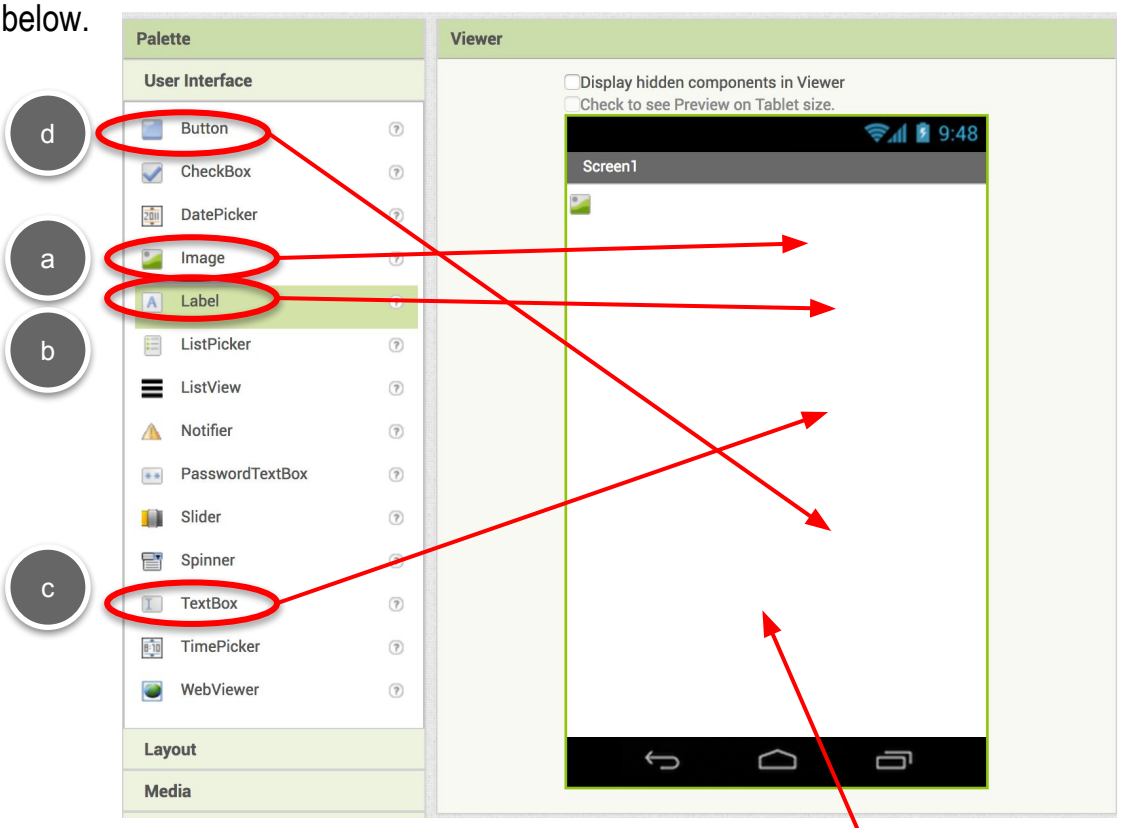
ITUNES MUSIC APP

In this unit you will use a Web API to get artist and music information from iTunes to display and play in your own app!

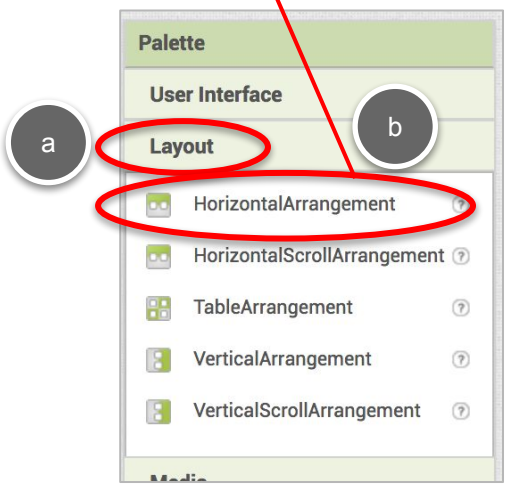
START HERE

Import the iTunes template project in App Inventor. In the Designer window, we'll add all our User Interface components for our app.

1 From the Palette window, drag to the View window an **Image**, **Label**, **TextBox**, and **Button**. Note that they appear in the order you drag them, from top to bottom, so follow the order a,b,c,d below.



2 Click on Layout, and drag in a **HorizontalArrangement** below the button.



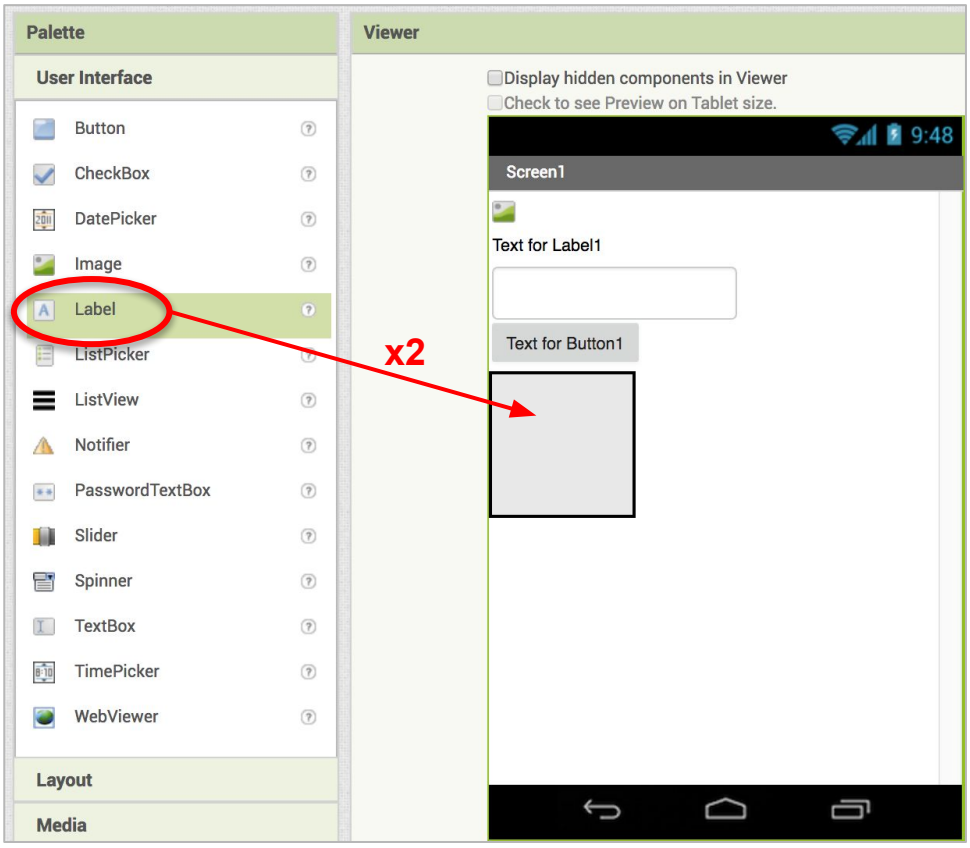
Credit to Aubrey Colter, MIT M.Eng. 2016 for app and tutorial idea

ITUNES MUSIC APP

DESIGNER

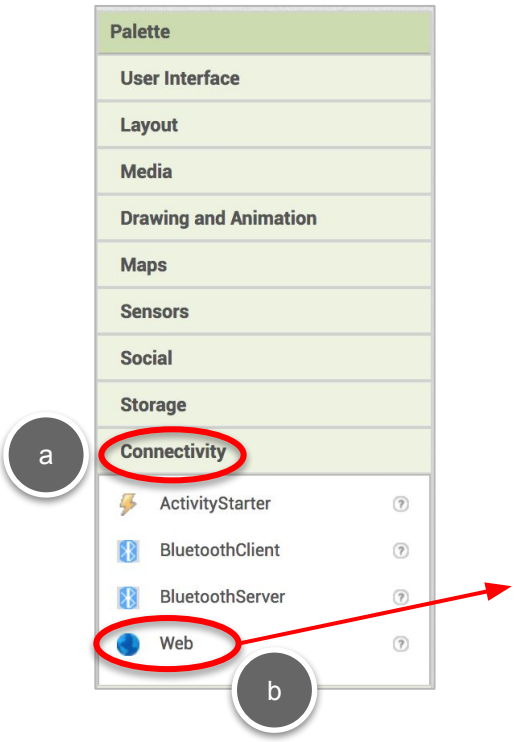
The **HorizontalArrangment** lets us place components next to each other horizontally, rather than the default, vertically.

3 Drag two **Labels** into the **HorizontalArrangment**.

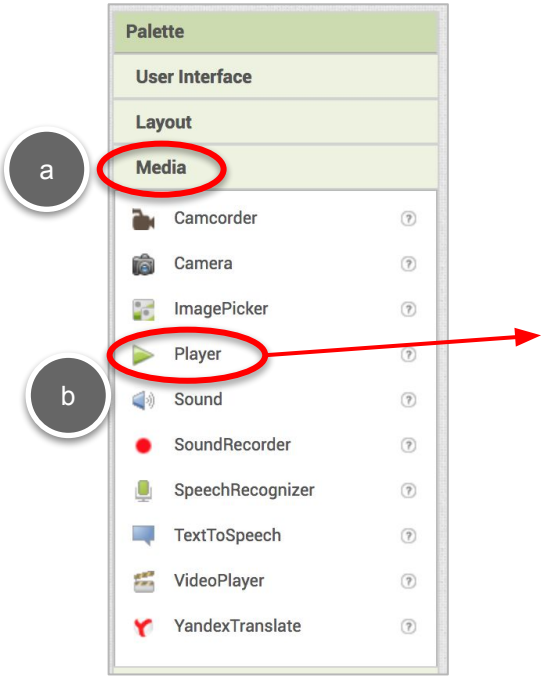


We'll also need two more components, **Web** and **Player**.

