How to Build Your Own Data Base using Blocks - Part 1

7/6/2014

Motivation

App Inventor 2 does not provide access to any kind of local SQL database. It offers TinyDB, a key/data local store.

(A Spreadsheet component is available for access to Google Sheets.)

In this paper, we will work our way up to relational data base concepts by building on the list facilities of App Inventor. We start with some familiarity with simple lists, and work our way up to tables.

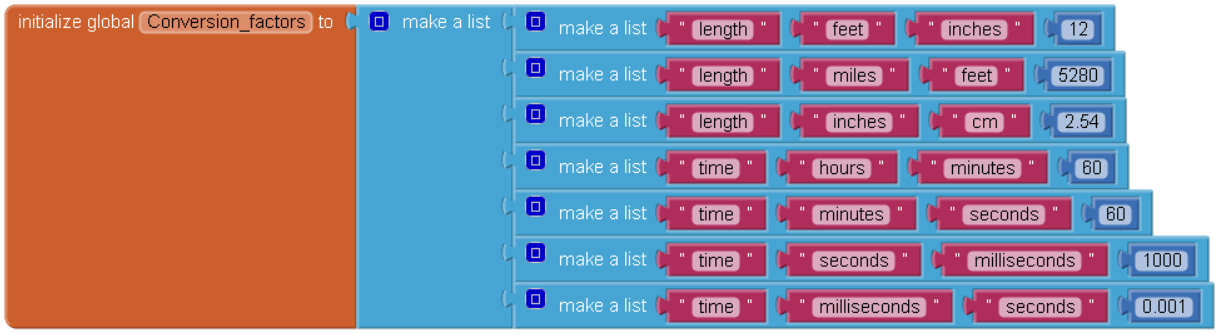
Prerequisite reading

http://www.appinventor.org/bookChapters/chapter19.pdf
(especially the end of the chapter)

http://www.appinventor.org/bookChapters/chapter21.pdf

(also the end of the chapter)

Some Sample Data



Here's part of a conversion table, done as a list of lists, also known as a table.

(Normally, you would read a table like this in from a .csv file. This is only an example.)

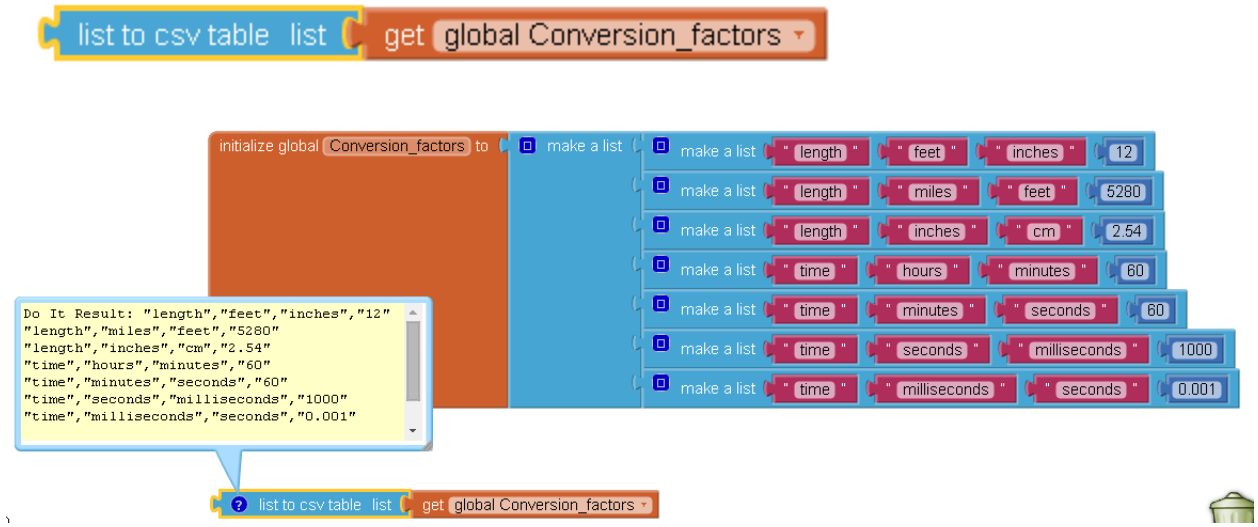
To read this, we go row by row. The first row says that to work with lengths, we can convert feet to inches by multiplying by 12. Likewise, the second row says that to work with lengths, we can convert miles to feet by multiplying by 5280.

