|  |  |
| --- | --- |
| Magic 8-Ball is a fortune-telling app. In this app, a virtual magic 8-ball can be asked questions and provide predictions. A user can ask a question, shake the phone, and hear the magic 8-ball’s prediction to their question. This app uses an accelerometer to handle the shaking event, a list of possible predictions, and text-to-speech to have the phone speak the magic 8-ball’s prediction, which is randomly selected from the list of predictions. CSP Learning Objectives:  * create an app that   + uses the *Accelerometer* sensor to respond to shaking events,   + uses the *TextToSpeech* component to convert Text output to speech, and   + randomly selects the 8-Ball's responses from a list variable. | ***[Click to watch Preview Video](http://www.youtube.com/watch?v=G_mdFf7phJs)*** |

## 

# Getting Ready

To begin this lesson open [App Inventor with the Magic 8 Ball](http://ai2.appinventor.mit.edu/?repo=templates.appinventor.mit.edu/trincoll/csp/unit3/templates/Magic8BallTemplate/Magic8BallTemplate.asc) template. It provides the media you need for this project and a partial version of the User Interface. When the project opens -- be patient, it may take a moment -- use the *Save As* option to rename it Magic8Ball.

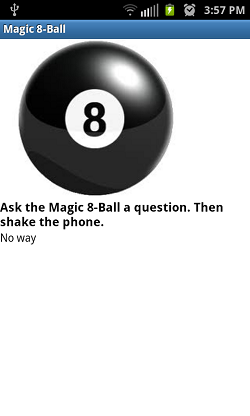
# 

# 

# Magic 8-Ball Tutorial

([Video Tutorial](http://www.youtube.com/watch?v=gtY1qGXBxjw))

## The Magic 8-Ball UI



The UI for our Magic 8 Ball app will consist of four categories of *Components:*

*Basic Components* - Image, two Labels

*Media Component* -Sound

*Sensor Component* - *Accelerometer*

*Other Stuff Component - Text-To-Speech*.

The Accelerometer detects the shaking of the phone. The shaking event is used to simulate the shaking of the Magic 8-Ball and results in a prediction being given.

The Image is used to display an image of a Magic 8-Ball.

As you can see, there is a sentence asking the user to ask the Magic 8-Ball a question. This is the question label. Below this is another label, the answer label, which will display the Magic 8-Ball’s prediction.

The Sound is used to play a sound when the phone has been shaken and a prediction is made.

The Text-to-Speech component is used to speak the Magic 8-Ball’s prediction.

**Adding the Magic 8-Ball**

You have used the Button component and Canvas component and added an image background to each. But perhaps you would just like to add a still image. You can do so by using App Inventor’s Image component. To add a Magic 8-Ball image:

1. Drag and drop an Image component from the Palette’s *User Interface* category in the Viewer.
2. Set the Image’s picture to one of the Magic 8-Ball images provided in the template.

**Adding the Labels**

This app contains two labels: a question label that gives instruction to ask a question and an answer label that displays the Magic 8-Ball’s prediction. Both labels are provided in the template. Feel free to modify these labels to your liking.

**Adding the Accelerometer**

An [*accelerometer*](http://ai2.appinventor.mit.edu/reference/components/sensors.html#AccelerometerSensor) is a sensor that can detect shaking of a device and it can measure acceleration in three dimensions. The neutral position is when the device is laying flat on its back and the accelerometer can detect when the device is moved out of this neutral position. You can use the accelerometer for the shaking event which will give a Magic 8-Ball prediction. To add the accelerometer to your app:

1. Drag and drop an Accelerometercomponent from the Palette’s *Sensors* category into the Viewer. It will be named *AccelerometerSensor1* and it will appear at the bottom of the Viewer under as a *non-visible component.*

**Adding the Sound**

The Sound component will be used to play a sound when the phone is shaken. The Sound component has already been added in the template. Now you must set the Sound’s source to one of the sounds provided in the template.