4 Drag in a **Web** component from the Connectivity drawer.



5 Drag in a **Player** component from the Media drawer.

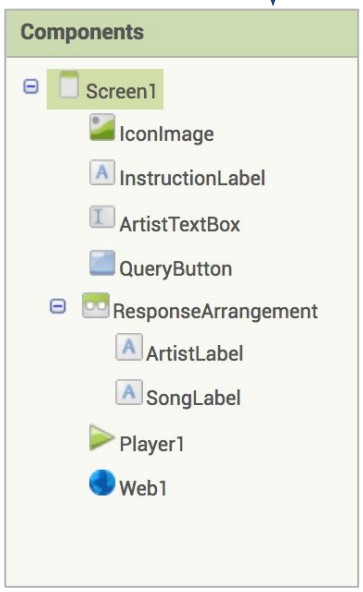


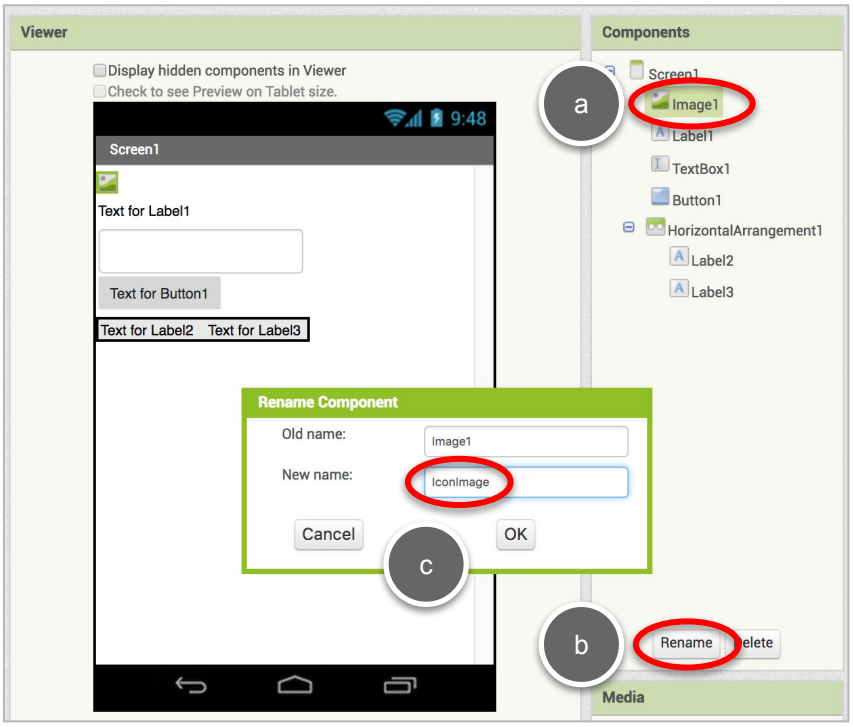
ITUNES MUSIC APP

DESIGNER

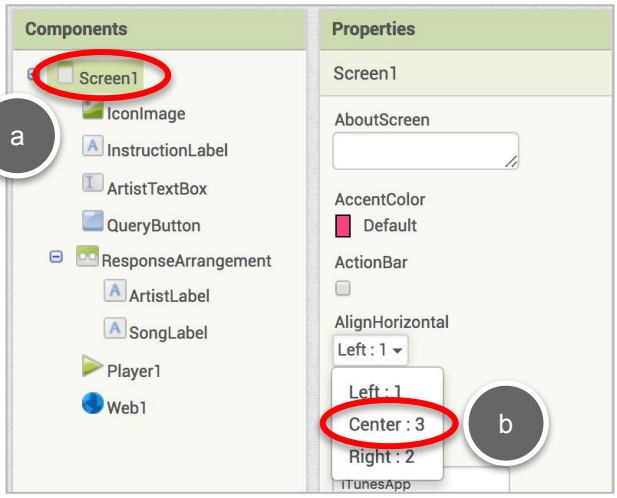
It's good practice to name your components descriptively, so you can identify which component is which when coding.

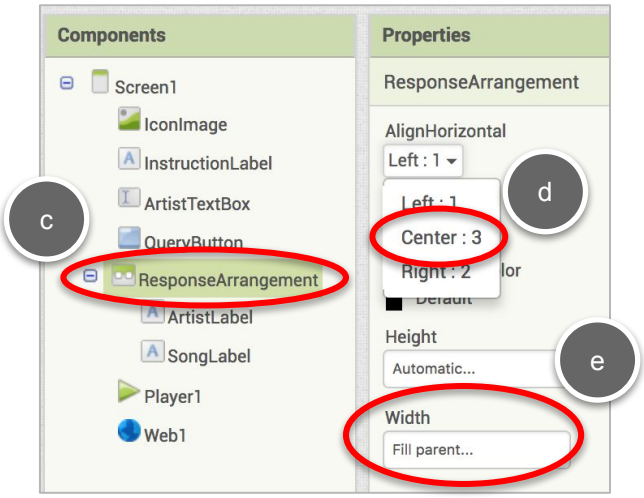
6 Rename all your components as shown below.





7 Let's make the user interface look a bit better. Change the *AlignHorizontal* property of both **Screen1** and **ResponseArrangement** to "**Center:3** ".
Change **ResponseArrangement**'s *Width* to "**Fill Parent**".



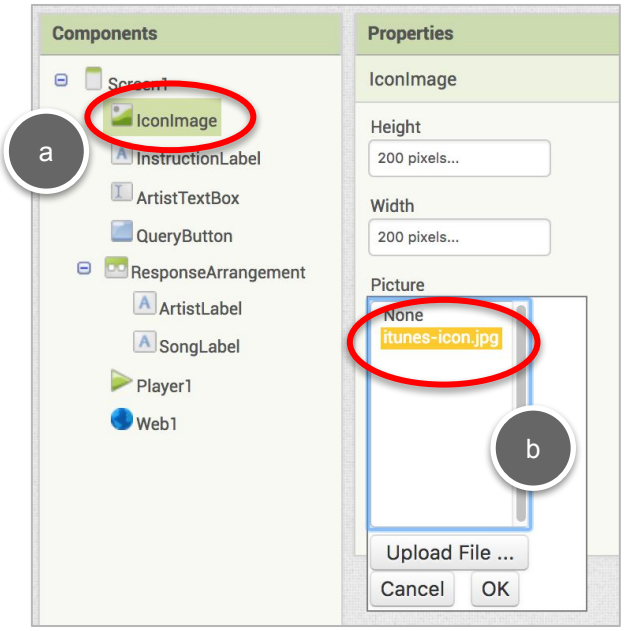


ITUNES MUSIC APP

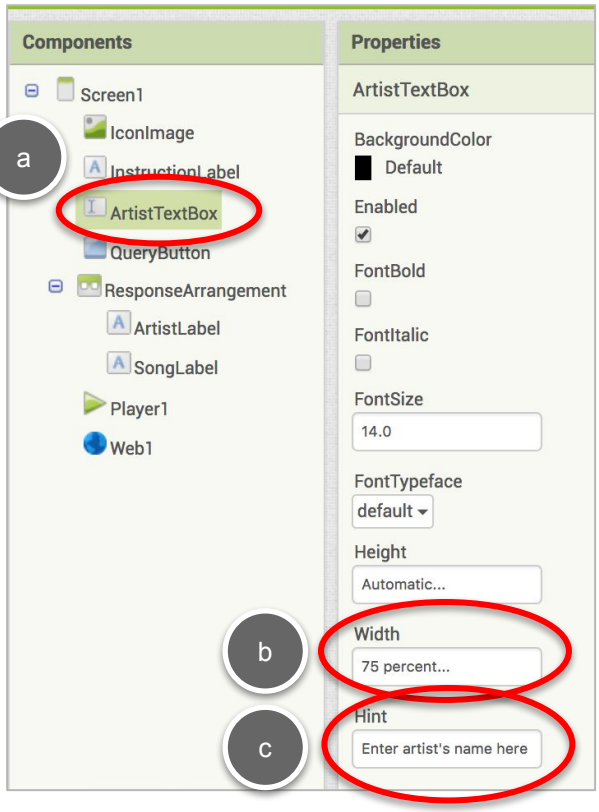
DESIGNER

Let's make a few more changes to make the user interface complete.