I threw in a few time conversions also following the same pattern. Each column serves a single purpose:

1. The Dimension: type of units being converted (length, time, weight, etc.)
2. The measure being converted From
3. The measure being converted To
4. What Factor we must multiply by to do the conversion.

I've done this as a screenshot to show the structure and how you can nest lists inside lists. I've used the right-click "Inline Inputs" and "External Inputs" Block Editor options to get this rows and columns effect. You can also type data into a spreadsheet program and export the data as a CSV table:

I'm going to cheat and use the App Inventor emulator, right-click-DoIt, and one of the table blocks to extract the data in CSV table format, in case you want to copy my data...



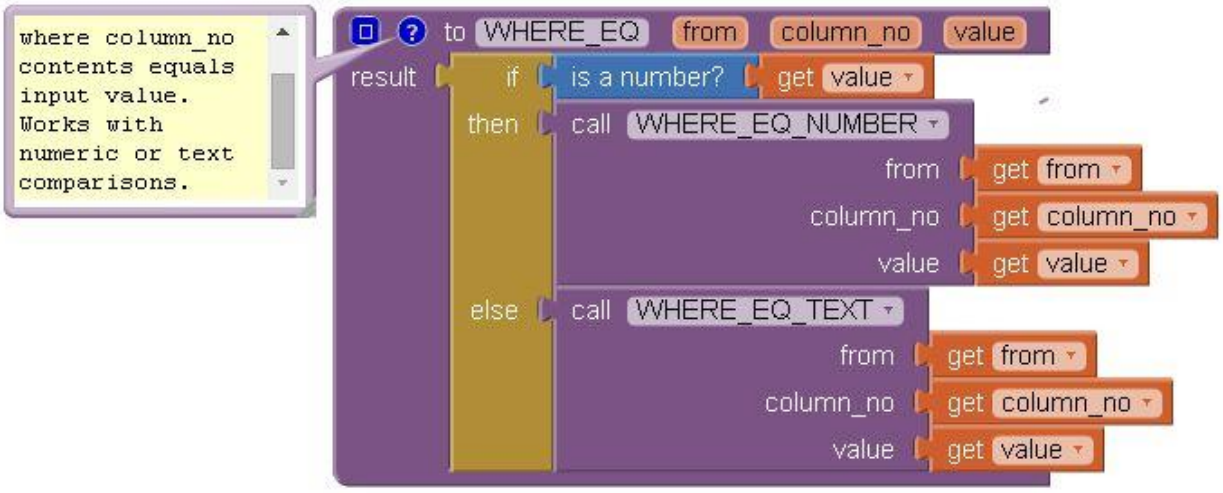
"length","feet","inches","12"
"length","miles","feet","5280"
"length","inches","cm","2.54"
"time","hours","minutes","60"
"time","minutes","seconds","60"
"time","seconds","milliseconds","1000"
"time","milliseconds","seconds","0.001"

Slicing and Dicing

Data Analysts use a cooking analogy for the act of chopping up a table by row and by column. One way to slice up a table is to pick only those rows that have a particular value in one column. In SQL this is specified with a WHERE clause, so let's name our procedures using the word WHERE…

