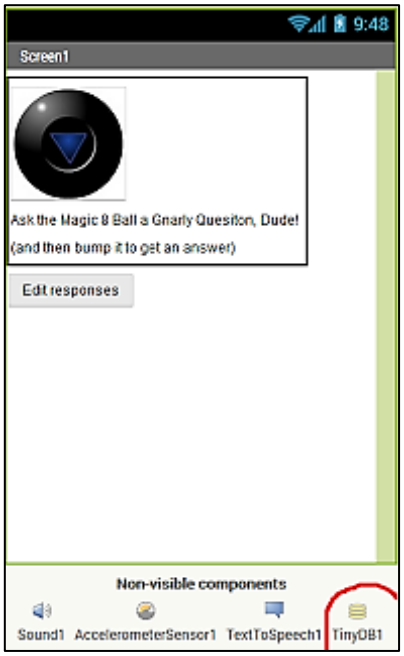
Class Session 4: Activity 1 Step-by-Step Instructions

Add a database to your Magic 8 Ball!

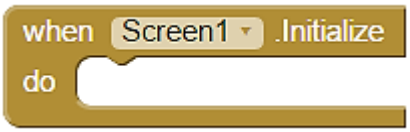


- **Open** your "Magic 8 Ball" app from the previous session.
- Add a database on the home screen by adding a **TinyDB** to your screen from the **Storage** drawer in **Palette** menu.
  - It will appear in the 'Non-visible components' bar after you drop it onto the page.
  - **Each "page" that interacts with your app's database needs to have this element added**, so if you add another screen, you will need to add a TinyDB component to it as well.
- Open the **Blocks** Editor.
- Click on the **Database** component to view various options.

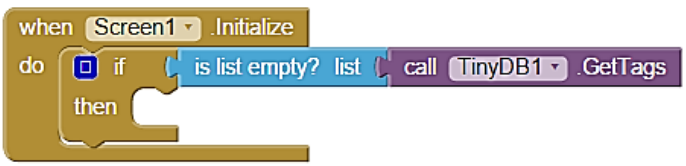
What do you think you need to do first?

If you want your "Magic 8 Ball" to load responses, it needs to load responses from the database. Does your database currently contain any of those? No, it does not.

- Next, your app has to **add some options** to the database. Take note, you ONLY need it to do that once, when the database is empty, when the app loads for the first time. If you have run this app on your device previously, it will remember what was already loaded. That's the benefit of a database!
- You need to **set the values to load**, ONLY if the database is empty. Otherwise, you'd be resetting your values every time the application loaded.
- To get the database ready the first time the App loads, you need to **run a command immediately** when the app opens. You can use the ***Screen1.Initialize*** event handler, found under the **SCREEN** component, to do this. Add a **Screen1.Initialize** to your viewer.

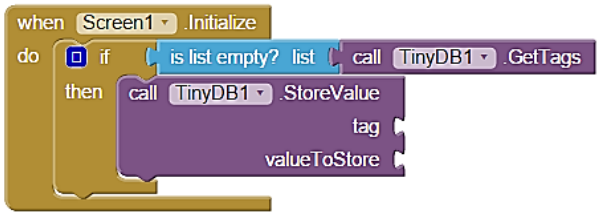


- Consider when you want this command to run. Does it always need to run, or only if there is no data in your database? To check that, add an ***If - Then*** control block from the built-in drawer.
  - How can we test that logic? Say it out loud: *If* the database does not contain any data, *then* call in some new values.
  - That means, if a list of tags from the TinyDB1 is empty, THEN add in new values. If it already contains values, there is nothing to do; TinyDB will use tags that have already been stored.



This uses two component blocks: an **event handler** for Screen1, and a **command** for TinyDB1. There are also two built-in blocks: a **list** block, and a **control** block.

