

A two-dimensional fail-soft array

```
/*
C#: The Complete Reference
by Herbert Schildt

Publisher: Osborne/McGraw-Hill (March 8, 2002)
ISBN: 0072134852
*/

// A two-dimensional fail-soft array.

using System;

class FailSoftArray2D {
int[,] a; // reference to underlying 2D array
int rows, cols; // dimensions
public int Length; // Length is public

public bool errflag; // indicates outcome of last operation

// Construct array given its dimensions.
public FailSoftArray2D(int r, int c) {
rows = r;
cols = c;
a = new int[rows, cols];
Length = rows * cols;
}

// This is the indexer for FailSoftArray2D.
public int this[int index1, int index2] {
// This is the get accessor.
get {
if(ok(index1, index2)) {
errflag = false;
return a[index1, index2];
}
}
}
}
```

```

} else {
errflag = true;
return 0;
}
}

// This is the set accessor.
set {
if(ok(index1, index2)) {
a[index1, index2] = value;
errflag = false;
}
else errflag = true;
}
}

// Return true if indexes are within bounds.
private bool ok(int index1, int index2) {
if(index1 >= 0 & index1 < rows & index2 >= 0 & index2 < cols)
return true; return false; } } // Demonstrate a 2D indexer.
public class TwoDIndexerDemo { public static void Main() {
FailSoftArray2D fs = new FailSoftArray2D(3, 5); int x; // show
quiet failures Console.WriteLine("Fail quietly."); for(int
i=0; i < 6; i++) fs[i, i] = i*10; for(int i=0; i < 6; i++) { x
= fs[i,i]; if(x != -1) Console.Write(x + " "); }
Console.WriteLine(); // now, generate failures
Console.WriteLine(" Fail with error reports."); for(int i=0; i
< 6; i++) { fs[i,i] = i*10; if(fs.errflag)
Console.WriteLine("fs[" + i + ", " + i + "] out-of-bounds"); }
for(int i=0; i < 6; i++) { x = fs[i,i]; if(!fs.errflag)
Console.Write(x + " "); else Console.WriteLine("fs[" + i + ",
" + i + "] out-of-bounds"); } } } [/csharp]

```