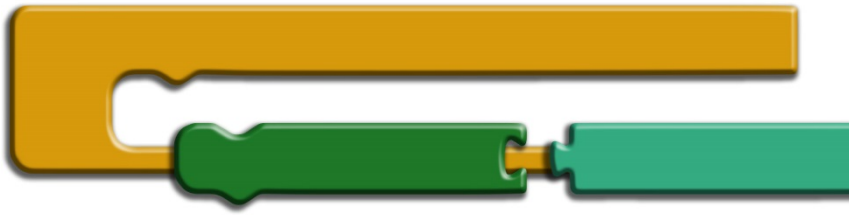
starting out with >>> APP INVENTOR FOR ANDROID

CHAPTER 5

Repetition Blocks, Times, and Dates



TONY GADDIS · REBECCA HALSEY

Topics

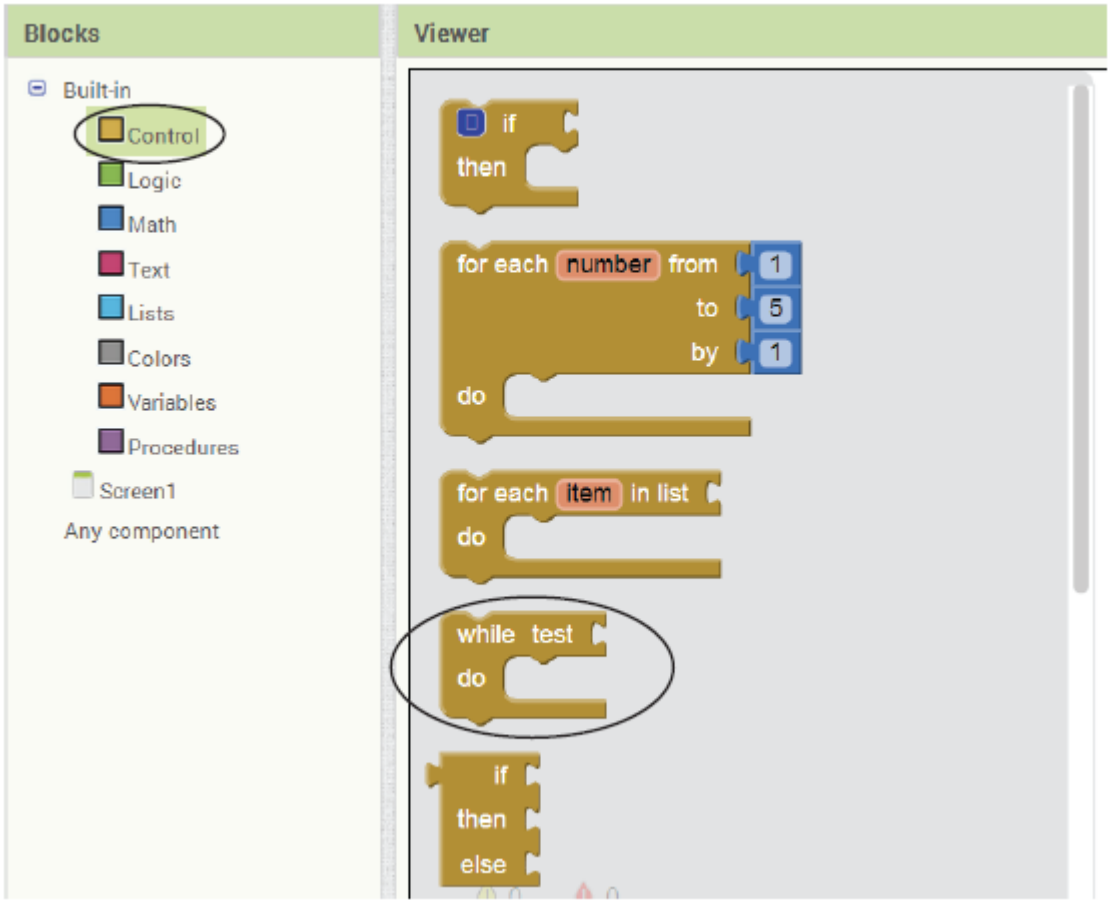
- The *Notifier* Component
- The *while* Loop
- The *for each* Loop
- The *Clock* Component
- The *DatePicker* Component

The `while` Loop

- The *`while`* loop causes a statement or set of statements to repeat as long as a Boolean expression is *`true`*.
- The while loop has two parts
  1. A Boolean expression that is tested for a true or false value.
  2. A statement or set of statements that is repeated as long as the Boolean expression is *`true`*.