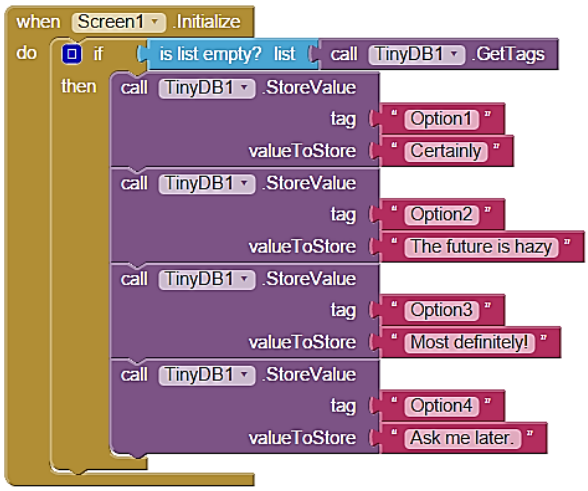
When Screen1 *initializes*, *if* the list of TinyDb Tags is empty, *then*...we want to call tags for the list. If it has already run and stored any, it will use those instead.



There is room for two blocks in the *.StoreValue* block - a **TAG value** and a ***ValueToStore***.
What do you think the difference is?

The **TAG** is like the location of a cell on a spreadsheet. The **TAG is the title of the information** you are storing, and it's used so that your app knows what information to look for when you want it back. What's the **ValueToStore**, then? It's **the value you want associated with the TAG**; in this case, it will be the text that Magic 8 Ball gives you when you shake it or press the button. Each time you shake it, the 8 Ball app will randomly select a TAG to respond, and when it calls on that TAG, it will you show you the VALUE associated with the tag.

- Now, fill out your database with a selection of **tags and values**.



Awesome. So now your database has initialized, but your 8 Ball is still looking at the LIST you defined originally. Let's change that. Instead, we want to **ask the database for information**. This gets a little complex, but we can do it!

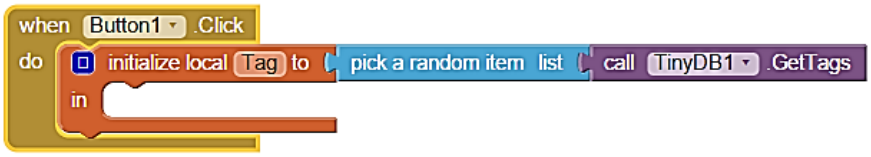
- We first need the app to pick an item from the database. That's easy. You can replace that random list with a command to ***Call TinyDB.GetTags***. Would this get you what you want?

No. This only calls the TAGS associated with the information, which are the titles of the options, but not the values of the options themselves. You want to **show the VALUE of the TAG**.

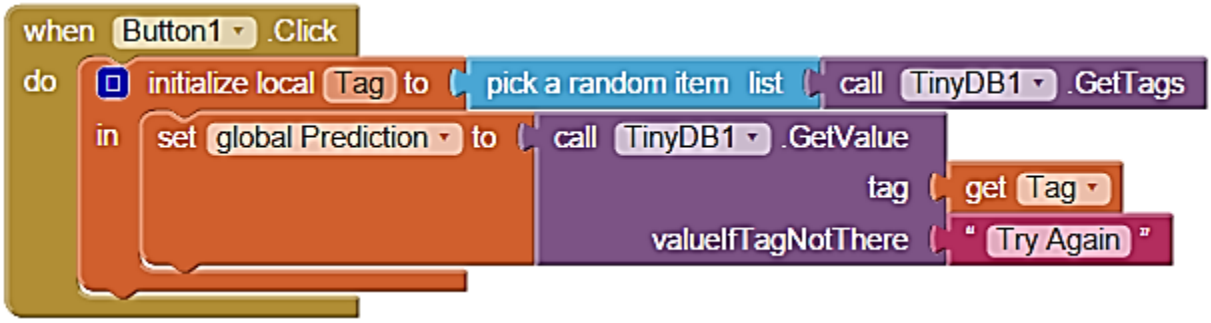
Here's one way to do that:

- First, we'll create a ***global variable*** for the "Prediction". Variables can change while the app runs, and can be any type of value. A variable is useful here because our App will likely get asked several questions, so the prediction will need to change each time it's needed. **Global variables can be called on anywhere in our code, so they are usually on their own line and don't snap into other blocks.** Our Prediction will be **text**, so this global variable needs a **text box**. Since it won't need a prediction when it initializes, that text box can be empty for now.

