Flow Control: Repetition with While Loops
(Alice In Action, Ch 4)

18 September 2012

Slides Credit: Joel Adams, Alice in Action

Flow of Control

Sequential Execution
Each instruction is executed in order they are written (after the previous one, before the next one).

Functions
Enable procedural decomposition.
Repeat statements by calling functions multiple times.
Flow of Control

Selection
Some statements are executed while others are not.

Repetition
Statements can be repeated some fixed number of time, or else can be repeated until some event signals they should not be repeated any more.

Flow Control

- Flow: sequence of steps for performing a user story
- Flow control statement: structure for managing flow
- Flow control statements used in previous chapters
  - `doInOrder`: produces a sequential execution
  - `doTogether`: produces a parallel execution
  - methods and functions: name a block of statements

![Diagram of doInOrder and doTogether statements](image)
Flow Control

- Control statements introduced in the current chapter
  - **if**: directs program flow along one of two paths
  - **for**: directs flow into a fixed number of loops
  - **while**: directs flow into an arbitrary number of loops

Objectives

- Review using the **if** statement to perform some statements while skipping others
- Use the **for** and **while** statements to perform (other) statements more than once
- Lean about design patterns.
if Statement

• Structure of an if statement:
  – if (Condition) {
    Statements$_1$
  } else {
    Statements$_2$
  }

• Value of a condition determines direction of flow
  – If Condition is true, Statements$_1$ are selected
  – If Condition is false, Statements$_2$ are selected

if Statement Mechanics (continued)

if statement behavior is also called selective flow or selection.
Introducing Repetition

- **Selection:**
  - The `if` statement executes its body 0 or 1 times based on the condition.

- **Indefinite Loop**
  - The `while` loop executes its body 0 or more times based on the condition.

- **Definite loop**
  - The `for` loop executes its body a fixed number of times.

---

**while** Statement Mechanics

- **Structure of a while loop**
  - `while ( Condition ) {
      Statements
    }

- **The while loop is more general than the for loop**
  - Flow enters while structure if `Condition` is true
  - One statement must eventually falsify `Condition`. Otherwise, you get an infinite loop that will never ends

---

FIGURE 4-30  Flow through a while statement
Design Patterns for Indefinite Loops

Several common patterns:
• Counter-controlled Loop
• Interactive Loop
• Sentinel-controlled Loop

Also... accumulator design pattern

Counter-Controlled Pattern

Counter-Controlled Loop
Evaluates a counter in loop's logical expression:

Pseudo-code for this pattern:
initialize counter
while counter > critical-value
    do controlled block statements
    decrement (increment) counter
Interactive Loop Pattern

Interactive Loop
Prompts user whether or not to continue.

Pseudo-code for this pattern:
set moredata to "yes"
while moredata is "yes"
  get next data item
  process data item
  ask user if there is moredata

Sentinel Loop Pattern

Sentinel Loop
Checks input for a special value to determine whether or not to continue.

eg. Use -1 while entering quiz grades means STOP.

get first data item
while data item is not the sentinel
  process data item
  get next data item
Accumulator Pattern

Accumulator Pattern
Used to combine many values into one value.
Eg. Sum several values in a loop. Can be used with any loop pattern.

initialize accumulator variable(s)
loop to repeat each item:
    combine new data item into accumulator

Introducing the **while** Statement

• Strategy for building third shot of dragon animation
  – Repeatedly have dragon flap its wings
  – Move dragon downward while it is above drawbridge
• Overview for building the third shot
  – Place `doTogether` statement in `doInOrder` statement
  – Use `setPointOfView()` to move camera closer
  – Send `flappingWings()` message to the `dragon`
  – Drag the `while` statement to the editing area
  – Drop the `while` statement below `doInOrder`
  – Insert placeholder value into the `while` condition
Introducing the while Statement (continued)

- Building the third shot (continued)
  - Drag dragon’s isAbove() over condition and drop
  - Add castle.bridge argument to isAbove()
  - Insert doTogether statement in the while loop
  - Add move() method to cause dragon to descend
  - Send another flapWings() message to dragon
  - Use setPointOfView() to zoom in for a close-up

- Infinite loop occurs if loop lacks falsifying condition
  - In third shot, move() eventually terminates loop
Introducing the **while** Statement (continued)

![Diagram of the while Statement](image)

**FIGURE 4-28** The `playScene1Shot1()` method (final version)

Introducing the **while** Statement (continued)

![Image of the dragon on the drawbridge](image)

**FIGURE 4-29** The dragon on the drawbridge
Indefinite Loop Problems

**Infinite Loop**
A loop with no end – no way out!
Generally, this is a very bad problem.

Example:
an argument with… your parents?

Have you ever noticed?
Indefinite Loop Problems

Busy Loop
An infinite loop, which never pauses (e.g. for input), and thus consumes all of the computer's resources.
This is a terrible, awful, no-good, very bad problem.

Example:
a counting loop with a faulty condition.

Comparing the if and while Statements

- Both test conditions before flow enters structure
- Both are bypassed if initial condition is false

- while statement is repeats after finishing its body.
  - 0 or more times.
- if statement’s body can only be executed once.
  - 0 or 1 times.
- if statement can have an else body.
A While-Loop Example

• Setting up the scene
  – Use `shebuilder` to create a soccer player (Jane)
  – Place `soccerBall` object in Jane’s hands
• Writing a `dropBounce()` method for `soccerBall`
  – Move the ball down `distanceToGround` meters
  – Change `distanceToGround` by bounce factor (2/3)
  – Move ball up reduced `distanceToGround` meters
  – Bouncing continues while `distanceToGround > 0`
• Writing a `janeDropsBall()` method
  – Send `roll()` messages to forearms
  – Send `dropAndBounce()` message to `soccerBall`

A While-Loop Example (continued)

![Sketch of the up-down motion of a bouncing ball](image)

FIGURE 4-31 Sketch of the up-down motion of a bouncing ball
A While-Loop Example (continued)

FIGURE 4-32 Jane with the soccer ball

A While-Loop Example (continued)

FIGURE 4-33 Method soccerBall.dropAndBounce()
Summary

- Flow control statement: controls the flow of statement execution
- if statement: directs flow along one of two paths based on evaluation of a condition
- for statement: repeats execution of a group of statements a fixed number of times
- while statement: repeats execution of a group of statements an appropriate number of times based on a loop continuation condition
- Design Patterns for writing programs.
  - Loops: Interactive, Counter, Sentinel
  - Accumulator Pattern

Student To Do’s

- Readings:
  - Alice in Action, Chapter 4 (This week)

- HW as posted on course schedule
  - Due Monday during lab