**Workshop Handout: Managing Arrays in C#**

**Overview**

In this workshop, we will explore how to use arrays in C#. We will create a simple console application that captures and displays pet names using a static array.

**Key Concepts**

1. **Arrays in C#**:
   * Arrays are a collection of variables of the same type stored in contiguous memory locations.
   * They provide a way to store multiple values in a single data structure, which can be easily accessed using an index.
2. **Static Arrays**:
   * Static arrays have a fixed size defined at the time of declaration.
   * This type of array is suitable for scenarios where the number of elements is known beforehand.
3. **User Input and Loops**:
   * We'll use loops to get user input and populate the array.
   * A **for** loop will allow us to prompt the user for each pet name.
   * A **foreach** loop will enable us to iterate over and display the elements of the array.

**Program Structure**

**1. Array Declaration**

* Declare a static array to hold pet names.
* The array size is fixed, accommodating a predefined number of pet names.

**2. Main Method Steps**

* **Prompt for Number of Pet Names**:
  + Ask the user how many pet names they want to add to the array.
* **Populate the Array**:
  + Use a **for** loop to prompt the user to enter each pet name.
  + Store each name in the corresponding array index.
* **Display Pet Names**:
  + Use a **foreach** loop to iterate through the array and display each pet name.

**Detailed Steps**

1. **Declare the Array**:
   * Create a string array with a fixed size to store pet names.
2. **Get User Input for Array Size**:
   * Prompt the user to specify how many pet names they want to enter.
3. **Enter Pet Names**:
   * Use a loop to prompt the user for each pet name.
   * Store the names in the array at the appropriate index.
4. **Display Stored Pet Names**:
   * Use a loop to iterate through the array.
   * Display each name stored in the array.

**Execution Flow**

1. **Prompt for Number of Names**:
   * The program begins by asking the user how many pet names they want to add.
2. **Input Pet Names**:
   * The user is prompted to enter each pet name one by one.
3. **Display Pet Names**:
   * After all names are entered, the program displays each pet name.

**Important Notes**

* **Array Size Limitation**:
  + Static arrays have a fixed size, so ensure the user does not enter more names than the array can hold.
  + Error handling for array bounds is crucial but not covered in this basic example.
* **Loops for Efficiency**:
  + Using loops simplifies the process of getting user input and displaying array elements.
  + The **for** loop is ideal for inputting a known number of elements, while the **foreach** loop is convenient for displaying all elements.
* **User Experience**:
  + Ensure the user is aware of how many names they can enter.
  + Clear prompts and instructions enhance usability.

**Conclusion**

By the end of this workshop, you should understand how to use arrays in C# to store and manage multiple values efficiently. Arrays provide a structured way to handle collections of data, and understanding their use is fundamental to programming in C#. This exercise demonstrates how to declare, populate, and iterate through a static array, focusing on practical applications of user input and loops.