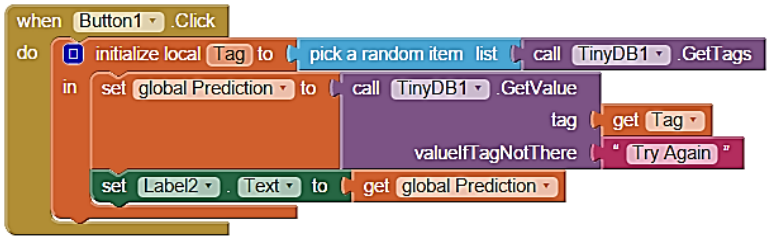
- We need **a second variable** to pick a random tag from our TinyDB Tags list. This time, we'll use a ***local variable***; these are only used in the procedure defined within it. We will create a local variable 'Tag' to represent a randomly chosen option for our global 'Prediction' variable.



- So, when **Button1** is clicked, local variable 'Tag' will be initialized and pick a random item from a list of TinyDB1's tags. But what should it do with that information? We need to **tell 'Prediction' that is should become the value of 'Tag'**.

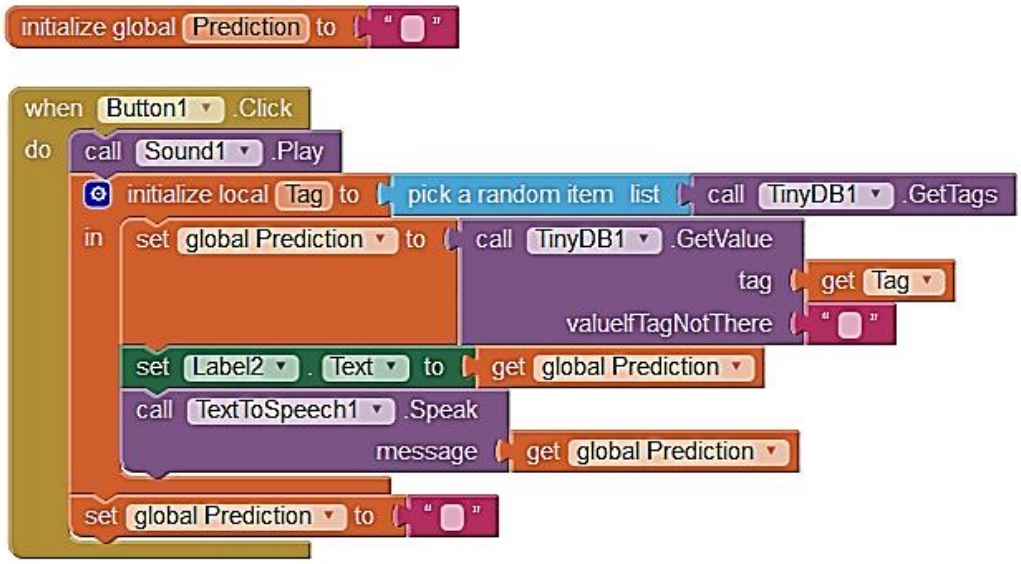


- The second purple command block is using ***.GetValue*** so that the global variable 'Prediction' (which we initialized above) will be set to the **VALUE** of that tag. If a Tag can't be found, it will use the value "Try Again". But how will we know what response 'Prediction' becomes? We need to **display it on our screen**!
- **What component shows us the response text?** We can still use that same one; in this case, it's **Label2**.



Label2 will now display the text of whatever value the 'Prediction' variable uses.

- Now that our Magic 8 Ball is using the database instead of the list, you can go in and **add in the options from the previous version**. You can still play sounds, speak the message, or use the accelerometer. Here is an example of code that will play a sound when the button is clicked, use our database code, and use TextToSpeech to read the answer out loud.



Look at that! Now you have a Magic 8 Ball driven by a simple list database.