position2 to 100
initialize global currentRandomAnimal to 0
initialize global tempAnimalList to create empty list
initialize global Position to 5
initialize global buttonNumber to 0
initialize global animalButtons to create empty list
initialize global currentTempRandomAnimal to 1
```

Resources

Images

The 50 x 50 pixel animal images are “public domain” clip-art from the Internet. The images are resized to fit the app. I researched the clip-art, if any image is not public domain, please inform me and I will remove the image. The animal grid background is made using *Paint*. The image is scaled to a 300 x 300 pixel image size (see the grid images earlier in the tutorial).

Sound

Background music is from an expired copyrighted song by Scott Joplin.   Scott Joplin was born in Texarcana, Texas in 1867. *Car-Barlick* is from http://www.thecompletevictorian.com/TheMusicRoom.html

Mr. Joplin’s copyright appears to have expired in the USA but possibly is still in effect in your country. Use background music appropriate for your version of the app. Many of the sound files available on the Internet have copyright protection. You may be able to use some of these recordings for personal use. Are you musical? Perhaps the best background music is a recording of a musical score created by you.

The animal sounds come from a variety of ‘free’ sources; some are my home recordings. All the sound files were “dumbed down” to low fidelity to make the audio compact, smaller files.

The TTS ‘language’ is set to **USA** in the Screen1.Initialize event handler to provide the correct pronunciation of the language of the game. Change the language code in the Screen1.Initial event handler as appropriate depending on the language you choose to name the animals in your game. Some possibilities for the language codes are described here or more complete here. For example, the following language codes may be appropriate:

English	en
French	fr
German	de
Italian	it
Norwegian	nb

How to “Play” *Animal Canvas*

There are three “ways” to play the ‘game’ and learn.

> **Learning using only the default app screen**.
> Touch an animal name on the grid on the left side of the screen -- the app finds the animal’s picture on the right. Drag an animal from the right to the correct name of the animal on the left and get a message if the match is correct. Use the Select button (the button with three small animals) to find the name of an animal in a list; select the animal name from the list; the app finds the animal, highlighting the creature with an orange highlight.
>
> **Using the Animal Ask button.**
> Use the Select button (the button showing the three animal images) to pick the “Animal Ask” option. Use the orange button that appears to get the app to ask a question. When the speech recognizer icon appears, say the name of the animal. The app knows if the correct name is said.
>
> **Using the Animal Sound Ask button.**
> Use the Select button (the one with the three animal images) to pick the “Animal Sounds Ask” option. Use the orange button that appears to get the app to ask a question about the sound the user hears. When the speech recognizer icon appears, say the name of the animal. The app knows if the correct name is said.

The background music can be turned off to hear the animals better (use the audio button with the sound wave). A second background music track could easily be added.

Issues

Be careful using the speech recognizer in your version of the game. If an animal image shows more than one animal of the same species, the speech recognizer will have issues with the user when a User responds using the plural versus the singular. Within *Animal Canvas* the issue occurs with the **tigers** image. Note, the image is tiger**s**, not **tiger**. To circumvent the plural/singular issue, the game is coded to allow the speech recognizer to accept as true either the word tiger or tigers when referring to this image. Another solution is to only provide images with a single animal.

When a User incorrectly pronounces an animal name, he/she gets a chance to try again. This opportunity occurs eventually, not immediately. Code needs to be written to re-ask a missed question immediately perhaps, however, the code needs to be written so the user does not go into a loop he/she cannot get out of. The work around in *Animal Canvas* is move on to the next animal in the event of an identification mistake or an issue of pronunciation. Eventually the problem animal will reappear. How to handle repeated problems with a User's mispronunciation is not solved. Perhaps an IGNORE button could be added to the app?

The aia File

The *Animals* aia file is xxxx

Important Facts

The tutorial and the *Animal Canvas* app are copyrighted. Please do not slightly modify the tutorial and claim it as your own or post *Animal Canvas* on *Google Play* or on any Web page. Have fun with the app for personal use. Use the algorithms and ideas in your own app and enjoy coding.