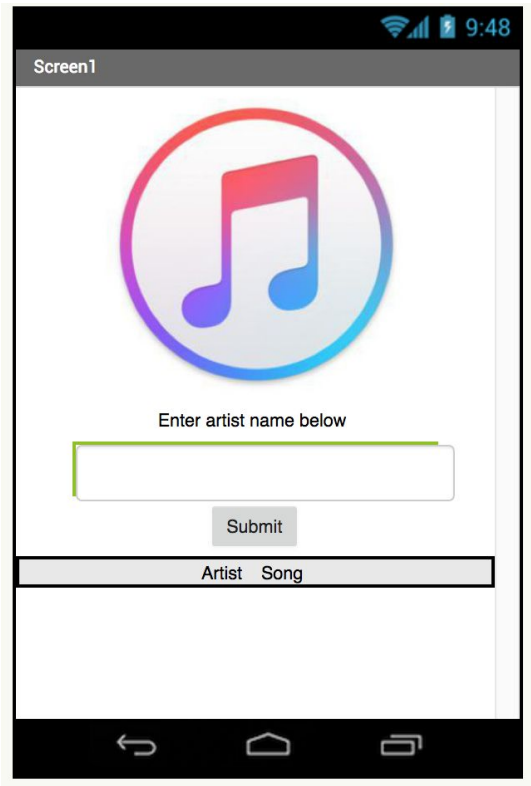
8 Change the *Picture* for **IconImage** to **itunes-icon.png**, which was preloaded in the template for you.



9 For **ArtistTextBox**, change its *Width* to **75%**, and its *Hint to* "**Enter artist's name here**".



10 And finally update the *Text* properties for Labels and Buttons so your user interface looks similar to the screen to the right.



ITUNES MUSIC APP

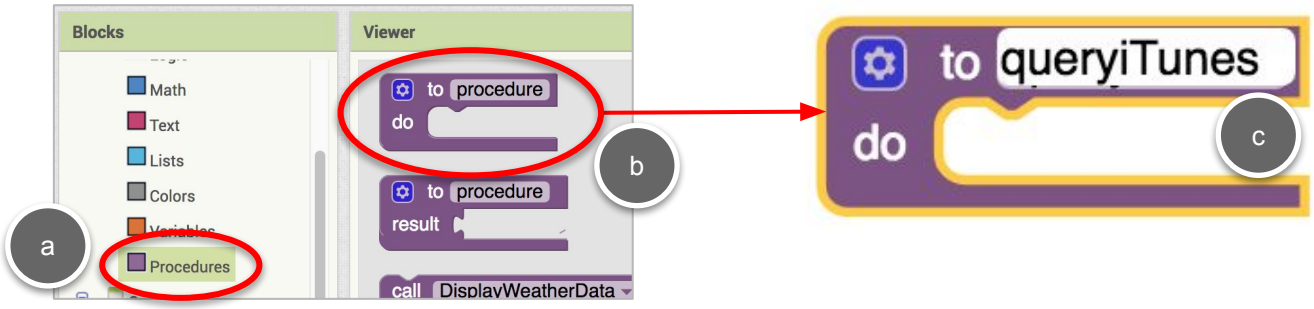
PROCEDURES

A procedure is a way to take a set of blocks and give them a name, so they can be used in multiple places, by "calling" the procedure. It is a way to organize code blocks by a particular function or action.

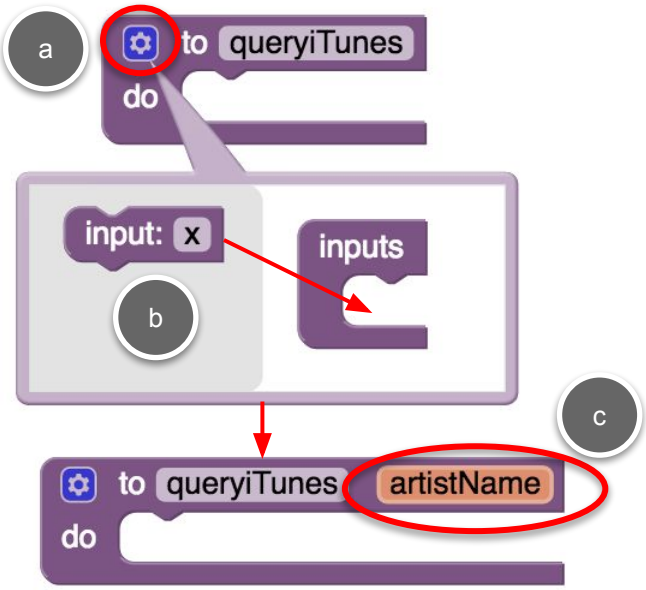
In this case, we're going to make a procedure that will query the iTunes server.

11 Switch to the Blocks Editor. - - - - - - - - ➔

12 Drag out a **to procedure** block from the Procedures drawer. Rename the procedure "**queryiTunes**".



13 Click on the blue mutator button, to add An input parameter to the procedure. - - - - ➔ Rename it "**artistName**".



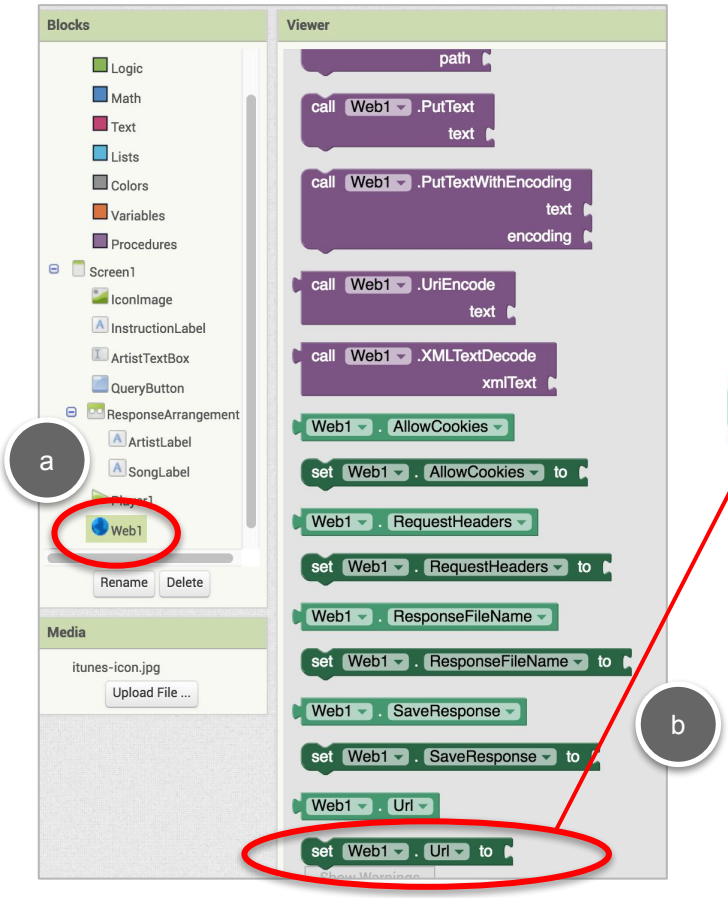
.

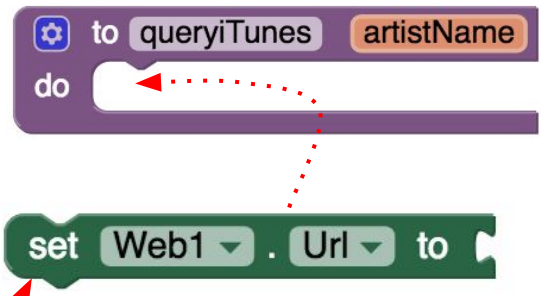
Input parameters allow you to pass information to a procedure. Parameters generalize a procedure, because the procedure can work differently, depending on the input.

ITUNES MUSIC APP

PROCEDURES

14 Drag out a **set Web1.Url** block and snap in to the procedure.





We will use the **Web** component for our API call to iTunes. We will set Web.Url to the iTunes API web server.

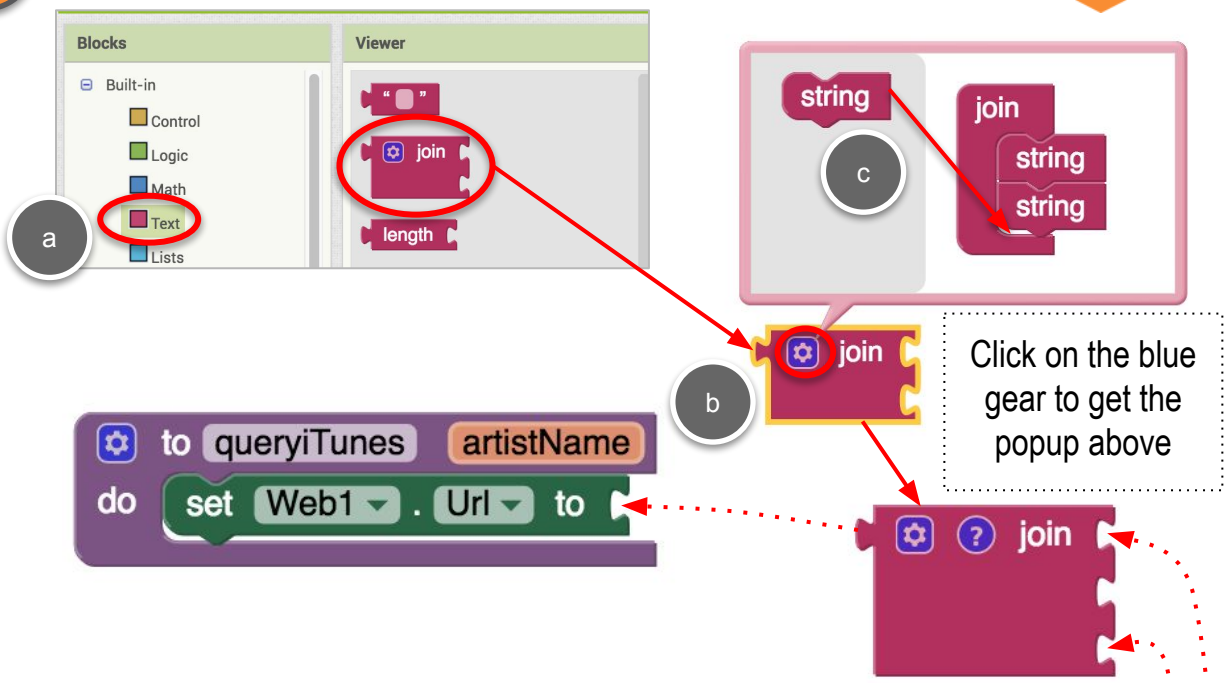
ITUNES MUSIC APP

QUERY ITUNES

Use a Text **join** block to make the query string to send to iTunes.