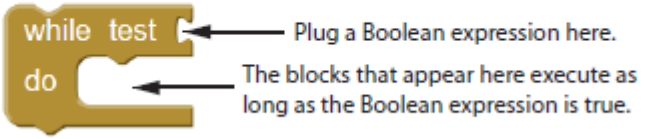
The while Loop

Figure 5-11 The `While` Loop Block (*Source:* MIT App Inventor 2)



The `while` Loop

Figure 5-12 The `While` Loop (*Source:* MIT App Inventor 2)

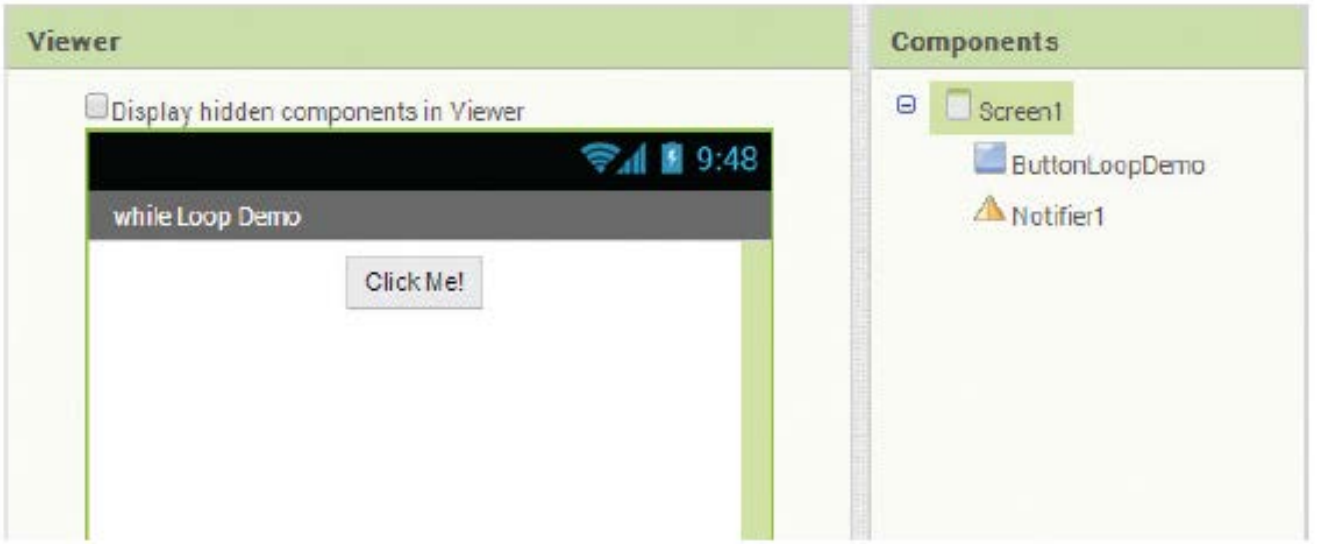


The *`while`* loop has two sockets: *`test`* and *`do`*.

- Test Socket
  - Holds a Boolean expression.
    - If TRUE the blocks in the *do* socket are executed.
    - If FALSE the loop ends.
  - Each time the loop executes the blocks in its *do* socket, the loop is *iterating* or performing an *iteration.*

The while Loop

Figure 5-13 The WhileLoopDemo Project in the Designer (*Source:* MIT App Inventor 2)



The `while` Loop

Figure 5-13 shows the WhileLoopDemo.

- Notice the project has
  - A Button named *`ButtonLoopDemo`*.
  - A *`Notifier`* named *`Notifier1`*.
- When the user clicks the button in the emulator or actual device, the message dialog is shown in Figure 5-14.
- When the user clicks OK to close the dialog, another identical dialog is displayed.
- This repeats until the dialog is displayed 5 times.

