

Namespaces

Antonio Espín Herranz

Namespaces

- Un espacio con nombre, como indica su denominación, es una zona separada donde se pueden declarar y definir objetos, funciones y en general, cualquier identificador de tipo, clase, estructura, etc, al que se asigna un nombre o identificador propio.
- Si el programa no indicamos ningún namespace, las declaraciones se ubican en lo que se denomina el *espacio global*.
- La utilidad los espacios con nombre es que nos ayudan a evitar problemas con identificadores en grandes proyectos o cuando se usan bibliotecas externas.
- Nos permite, por ejemplo, que existan objetos o funciones con el mismo nombre, declarados en diferentes ficheros fuente, siempre y cuando se declaren en distintos espacios con nombre.

Sintaxis

```
namespace [<identificador>] {  
    ... <declaraciones y definiciones> ...  
}
```

```
// Fichero de cabecera "puntos.h"
```

```
namespace espacio_2D {  
    struct Punto {  
        int x; int y;  
    };  
}
```

```
namespace espacio_3D {  
    struct Punto {  
        int x;  
        int y;  
        int z;  
    };  
}
```

```
// Fin de fichero
```

using

```
using namespace <identificador>;  
#include "puntos.h"
```

```
// Activar el espacio con nombre espacio_2D  
using namespace espacio_2D;
```

```
// Define la variable p1 de tipo espacio_2D::Punto  
Punto p1;
```

```
// Define la variable p2 de tipo espacio_3D::Punto ...  
espacio_3D::Punto p2;
```

alias

- Definición de alias:
 - **namespace** <alias_de_espacio> = <nombre_de_espacio>;
- Ejemplo:
 - **namespace** nombre_namespace
 - { ... declaraciones ... } ...
- **// Alias ...**
 - **namespace** nn= nombre_namespace;

Ejemplo

```
#include <iostream>
using namespace std;

namespace MyNamespace1 {
    int i;
    namespace MyNamespace2 {
        int j;
    }
}

int main()
{
    MyNamespace1::i = 19;
    MyNamespace1::MyNamespace2::j = 10;

    cout << MyNamespace1::i << " " << MyNamespace1::MyNamespace2::j << "\n";

    // use MyNamespace1
    using namespace MyNamespace1;
    cout << i * MyNamespace2::j;
    return 0;
}
```

Declaraciones varios ficheros

- Las declaraciones de los namespace no tienen porque hacerse dentro del mismo fichero.
- Podemos tener varios ficheros con clases que pertenezcan al mismo namespace.
- En el punto H, las declaraciones de las clases se especifican dentro del namespace, como se ha especificado antes.

```
namespace ns1 {  
    // Declaraciones  
    class Clase1 { ... }  
}
```

- En el fichero CPP hay que indicar el namespace:

- Por ejemplo, un constructor de la clase:

```
ns1::Clase1::Clase1(){ ... }
```