

Multidimensional Packed Bit Array

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2007-12/msg03217.html>

- *From:* Thomas Bruckner <tbrstore_at_gmail_dot_com>
 - *Date:* Sat, 29 Dec 2007 10:20:36 +0100
-

Hello,

I've looked online for an implementation of a multidimensional packed bit array (like BitArray but with more than one dimension), and could not find any, so I'm trying to create my own. However with limited success so far. I'm having troubles with calculating the required size of the storage and also with the bit shifts... my brain is turning in circles trying to figure this out ;-)

Anyone want to help with this?

Thomas Bruckner

```
using System;
using System.Collections.Generic;
using System.Text;

namespace ConsoleApplication1
{
    class Program
    {
        static void Main(string[] args)
        {
            // new array 4 dimensions each with 2, 2, 4 and 4 elements
            MultiBitArray mba = new MultiBitArray(new int[] { 2, 2, 4, 4 });

            for (int i = 0; i <= 3; i++)
            {
                for (int j = 0; j <= 3; j++)
                {
                    for (int k = 0; k < 15; k++)
                    {
                        for (int l = 0; l < 15; l++)
                        {
                            int[] coord = new int[] { i, j, k, l };

                            // make sure it is initially false

                            System.Diagnostics.Debug.Assert(mba.GetValue(coord) == false);
                        }
                    }
                }
            }
        }
    }
}
```

Multidimensional Packed Bit Array

```
// set to true, and check
mba.SetValue(coord, true);

System.Diagnostics.Debug.Assert(mba.GetValue(coord) == true);

Console.WriteLine(
    mba.GetLocation(coord).ToString());
}
}
}
}

Console.ReadKey();
}
}

public class MultiBitArray
{
    /// <summary>
    /// Number of dimensions.
    /// </summary>
    private int dimCount;

    /// <summary>
    /// bite size of each dimension (may be different)
    /// </summary>
    private int[] bitSize;

    /// <summary>
    /// storage for bits
    /// </summary>
    internal byte[] storage;

    public MultiBitArray(int[] bitSize)
    {
        this.dimCount = bitSize.Length;
        this.bitSize = bitSize;

        AllocStorage();
    }

    /// <summary>
    /// Allocates storage space for bits.
    /// </summary>
    private void AllocStorage()
    {
        long storageSize = 1;

        for (int i = 0; i < this.dimCount; i++)
        {
```

Multidimensional Packed Bit Array

```
storageSize *= (this.bitSize[i] * dimCount);
}

// i can't even figure out how to determine the
// amount of memory... so I set it very large for testing
//storageSize *= 4;

this.storage = new byte[storageSize];
}

/// <summary>
/// Gets the location of a bit according to dimension indexes
/// </summary>
internal int GetLocation(int[] coord)
{
    int location = 0;
    int bitcount = 0;

    for (int i = 0; i < dimCount; i++)
    {
        location = (location << bitcount) | coord[i];
        bitcount = this.bitSize[i];
    }

    return location;
}

/// <summary>
/// Read a bit.
/// </summary>
public bool GetValue(int[] coord)
{
    int location = GetLocation(coord);
    return (this.storage[location / 8] & (byte)(1 << (location % 8))) != 0;
}

/// <summary>
/// Set a bit.
/// </summary>
public void SetValue(int[] coord, bool value)
{
    int location = GetLocation(coord);
    this.storage[location / 8] |= (byte)(1 << (location % 8));
}
}
.
```