# C# Array Types and Examples

## Single-Dimensional Array

A single-dimensional array is a list of elements accessible by a single index.  
  
Example:  
```csharp  
int[] numbers = new int[5]; // Declares an array of 5 integers  
numbers[0] = 1; // Assigning values  
numbers[1] = 2;  
numbers[2] = 3;  
numbers[3] = 4;  
numbers[4] = 5;  
  
// Accessing elements  
for (int i = 0; i < numbers.Length; i++)  
{  
 Console.WriteLine(numbers[i]);  
}  
```

## Multi-Dimensional Array

A multi-dimensional array can be thought of as a table with rows and columns.  
  
Example:  
```csharp  
int[,] matrix = new int[3, 4]; // 3 rows and 4 columns  
  
// Assigning values  
for (int i = 0; i < 3; i++)  
{  
 for (int j = 0; j < 4; j++)  
 {  
 matrix[i, j] = i + j;  
 }  
}  
  
// Accessing elements  
for (int i = 0; i < 3; i++)  
{  
 for (int j = 0; j < 4; j++)  
 {  
 Console.Write(matrix[i, j] + " ");  
 }  
 Console.WriteLine();  
}  
```

## Jagged Array (Array of Arrays)

A jagged array is an array whose elements are arrays. The dimensions and sizes of the child arrays can vary.  
  
Example:  
```csharp  
int[][] jaggedArray = new int[3][]; // An array of three arrays  
  
jaggedArray[0] = new int[4] { 1, 2, 3, 4 };  
jaggedArray[1] = new int[5] { 5, 6, 7, 8, 9 };  
jaggedArray[2] = new int[3] { 10, 11, 12 };  
  
// Accessing elements  
for (int i = 0; i < jaggedArray.Length; i++)  
{  
 for (int j = 0; j < jaggedArray[i].Length; j++)  
 {  
 Console.Write(jaggedArray[i][j] + " ");  
 }  
 Console.WriteLine();  
}  
```

## Implicitly Typed Arrays

You can create an array without explicitly defining the data type of its elements using the `var` keyword, provided all elements are of the same type.  
  
Example:  
```csharp  
var cities = new[] { "New York", "London", "Mumbai" };  
  
// Accessing elements  
foreach (var city in cities)  
{  
 Console.WriteLine(city);  
}  
```

## Arrays as Object Initializers

Arrays can be initialized using object initializer syntax without specifying the size.  
  
Example:  
```csharp  
string[] names = { "Alice", "Bob", "Charlie" };  
  
// Accessing elements  
foreach (var name in names)  
{  
 Console.WriteLine(name);  
}  
```