multidimensional vector, how to

jdstufu (43) Mar 18, 2008 at 3:15pm

Im trying to make use of a vector container to hold my matrix. But I dont know how to configure the vector container to be multidimensional.

How does one configure the vector container to hold a multidimensional array data (matrix).

Thanks :)

ropez (310) Mar 18, 2008 at 7:33pm

```
vector< vector<data> > vec;
```

Note that in the current standard, you must have spaces between the last two > >.

jdstufu (43) Mar 20, 2008 at 5:25pm

vector< vector<data> > vec;

what is data? variable type? STL Container?

jdstufu (43) Mar 20, 2008 at 5:29pm

Sorry, Im so dumb for not understanding it right away, is a vector of vector of data where data can be any class or STL container or any native data type such as int, double, etc....

Im I correct?

jdstufu (43) Mar 20, 2008 at 5:33pm

But, how does one address element 2:3 of a 4x4 dimension vector?

ropez (310) Mar 20, 2008 at 9:13pm

You are correct (you're not dumb!). Address it like this:

```
1     // Create
2     vector< vector<int> > vec(4, vector<int>(4));
3     // Write
4     vec[2][3] = 10;
5     // Read
6     int a = vec[2][3];
```

jdstufu (43) Mar 20, 2008 at 9:46pm

Thank you sir!

Now I understand.

jdstufu (43) Mar 21, 2008 at 2:00am

Sir, how about if you are going to dynamically add an element into your multidimensional vector?

If my vector is one dimensional I do it this way:

```
1 vector< int> mnumber;
2
3 for(unsigned i=0; i<10; i++)
4      number.push_back(i+1);
5
6 number.clear();
7
8 for(unsigned i=0; i<30; i++)
9      number.push_back(i+1);
```

but if its multidimensional? How do you do it?

ropez (310) Mar 21, 2008 at 9:14am

Remember that a two dimentional vector is simply a vector of vectors. Each "subvector" or "row" is completely independant from the others, meaning that they can have different size. You can dynamically add elements to change the size of each row, and add elements to the main vector to add more rows.

You can simply use nested loops to build the structure. This is one way to do it:

```
vector< vector<int> > vec;

for (int i = 0; i < 10; i++) {
    vector<int> row; // Create an empty row
    for (int j = 0; j < 20; j++) {
        row.push_back(i * j); // Add an element (column) to the row
    }
    vec.push_back(row); // Add the row to the main vector
}
```

It's not the only way, you may also add empty rows in one loop, and then use another loop that adds columns by inserting an element to all rows:

```
vector< vector<int> > vec;

for (int i = 0; i < 10; i++) {
    vec.push_back(vector<int>()); // Add an empty row
}

for (int j = 0; j < 20; j++) {
    for (int i = 0; i < vec.size(); i++) {
        vec[i].push_back(i * j); // Add column to all rows
    }

    // You could also use iterators:
    for (vector< vector<int> >::iterator it = vec.begin(); it != vec.end(); ++it) {
        it->push_back(j); // Add column to all rows
    }
}
```

Last edited on Mar 21, 2008 at 9:19am

ropez (310) Mar 21, 2008 at 9:25am

There is also another possible way to use a vector as a two dimentional array. If each row will always have the same fixed length, you can use a single vector, and simulate two dimentions by calculating the index from a row and column number:

```
const int rows = 10;
const int columns = 20;

vector<int> vec;
vec.resize(rows * columns);
```

```
 6
 7 for (int row = 0; row < rows; row++) {
 8     for (int col = 0; col < columns; col++) {
 9         vec[row * columns + col] = row * col;
10     }
11 }
```

Last edited on Mar 21, 2008 at 9:26am

Ganon11 (54) Mar 21, 2008 at 12:51pm

The last idea is great if you're not going to be changing the matrix a lot, but if you're not going to be changing it, why bother using vectors at all?

Taking the 'simulate-a-2D-array-with-one-dimension' approach runs into trouble when you want to add a new column. Suddenly there's a lot of value-shifting that have to take place. Adding rows isn't bad at all. You could never use .push_back - it would always have to be a .resize() and then adding all the elements, etc. etc.

But, heck, if you just want to build this matrix and then leave the size alone, that's a great idea.

jdstufu (43) Mar 22, 2008 at 3:35am

Thank you Sir Ropez.

Indeed there are a lot of ways on how to do it. But I prefer the 1st technique. Its a lot easier to understand. Thank you once again.

Topic archived. No new replies allowed.