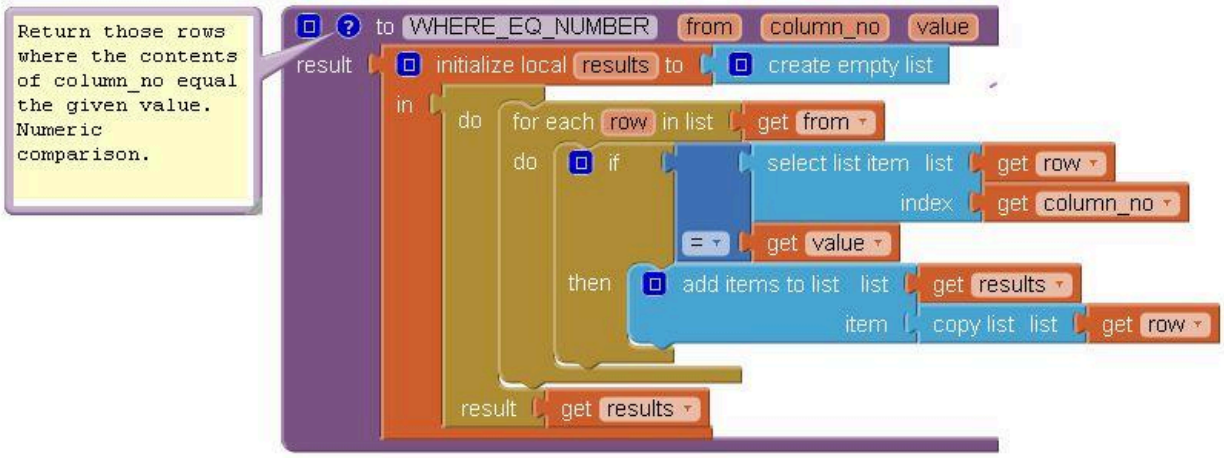
WHERE_EQ

This is a result procedure that accepts a table (a list of lists) and returns a smaller table containing the same columns, but only the rows where a particular column (specified by column number) equals a particular value.

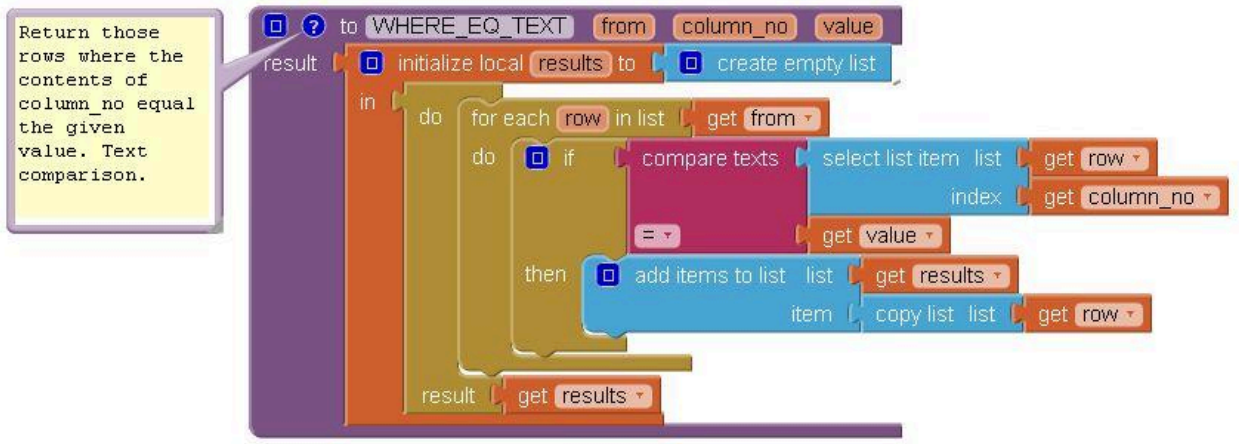


Because App Inventor has two different kinds of comparison blocks, one for numbers and another for text, we decide early on which comparison we will use rather than deciding on a row by row basis. People usually store either numbers or text in any one table column, and never mix them.