The iTunes Web API server is at itunes.apple.com. By adding a "search?term=" to the end of the URL, we can ask iTunes for information on our favorite artist.

15 Drag out a **join** block, and add a third input slot.



16 Add the following text strings in the first and third slots.

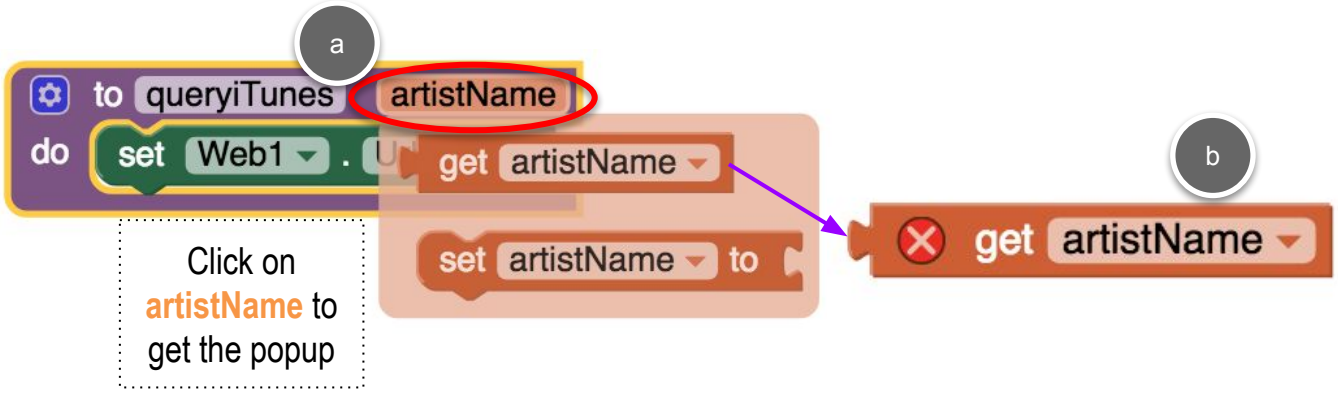
"&Limit=1" tells iTunes to give us the first result it finds in the database

ITUNES MUSIC APP

QUERY ITUNES

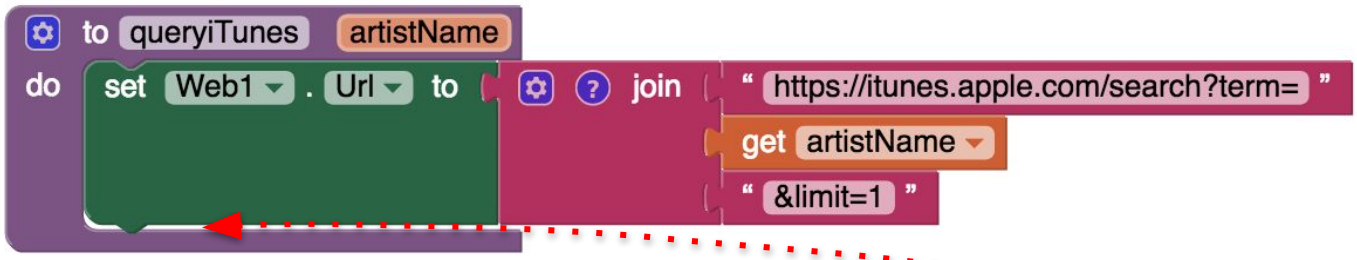
- ❑ Add the input **artistName** in the middle slot of the join block.

.

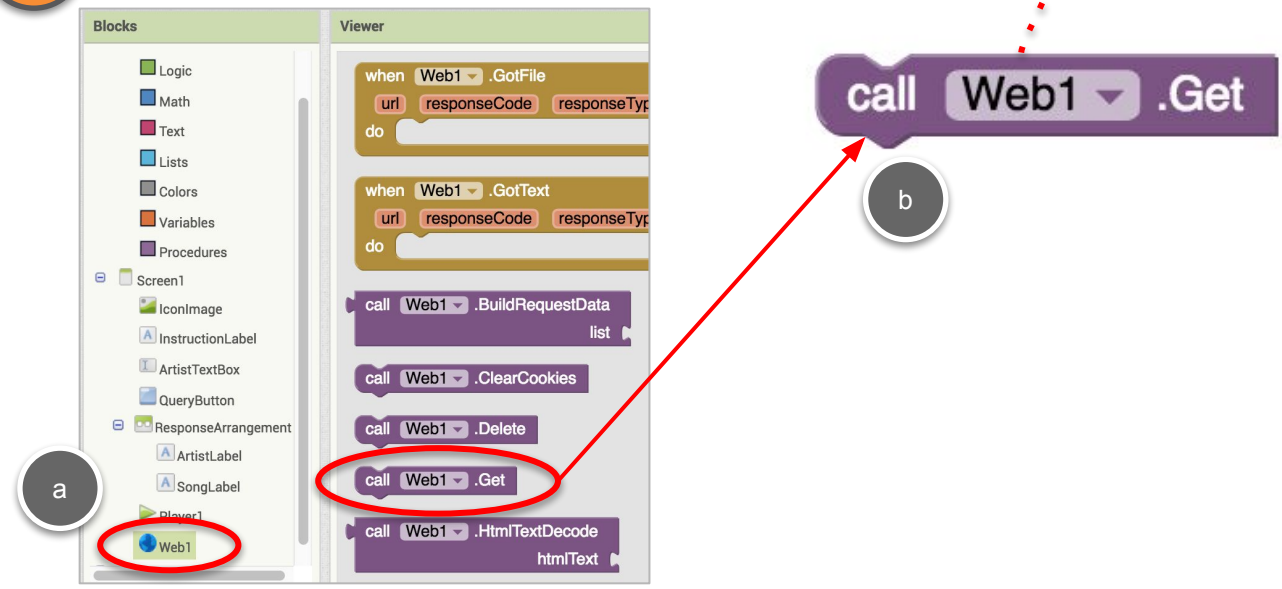


ITUNES MUSIC APP

QUERY ITUNES



18 Finally snap in a **Web1.Get** to go out to iTunes with the request.



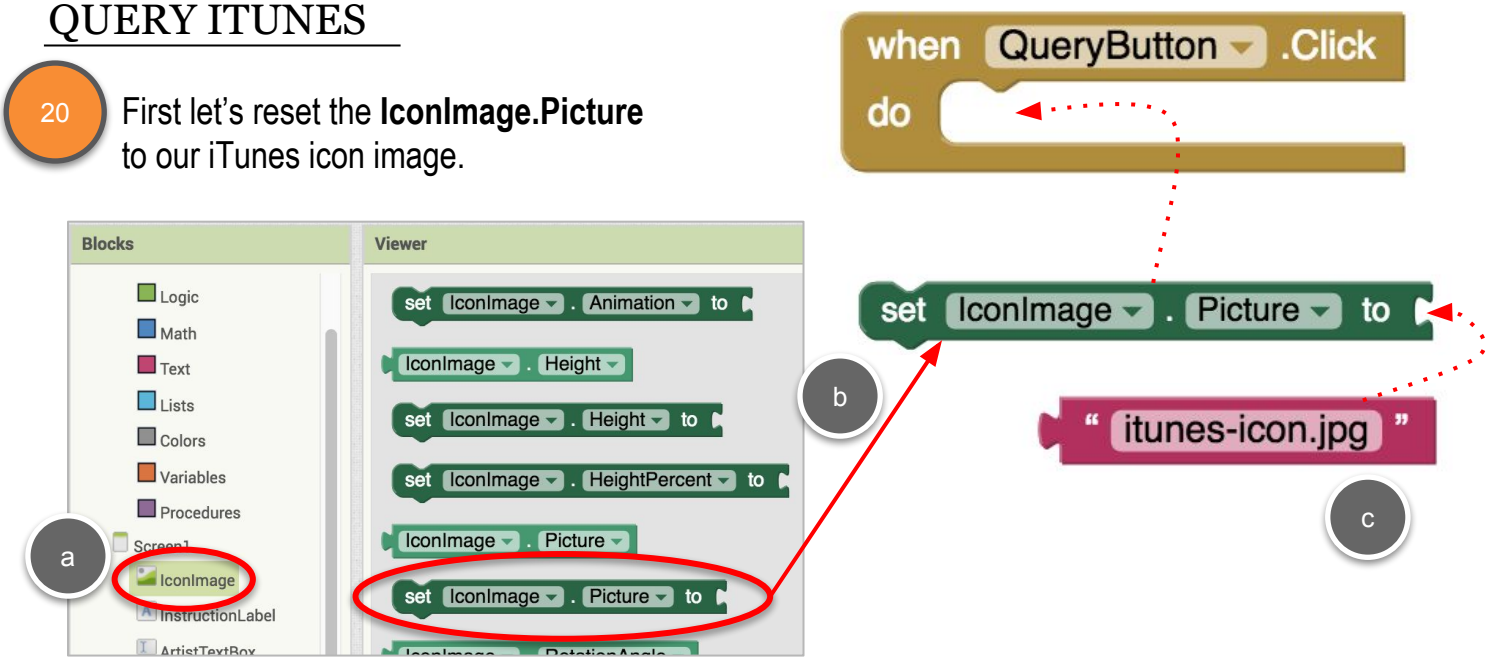
Now let's code the **QueryButton** that will trigger this request.