The while Loop

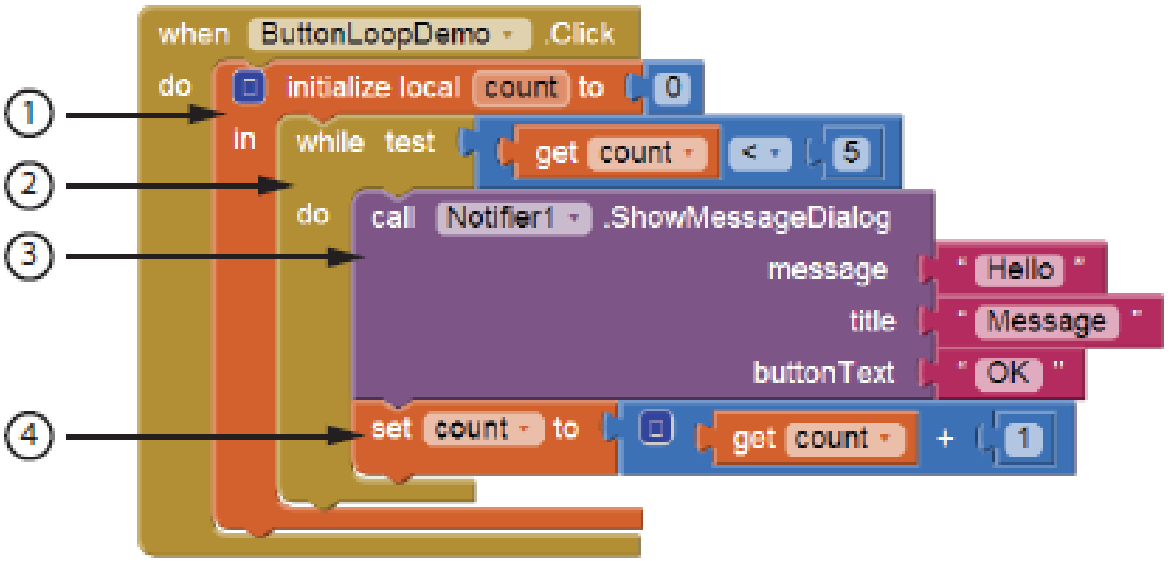
Figure 5-14 Message Dialog is Displayed Five Times (*Source:* MIT App Inventor 2)

The `while` Loop

- Figure 5-15 shows the project's workspace in the Blocks Editor.
- This initializes the local variable named *`count`* to the value 0. Each time the user clicks the *`ButtonLoopDemo`* button, the *`count`* variable will keep count of the number of times the message dialog is displayed.
- The *`while`* loop uses the Boolean expression *`count < 5`*. The loop will repeat as long as the value of the count variable is less than 5.
- This block displays a message dialog.
- This block adds 1 to the *`count`* variable.

The while Loop

Figure 5-15 The Project's Workspace in the Blocks Editor (*Source:* MIT App Inventor 2)



The `while` Loop

The *`while`* Loop is a Pretest Loop

- The *`while`* loop tests it condition *before* performing an iteration.
- The test is done at the beginning of the loop.
- You usually have to perform some steps before the loop to ensure it executes at least one time.

The `while` Loop

The *`while`* Loop is a Pretest Loop

- In Figure 5-15 the first action to take place in the *`ButtonLoopDemo.Click`* is that the count variable is set to 0.
- If *`count`* was set to a value greater than 5 the loop would not have executed.
- A *`while`* loop will never iterate if its Boolean expression is *`false`* to start with.

The `while` Loop

Counter Variables

In the WhileLoopDemo app

- The *`count`* variable is set to the value 0.
- One is added to the *`count`* variable during each loop iteration.
- The loop executes as long as count is less than 5.
- The *`count`* variable keeps track of the number of iterations the loop has performed.

The `while` Loop

Infinite Loops

- Loops must contain a way to terminate otherwise the loop will continue to repeat until the program is interrupted.
- Infinite loops usually occur when the programmer forgets to write code inside the loop making the Boolean expression false.

The `while` Loop

Figure 5-22 An Infinite `While` Loop (*Source:* MIT App Inventor 2)