WHERE_EQ_NUMBER



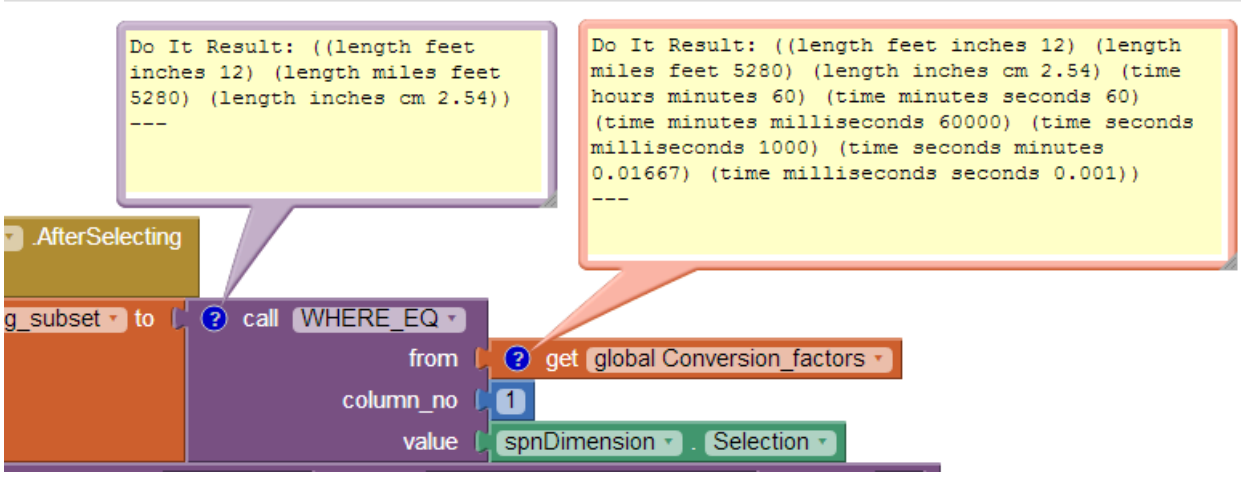
WHERE_EQ_TEXT



Notice that these two return procedures differ only in the type of block used for the comparison in the inner if-then block.

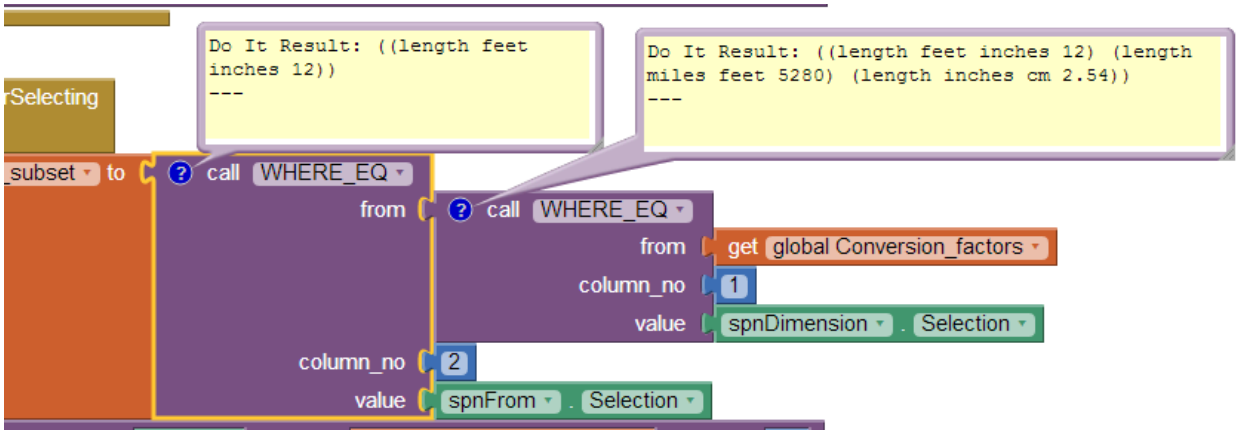
For convenience, we only need to remember to use the WHERE_EQ procedure …

Here's a sample application of WHERE_EQ, using a spinner selection of "length" for column 1 of our data table, to give us just the rows for length conversions …



Because WHERE_EQ is a return procedure it can be combined with other return procedure calls. If we want to return just the rows from our sample data with a Dimension (column 1) value of "length" and a From value of "feet" we nest the calls …