19 Drag out a **QueryButton.Click** event block.

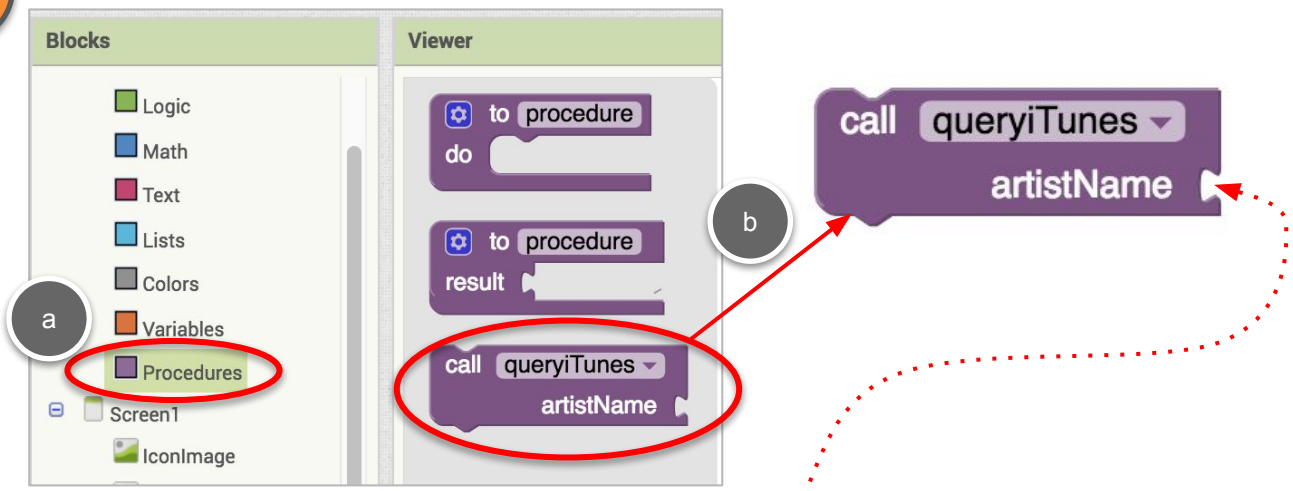
ITUNES MUSIC APP

QUERY ITUNES

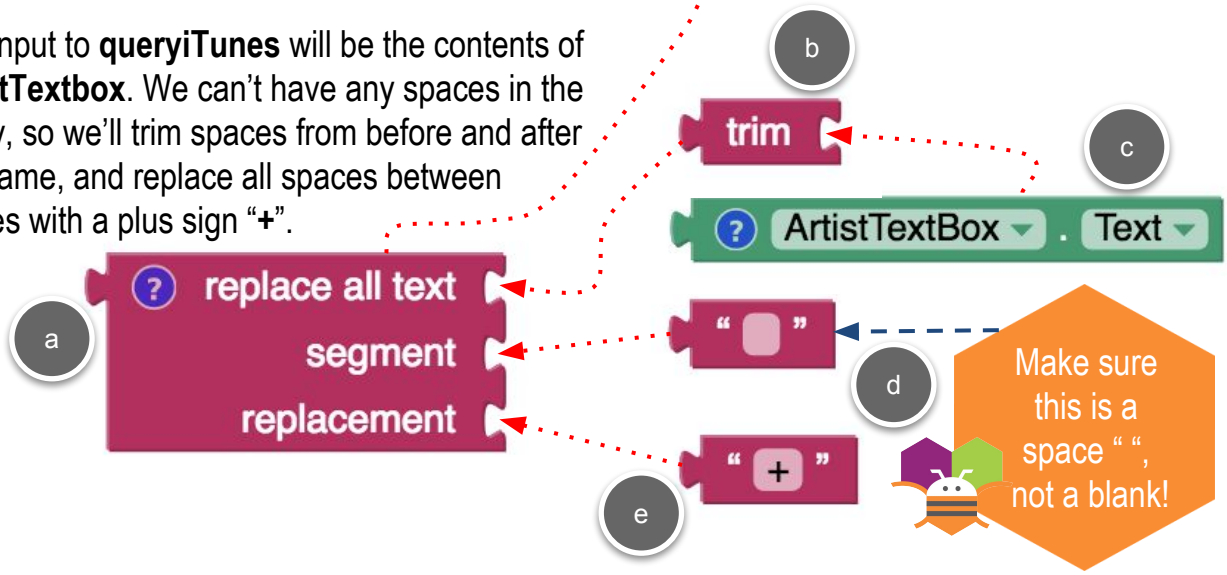
20 First let's reset the **IconImage.Picture** to our iTunes icon image.



21 Then call the **queryiTunes** procedure.



22 The input to **queryiTunes** will be the contents of **ArtistTextbox**. We can't have any spaces in the query, so we'll trim spaces from before and after the name, and replace all spaces between names with a plus sign "**+**".



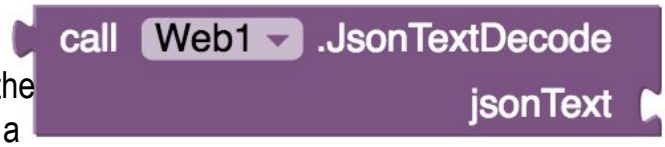
ITUNES MUSIC APP

GET RESPONSE

When you make a Web API request with the **Web1.Get** block, it waits until a response is received, and it triggers the **Web1.GotText** event. Here is an example of a response, which may look confusing.


```
{
 "resultCount":1,
 "results": [
{"wrapperType":"track", "kind":"song", "artistId":136975, "collectionId":401186200, "trackId":
401187150, "artistName":"The Beatles", "collectionName":"Abbey Road", "trackName":"Here Comes the
Sun", "collectionCensoredName":"Abbey Road", "trackCensoredName":"Here Comes the Sun",
"artistViewUrl":"https://itunes.apple.com/us/artist/the-beatles/136975?uo=4",
"collectionViewUrl":"https://itunes.apple.com/us/album/here-comes-the-sun/401186200?
i=401187150&uo=4", "trackViewUrl":"https://itunes.apple.com/us/album/here-comes-the-sun/401186200?
i=401187150&uo=4",
"previewUrl":"https://audio-ssl.itunes.apple.com/apple-assets-us-std-000001/AudioPreview71/
v4/46/48/7d/46487d90-d40c-7c47-7285-5edbfd0fd2c0/mzaf_5516723347634890825.plus.aac.p.m4a",
"artworkUrl30":"http://is2.mzstatic.com/image/thumb/Music/v4/40/d0/29/40d029b5-4c32-53d2-69d1-
ea04a513c345/source/30x30bb.jpg", "artworkUrl60":"http://is2.mzstatic.com/image/thumb/Music/v4/40/
d0/29/40d029b5-4c32-53d2-69d1-ea04a513c345/source/60x60bb.jpg", "artworkUrl100":"http://
is2.mzstatic.com/image/thumb/Music/v4/40/d0/29/40d029b5-4c32-53d2-69d1-ea04a513c345/source/
100x100bb.jpg", "collectionPrice":7.99, "trackPrice":1.29, "releaseDate":"1969-09-26T07:00:00Z",
"collectionExplicitness":"notExplicit", "trackExplicitness":"notExplicit", "discCount":1,
"discNumber":1, "trackCount":17, "trackNumber":7, "trackTimeMillis":185733, "country":"USA",
"currency":"USD", "primaryGenreName":"Rock", "isStreamable":true}]
}
```


App Inventor provides a block, **Web1.JsonTextDecode**, that puts this text into a series of lists, so that you can extract the pieces you want by using **select list item**. In the text above, think of the brackets "**{**" and "**[**" as the start of a new list. And each comma is a separator between list items.



The large text block above that starts with "**wrapperType**" is another kind of list organizer, called key/value pairs. For example, *wrapperType* is the key, and *track* is the value.

App Inventor supplies a block called **look up in pairs** that we can use to extract specific information using the key, once we've stripped out all the extra information at the beginning of the text string.

