

Goal and Task Analysis

Basic Loop Structures in Ruby

Ruby is a popular, versatile, and easy-to-learn scripting language available on many computing platforms. Most programming languages rely on loops (blocks of code that execute a determined number of times, usually based on a conditional value) and Ruby is no exception. Most logic operations in programs are based on some kind of loop—whether it tests if a value is true, continues until something changes, or applies a process or calculation to each item in a list of many values. This individualized instruction module will teach learners both how to construct valid loops and when to apply each type as appropriate.

Audience

The tutorial will be targeted toward learners of the Ruby language. Previous modules will have instructed the user in the basics and syntax of the Ruby language, allowing this module to focus on the loop structure. Further modules could be developed to cover additional aspects of the language (Functions, etc.). The tutorial will be developed in Adobe Captivate.

Goal: *The learner will be able to implement an appropriate loop structure in Ruby.*

Task Analysis

1. **Definition of a loop:** “A block of code that executes a defined number of times based on a condition.”
 - a. Select examples and non-examples from listed options.
2. **Understand a ‘for’ loop:** A block of code that executes once for each item in a sequence. An example of this is a program that reads words from a file, capitalizes each one, and displays them.
 - a. Which situation would be handled correctly by a ‘for’ loop?
 - b. ‘Put the following lines in the correct order’
3. **Understand a ‘while’ loop:** A block of code that executes over and over until a condition changes. An example of this is a program which displays, “No, wrong number!” until the user guesses the correct number.
 - a. Which situation would be handled correctly by a ‘while’ loop?
 - b. ‘Put the following lines in the correct order’
4. **Understand an ‘if’ loop:** A block of code that executes only when a condition is true. An example of this is a program that tells the user whether a number entered is even or odd.
 - a. Which situation would be handled correctly by an ‘if’ loop?
 - b. ‘Put the following lines in the correct order’
5. **Determine which kind of loop is suitable for various situations**
 - a. A random number game
 - b. A program that counts to 100, then stops
 - c. A program that runs until you enter the ‘exit’ command
 - d. A program that returns the square of a list of numbers
 - e. A program which displays the first 10 members of the Fibonacci sequence

Educational Outcomes

Task	Instructional Outcome	Activity	Type
1	The learner will indicate knowledge the correct definition of a loop.	The learner will select examples and non-examples of loop architecture	Apply Concept
2a	The learner will correctly identify situations in which a 'for' loop is appropriate.	The learner will select examples for which a 'for' loop is appropriate from a list of options.	Apply Concept
2b	The learner will correctly order lines of a 'for' loop.	The learner will drag lines of code into the correct sequence for a given prompt.	Remember Principle
3a	The learner will correctly identify situations in which a 'while' loop is appropriate.	The learner will select examples for which a 'while' loop is appropriate from a list of options.	Apply Concept
3b	The learner will correctly order lines of a 'while' loop.	The learner will drag lines of code into the correct sequence for a given prompt.	Remember Principle
4a	The learner will correctly identify situations in which a 'while' loop is appropriate.	The learner will select examples for which a 'while' loop is appropriate from a list of options.	Apply Concept
4b	The learner will correctly order lines of a 'while' loop.	The learner will drag lines of code into the correct sequence for a given prompt.	Remember Principle
5	Given a situation, the learner will correctly identify which kind of loop is appropriate.	The learner will be presented with a hypothetical situation, and will select from a list the appropriate type of loop to use.	Apply Principle