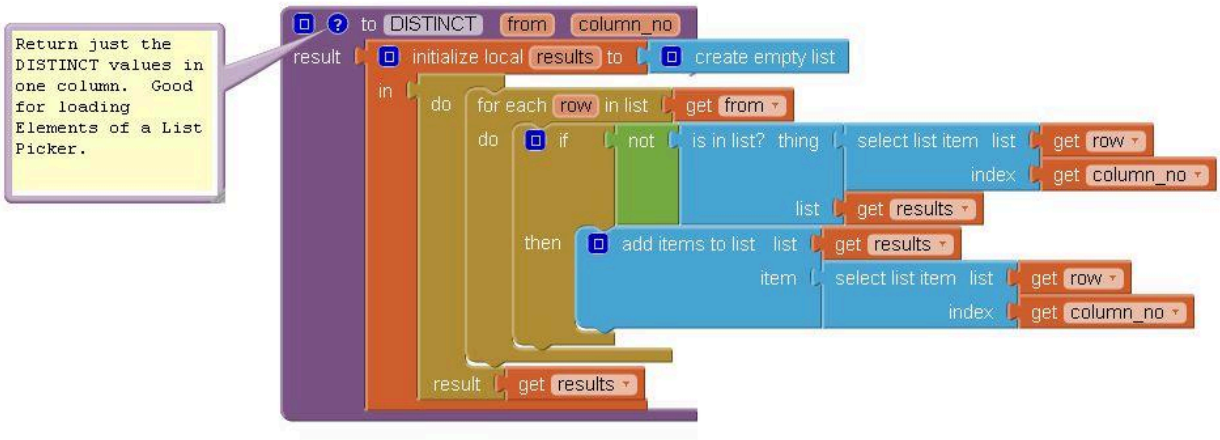
Nesting and piping example



If we want to test for other conditions such as “less than”, “greater than”, etc., we can duplicate our WHERE_EQ procedures, and change the names and test criteria, such as WHERE_LT, WHERE_GT, etc.

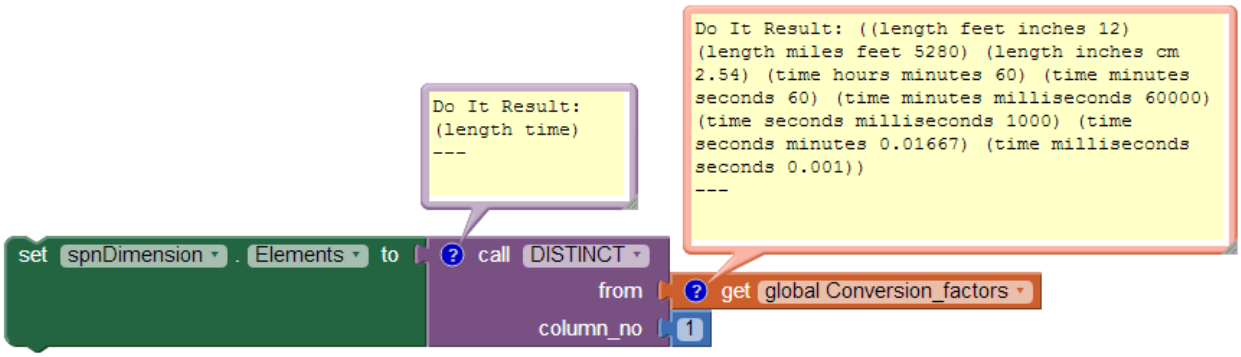
DISTINCT

If you want to filter your table data by particular column values, you need a procedure that will return a list of distinct values from one particular table column. This can be used to prime the Elements list of a List Picker, List View, or a Spinner.



This builds and returns a result list by starting with an empty list, running through the input table, and checking each value in the requested column number against the result list before adding it.

