Multidimensional Arrays

This lesson will give you insights about how to declare, access and use multidimensional arrays in C#.

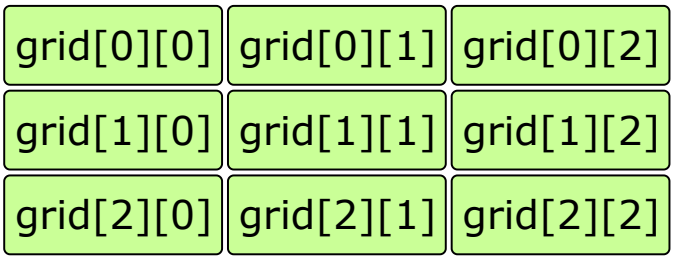
A *multidimensional* array allows **nesting** arrays.

Arrays can have more than one dimension. The following example creates a two-dimensional array of **3** rows and **3** columns:

```
int[,] arr = new int[3, 3];
```

This allocates 3*3 elements in **one** memory block.

Here is a visual representation of the *multi-dimensional* array `grid[3][3]`:



Representation of grid[i][j]

Here, every element is distinguished and accessed by `grid[i][j]`, where **grid** is the name of the array and `i` and `j` are the subscripts with which we can access each element in this 2-D array.

Similarly, an array of **three** dimensions can be represented as:

```
int[,,] arr = new int[3, 3, 3];
```

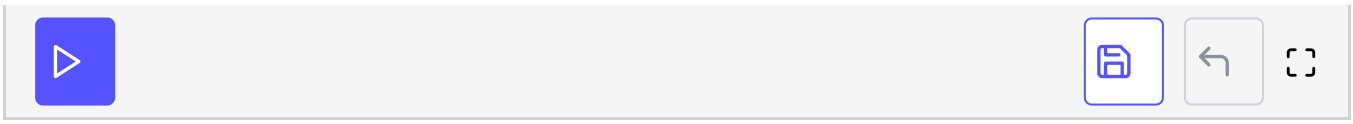
You can also initialize the array upon declaration:

You can also initialize the array upon declaration.

```
using System;

public class MainClass {

   public static void Main(String [] args)
   {
        int[,] arr = new int[4, 2] { {1, 1}, {2, 2}, {3, 3}, {4, 4} };
        // Access a member of the multi-dimensional array:
        Console.Out.WriteLine(arr[3, 1]); // 4
   }
}
```



Interesting, right? Now let us learn about jagged arrays which are actually arrays of arrays!