**Adding Text-to-Speech**

The [*Text-to-Speech*](http://ai2.appinventor.mit.edu/reference/components/media.html#TextToSpeech) component is used to have your device speak text audibly. In order for this component to work, you must have TTS Extended Service on your device. To add a text-to-speech component:

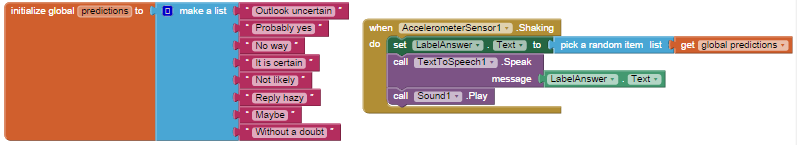
1. Drag and drop a Text-to-Speech component from the Palette’s *Media* drawer into the Viewer. It will be named *TextToSpeech1* and it will appear at the bottom of the Viewer under as a *non-visible component.*
2. View the properties for *TextToSpeech1*. The default country code is *USA* and language is *eng*. If you want to use the default values, you can just leave those properties blank. If you want to change the defaults, [view the list of country and language codes:](http://beta.appinventor.mit.edu/learn/reference/components/other.html#TextToSpeech)

# Coding the Behavior

## Now that you have all of the components set up, you need to set up the functionality of the app. Remember that this app should simulate the process of asking a Magic 8-Ball a question and shaking the Magic 8-ball to read and hear it’s answer.

## Handling the Shaking Event

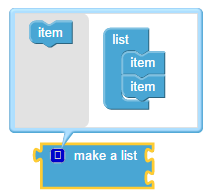
The only event in this app is the *Shaking event*. The Shaking event results in the Magic 8-Ball’s prediction being displayed in the answer label, the text-to-speech component speaking the prediction, and a sound being played. The completed Shaking event looks like this:



Begin by getting the *Shaking event handler* from the *AccelerometerSensor1* drawer.

**Making a List**

Now, you will need to create a list of possible predictions that the Magic 8-Ball might make. This is our first look at lists. We will be making lots of apps that use lists to organize data.To make a list in App Inventor, you will first need to initialize a global variable. This time, instead of connecting a number block to the initialize global variable block, you will need to connect the *make a list* block from the Lists drawer.



By default, the *make a list* block has two slots. This means you can only add two items to this list. In order to increase the number of items, click the blue plus sign to open the list editor. This shows you an *item* block on the left and a list with two items on the right. You can increase the number of items allowed in the list by dragging and dropping the *item* block from the left over to the list of two items on the right. Do this repeatedly until the number of items allowed in your list is what you would like it to be.

Items that you add to your list may be numbers or text. For this app, you will need to add text items. Get an empty *text* block from the Text drawer and add it to your list. Change the text to say a possible prediction. For example:



Repeat this for each possible prediction that you would like to have for your Magic 8-Ball.

Once you have finished making your list, here are some things to consider:

* lists have a *length* property that keeps track of how many items or elements are in a given list. You will use the *length of list* property in a future lesson. What is the length of list you made?
* lists are *indexed*, or numbered, starting with 1, which means that you can retrieve any item from a list by giving its index. You will use *indexing* in a future lesson. Select a random prediction in your list. What is its index?

**Picking a Random Item From a List**

Now that you have a list of possible predictions, you need to randomly select one prediction to be displayed and spoken each time the Magic 8-Ball is asked a question and the phone is shaken. To do this you, you could randomly pick a prediction from the list using its index, however App Inventor provides us with a *pick random item* block. Use this block to set the *LabelAnswer.Text* to be a random item from your list:

1. Get a setter LabelAnswer.Text block from the LabelAnswer drawer.
2. Get a *pick a random item* block from the Lists drawer
3. Set the list for the *pick a random item* block to be the list you just created.

It should look like this:



**Using Text to Speech**

Once you have set LabelAnswer to be one of the items from the list of predictions, you can now tell the Text-to-Speech component to speak the text in the label. In order to have the Text-To-Speech component speak, you must assign it a message to be spoken.



For your app:

1. Get a call *TextToSpeech1.Speak* block from the TextToSpeech1 drawer
2. Set the message to be spoken as the getter *LabelAnswer.Text* block from the LabelAnswer drawer.

**Playing the Sound**

Play a sound when the device is shaken. Recall how to do this from the I Have a Dream app.

## Testing the App

Now that you have a fully functioning Magic 8-Ball, it’s time to test your app on your device. Connect App Inventor to your device (phone or tablet) and run the app:

* Start the ***MIT AI Companion*** app on your device.
* Click **Build** and select  ***AI Companion***.
* If the app doesn’t start downloading to the device almost immediately:
  + Make sure both your laptop and device are on the same Wifi network.
  + Try refreshing the App Inventor page in the browser and try again.

If your code is correct, the app should speak a response every time the phone is shaken.

***Nice work! Complete the Self-Check Exercises and Portfolio Reflection Questions as directed by your instructor.***