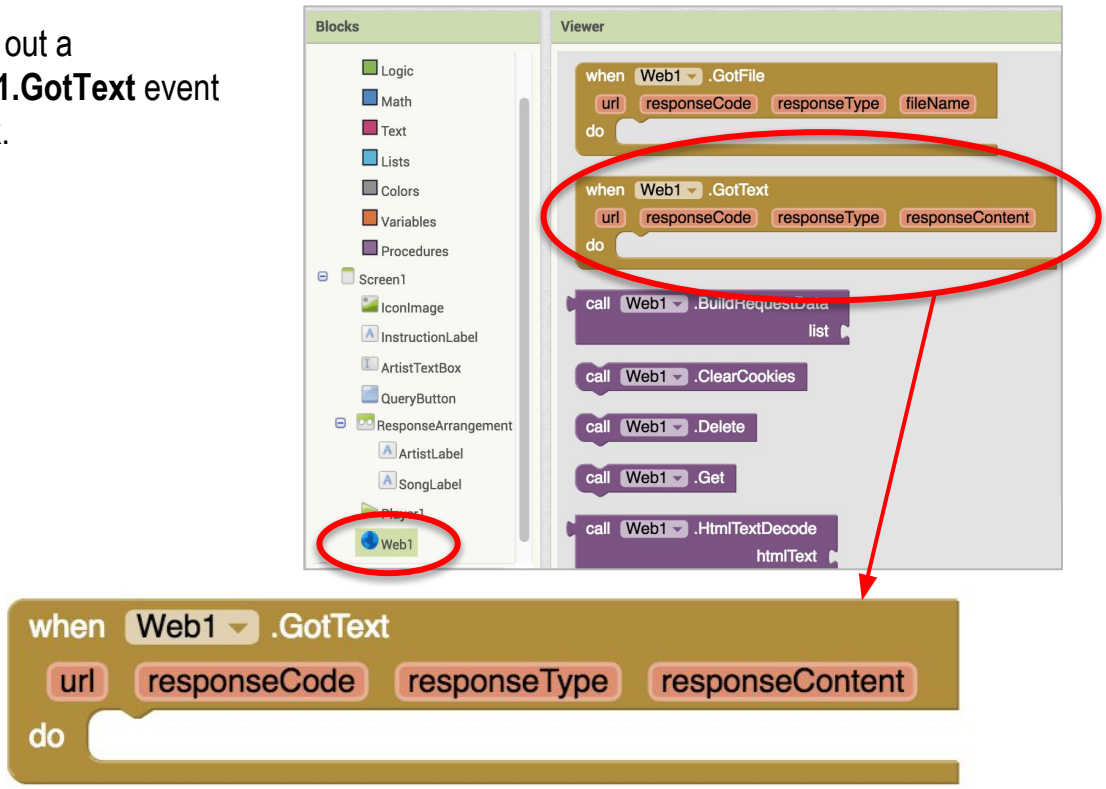
We'll use these different blocks on the next page to extract just the information we need for our app.

ITUNES MUSIC APP

GET RESPONSE

When you make a Web API request with the **Web1.Get** block, the apps waits until a response is received, and then the app triggers the **Web1.GotText** event.

23 Drag out a **Web1.GotText** event block.



We need to strip out the extraneous text to get to those key/value pairs. There are a few steps we need to take to do that, so we're going to store the text in a variable as we do it.

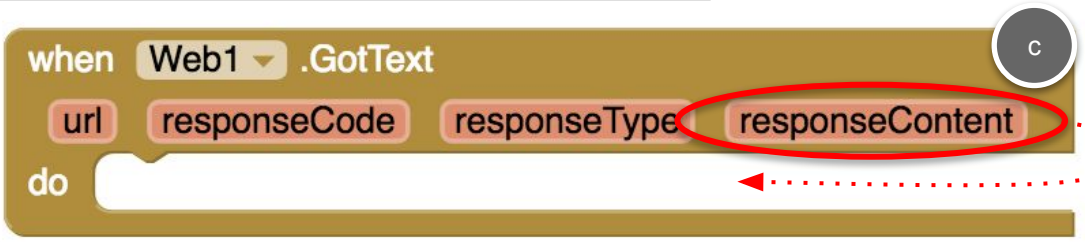
Variables are a way to store information by giving it a name, and referencing the name in our code. We can also change the value of a variable if we need to. We will do that as we strip away the extra text to get to the key/value pairs.

The Variables drawer holds the blocks for variables. Makes sense!

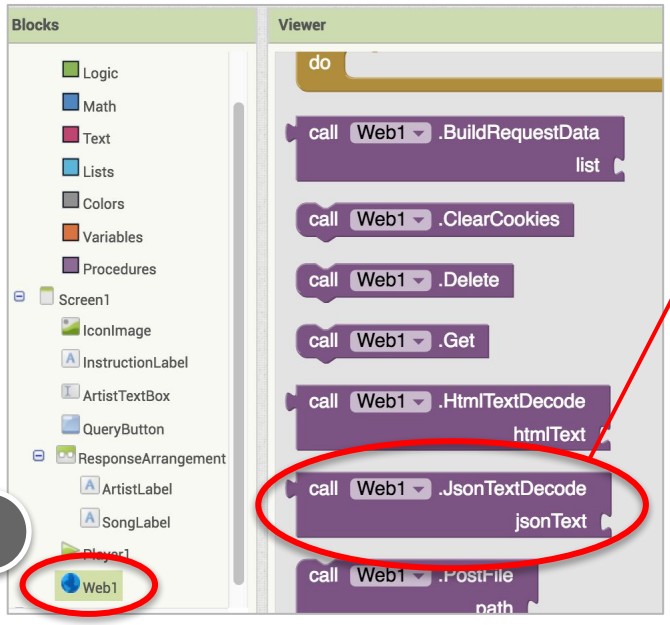
ITUNES MUSIC APP

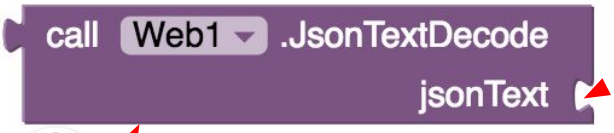
GET RESPONSE

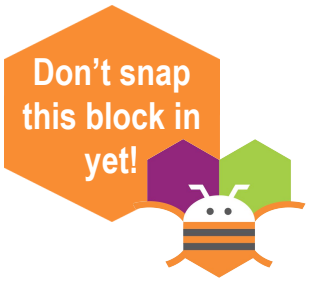
24 The first step is make a local variable called "**responsePairs**". Local means it's only being used within the event block.



25 We'll use **JsonTextDecode** to put the long text response into lists.





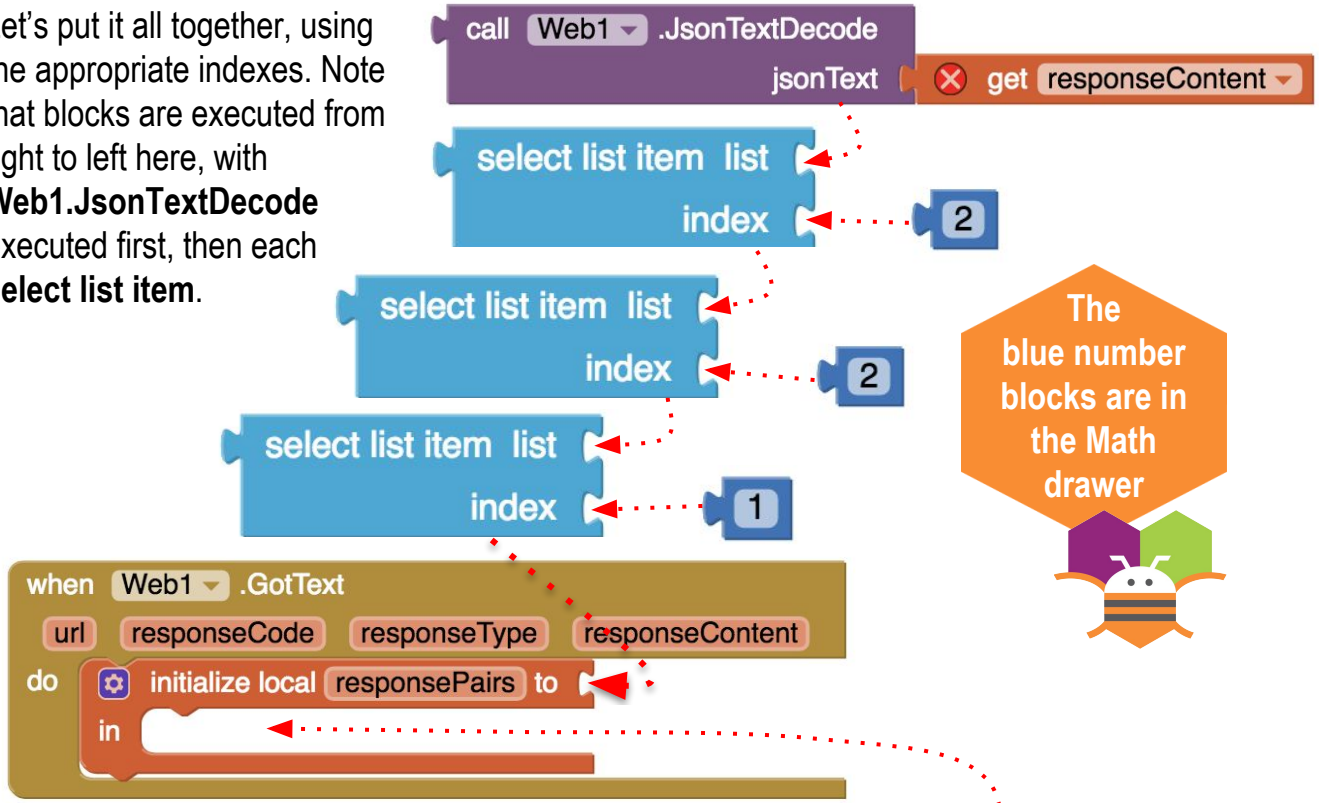


ITUNES MUSIC APP

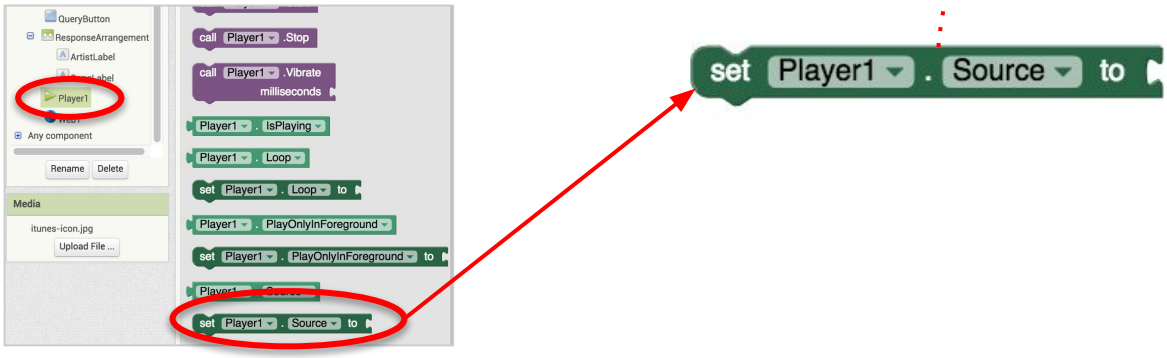
GET RESPONSE

26 Then we'll need <u>three</u> **select list item** blocks, to get to the inner pair lists.

27 Let's put it all together, using the appropriate indexes. Note that blocks are executed from right to left here, with **Web1.JsonTextDecode** executed first, then each **select list item**.