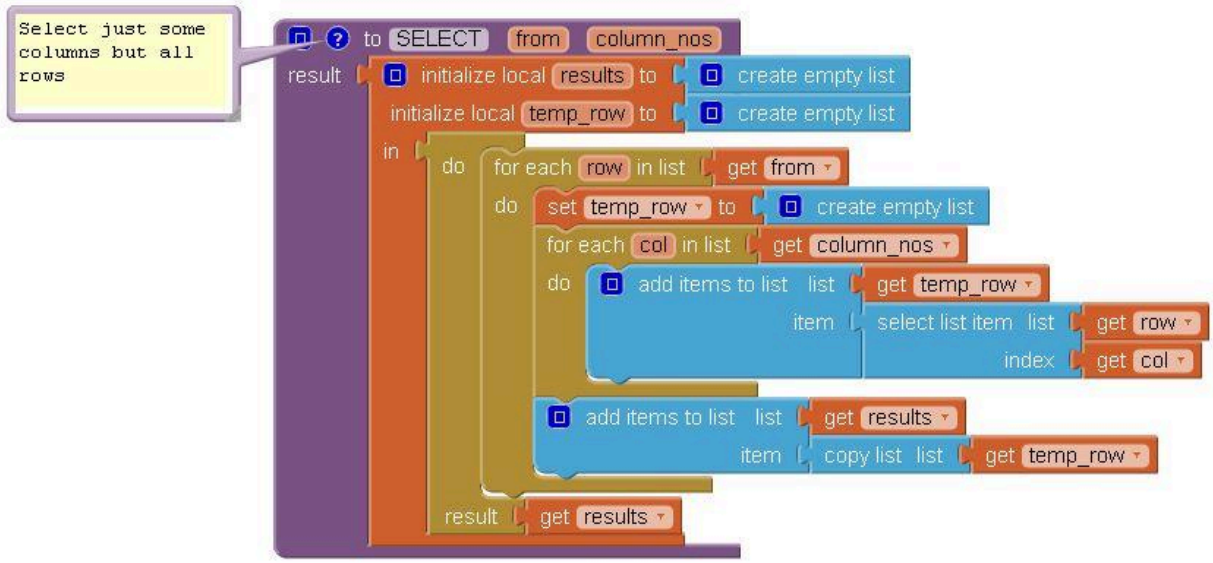
This is an example of the expression you would use to feed the Elements of a Spinner or List Picker to decide which dimension (length or time) to use to filter our conversion table...



SELECT

This is for when you want just some of the columns in a table, but all the rows…

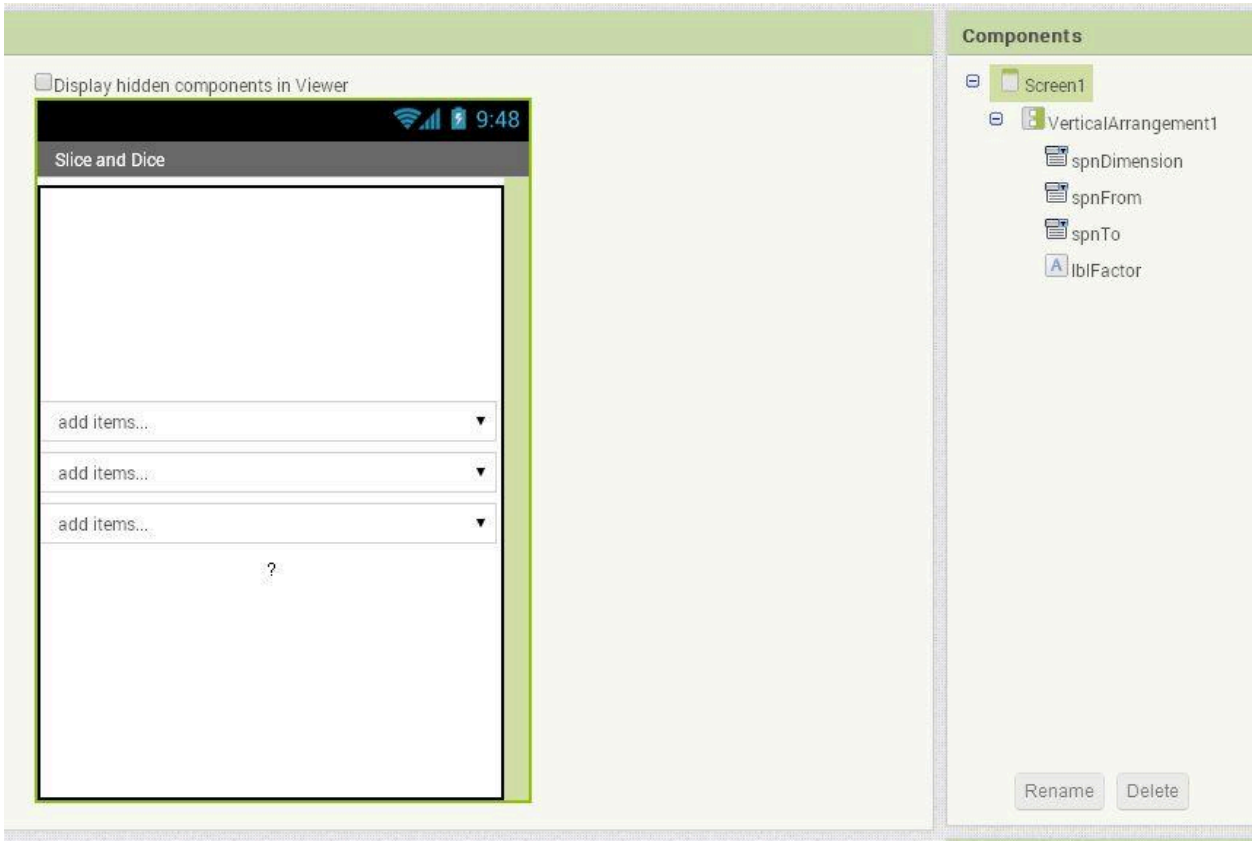
from = a table (list of lists) input
column_nos = a list of column numbers in the input table we want in the result
result = a table with all **from**'s rows, but just the requested column numbers, slid down.



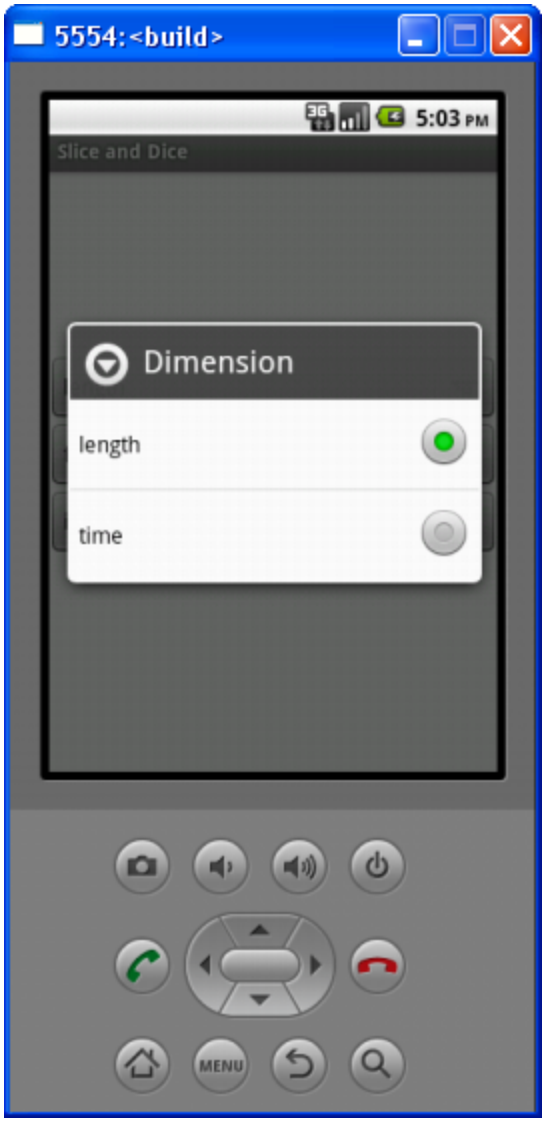
A sample app - conversions

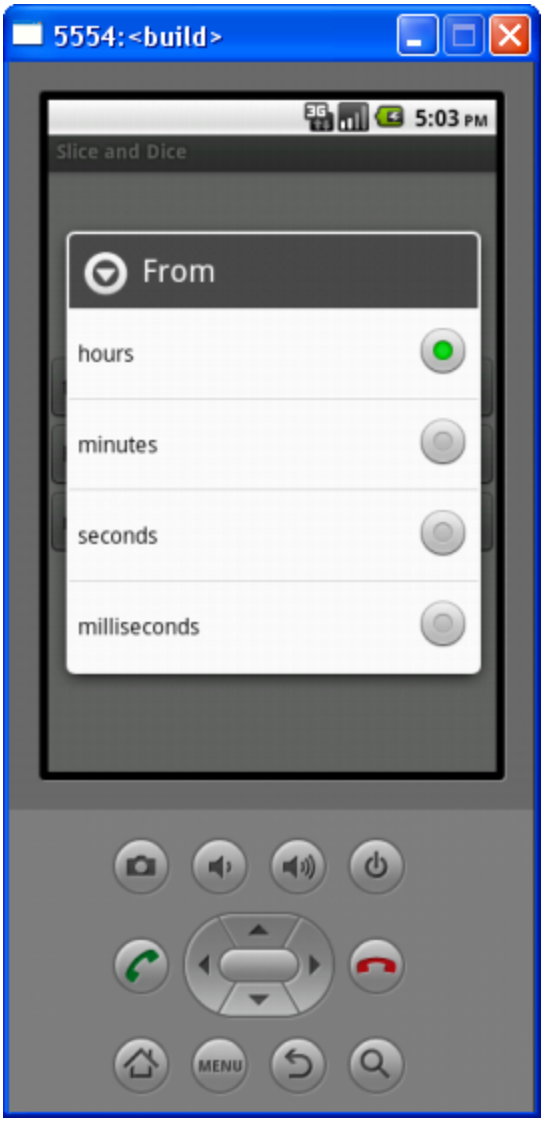
[AI2 gallery link]

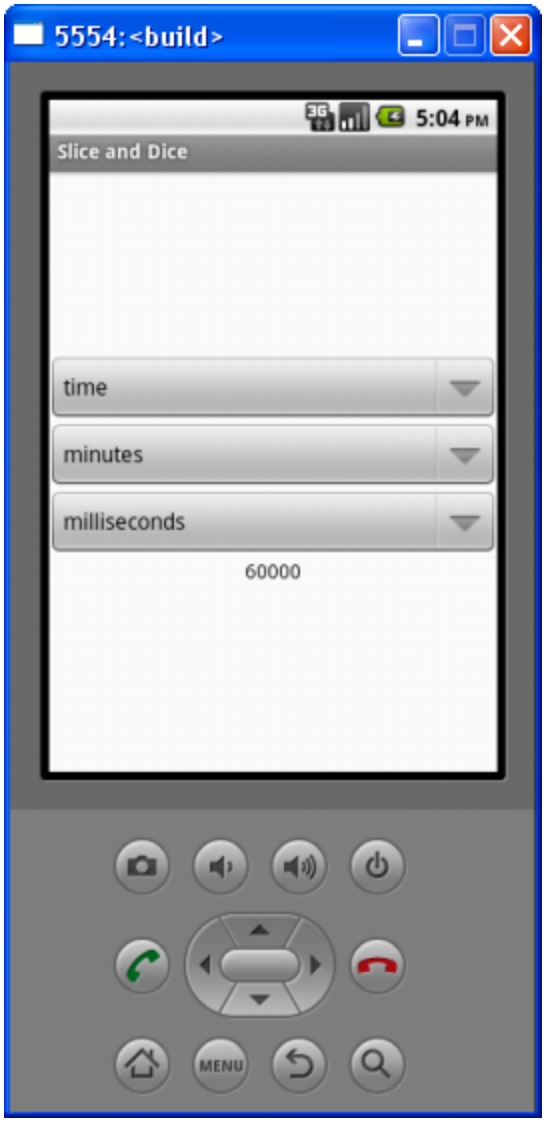
Screen shots - Designer




5554:<build>
4:58 PM
Slice and Dice
length
feet
inches
12
MENU


5554:<build>
5:03 PM
Slice and Dice
Dimension
length
time
MENU


5554:<build>
5:03 PM
Slice and Dice
From
hours
minutes
seconds
milliseconds
MENU



More projects

https://docs.google.com/document/d/1acg2M5KdunKjJgM3Rxpy_Rf6vT6OozxdIWglgbmzroA/edit?usp=sharing