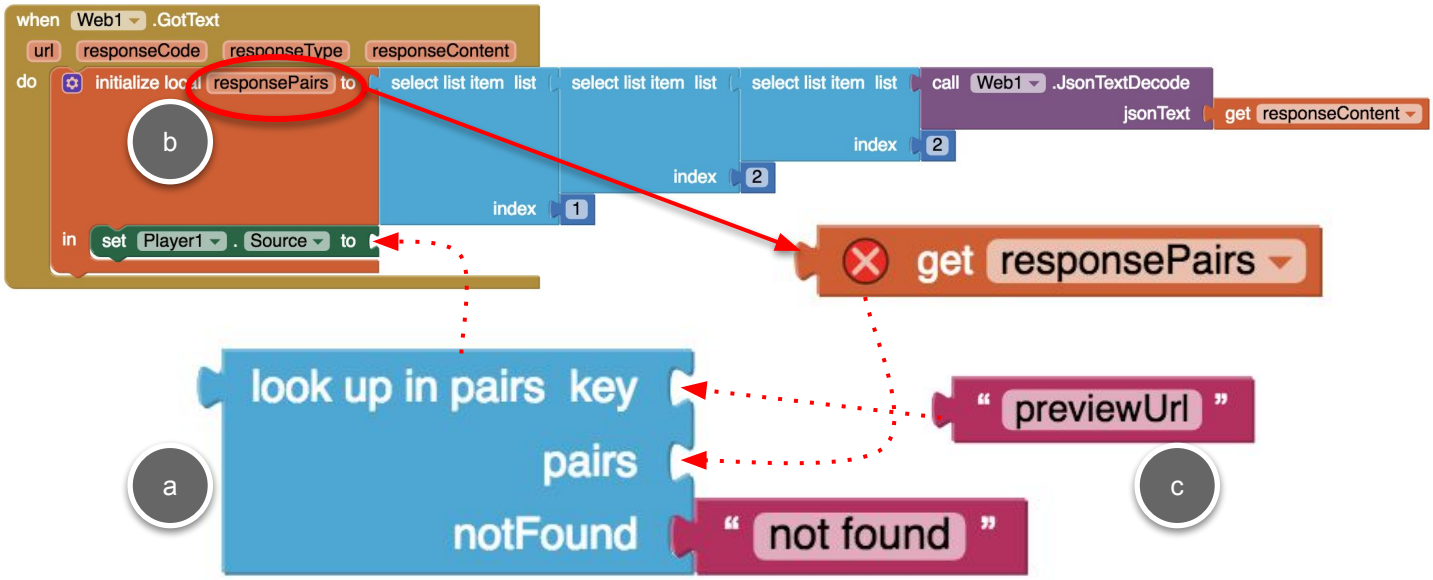
28 The Player component plays music files. Drag out a **set Player1.Source** block so we can set it based on the response.



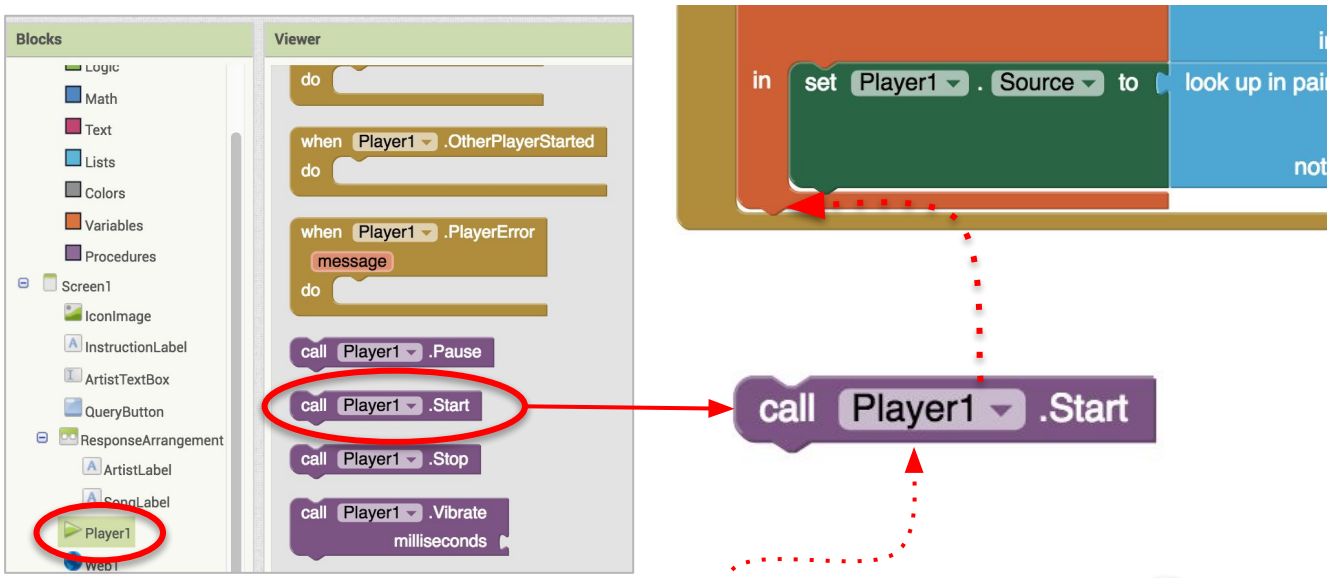
ITUNES MUSIC APP

GET RESPONSE

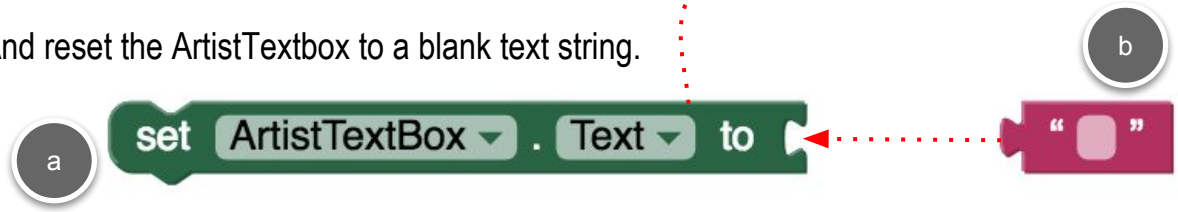
29 Drag out a **look up in pairs** block from the Lists palette and snap it in. The key we're looking for is "previewUrl". We'll extract this from our **responsePairs** variable.



30 Drag out a **Player1.Start** block and snap it in below.



31 And reset the ArtistTextbox to a blank text string.



ITUNES MUSIC APP

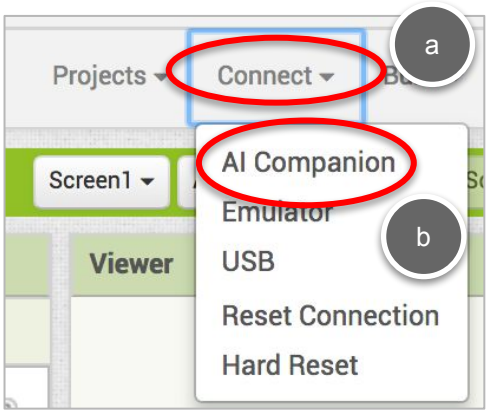
GET RESPONSE

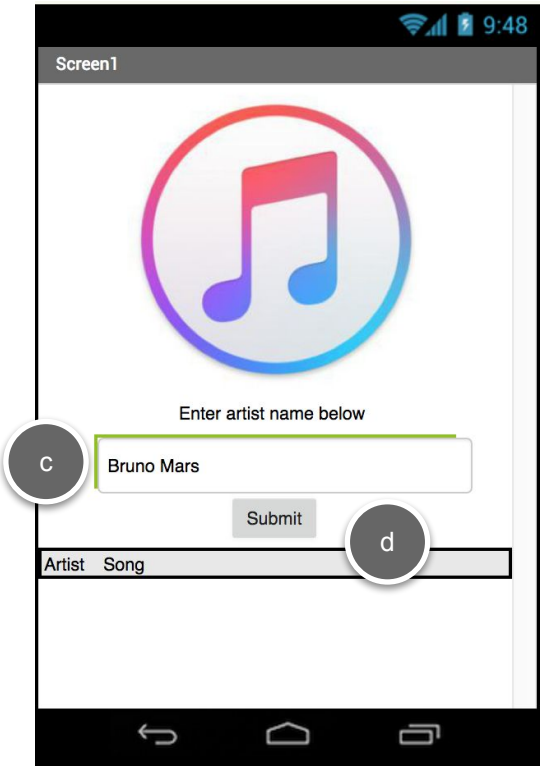
29 Here is a challenge for you. Can you add code to set **ArtistLabel.Text** and **SongLabel.Text** by using the **look up in pairs** block?

For the keys, you will use the highlighted names shown below.


```
{
 "resultCount":1,
 "results": [
{
"wrapperType":"track",
"kind":"song",
"artistId":136975,
"collectionId":401186200,
"trackId":401187150,
"artistName":"The Beatles",
"collectionName":"Abbey Road",
"trackName":"Here Comes the Sun",
"collectionCensoredName":"Abbey Road",
"trackCensoredName":"Here Comes the Sun",
```


30 Test your app out using the MIT AI2 Companion! Type in the name of your favorite musical artist, submit, and listen!



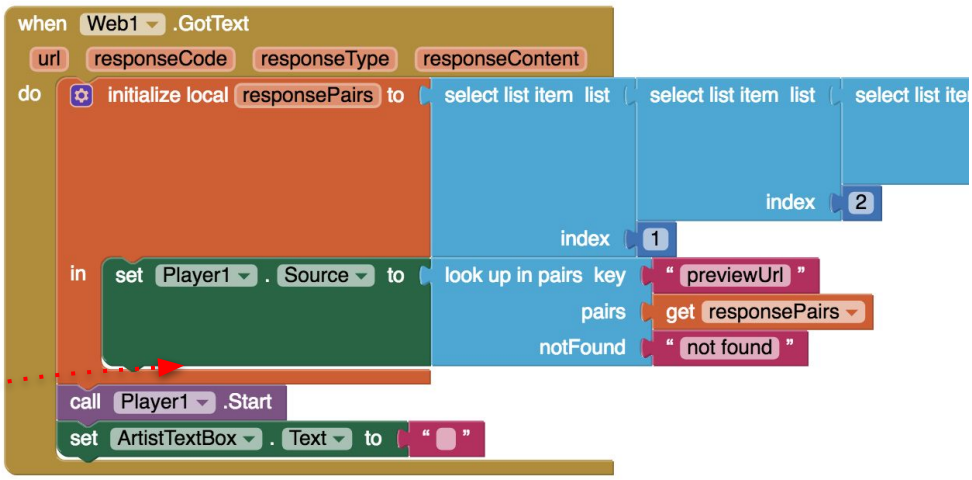


ITUNES MUSIC APP

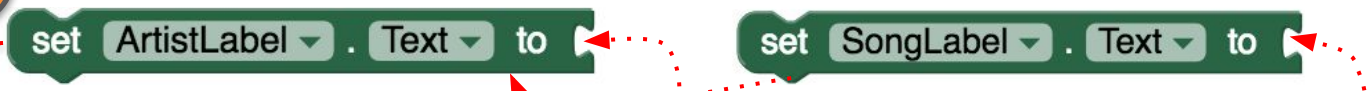
ARTISTLABEL AND SONGLABEL

In case you need a little help with the challenge on the previous page.

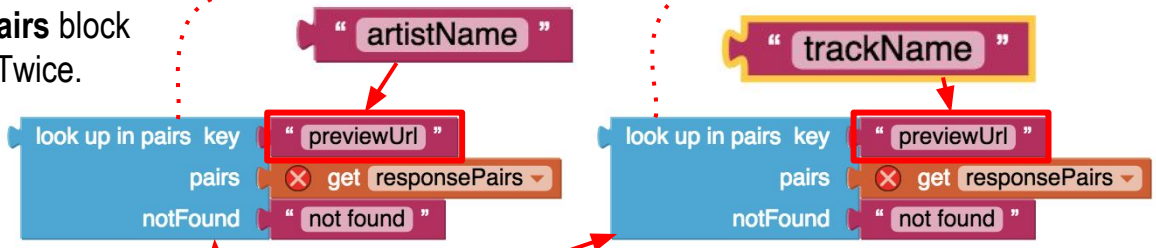
You need to follow the same format as the **set Player1.Source** block.

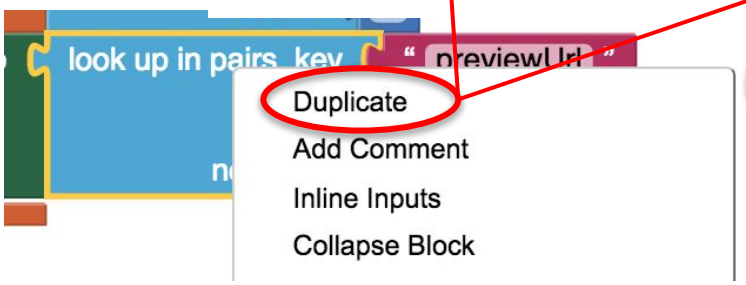


1 Drag out blocks to set the Text for **ArtistLabel** and **SongLabel**.



2 Duplicate the **look up in pairs** block from above. Twice.



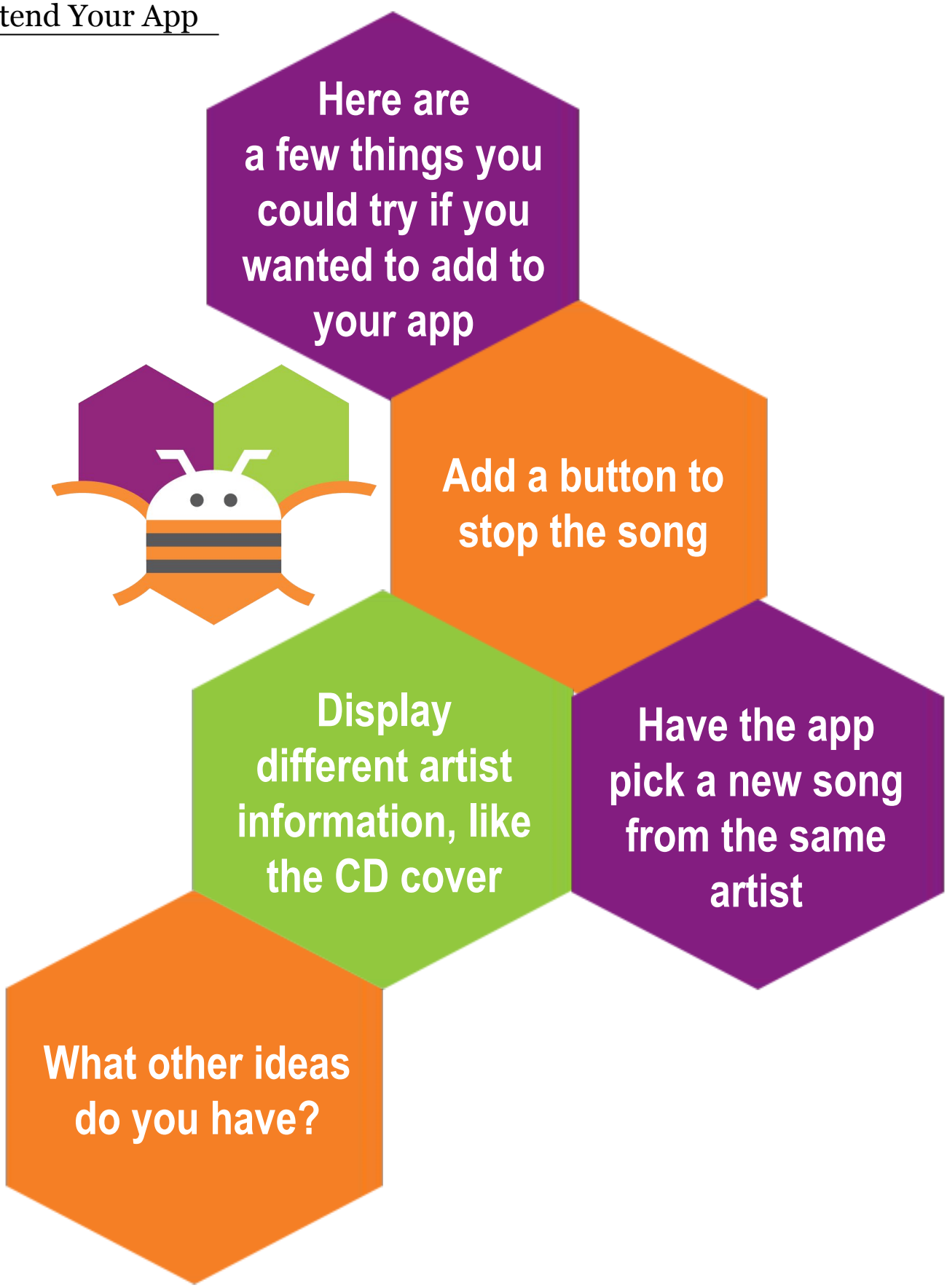


3 Change "previewUrl" to "artistName" for ArtistLabel, and to "trackName" for SongLabel. Get the spelling exactly as shown.

4 Snap in both below **set Player1.Source.**

ITUNES MUSIC APP

Extend Your App



ITUNES MUSIC APP

Extend Your App

Here are a couple of things you could try if you wanted to add to your app

Add a button to stop the song

Have the app pick a new song from the same artist