GeeksforGeeks

A computer science portal for geeks

COURSES

Login

HIRE WITH US 🔍

Namespace in C++ | Set 1 (Introduction)

Consider following C++ program.

```cpp
// A program to demonstrate need of namespace
int main()
{
    int value;
    value = 0;
    double value; // Error here
    value = 0.0;
}
```

Output :

```
Compiler Error:
 'value' has a previous declaration as 'int value'
```

In each scope, a name can only represent one entity. So, there cannot be two variables with the same name in the same scope. Using namespaces, we can create two variables or member functions having the same name.

```cpp
// Here we can see that more than one variables
// are being used without reporting any error.
// That is because they are declared in the
// different namespaces and scopes.
#include <iostream>
using namespace std;

// Variable created inside namespace
namespace first
{
    int val = 500;
}

// Global variable
int val = 100;

int main()
{
    // Local variable
    int val = 200;

    // These variables can be accessed from
    // outside the namespace using the scope
    // operator ::
    cout << first::val << '\n';

    return 0;
}
```

Output:

```
500
```

Definition and Creation:

Namespaces allow us to group named entities that otherwise would have *global scope* into narrower scopes, giving them *namespace scope*. This allows organizing the elements of programs into different logical scopes referred to by names.

- Namespace is a feature added in C++ and not present in C.
- A namespace is a declarative region that provides a scope to the identifiers (names of the types, function, variables etc) inside it.
- Multiple namespace blocks with the same name are allowed. All declarations within those blocks are declared in the named scope.

A namespace definition begins with the keyword **namespace** followed by the name-space name as follows:

```
namespace namespace_name
{
    int x, y; // code declarations where
              // x and y are declared in
              // namespace_name's scope
}
```

- Namespace declarations appear only at global scope.
- Namespace declarations can be nested within another namespace.
- Namespace declarations don't have access specifiers. (Public or private)
- No need to give semicolon after the closing brace of definition of namespace.
- We can split the definition of namespace over several units.

```cpp
// Creating namespaces
#include <iostream>
using namespace std;
namespace ns1
{
    int value()     { return 5; }
}
namespace ns2
{
    const double x = 100;
    double value() {  return 2*x; }
}

int main()
{
    // Access value function within ns1
    cout << ns1::value() << '\n';

    // Access value function within ns2
    cout << ns2::value() << '\n';

    // Access variable x directly
    cout << ns2::x << '\n';

    return 0;
}
```

Output:

```
5
200
100
```

Classes and Namespace:

Following is a simple way to create classes in a name space

```cpp
// A C++ program to demonstrate use of class
// in a namespace
#include <iostream>
using namespace std;

namespace ns
{
    // A Class in a namespace
    class geek
    {
    public:
        void display()
        {
            cout << "ns::geek::display()\n";
        }
    };
}

int main()
{
    // Creating Object of geek Class
    ns::geek obj;

    obj.display();

    return 0;
}
```

Output:

```
ns::geek::display()
```

Class can also be declared inside namespace and defined outside namespace using following syntax

```cpp
// A C++ program to demonstrate use of class
// in a namespace
#include <iostream>
using namespace std;

namespace ns
{
    // Only declaring class here
    class geek;
}

// Defining class outside
class ns::geek
{
public:
    void display()
    {
        cout << "ns::geek::display()\n";
    }
};

int main()
{
    //Creating Object of geek Class
    ns::geek obj;
    obj.display();
    return 0;
}
```

Output:

```
ns::geek::display()
```

We can **define methods also outside the namespace**. Following is an example code.

```cpp
// A C++ code to demonstrate that we can define
// methods outside namespace.
#include <iostream>
using namespace std;

// Creating a namespace
namespace ns
{
    void display();
    class geek
    {
    public:
        void display();
    };
}

// Defining methods of namespace
void ns::geek::display()
{
    cout << "ns::geek::display()\n";
}
void ns::display()
{
    cout << "ns::display()\n";
}

// Driver code
int main()
{
    ns::geek obj;
    ns::display();
    obj.display();
    return 0;
}
```

Output:

```
ns::display()
ns::geek::display()
```

namespace in C++ | Set 2 (Extending namespace and Unnamed namespace)

Namespace in C++ | Set 3 (Accessing, creating header, nesting and aliasing)

Can namespaces be nested in C++?

Reference:

http://www.cplusplus.com/doc/tutorial/namespaces/

This article is contributed by **Abhinav Tiwari**. If you like GeeksforGeeks and would

like to contribute, you can also write an article and mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above

Recommended Posts:

namespace in C++ | Set 2 (Extending namespace and Unnamed namespace)

Cons of using the whole namespace in C++

Why "using namespace std" is considered bad practice

Difference between namespace and class

Namespace in C++ | Set 3 (Accessing, creating header, nesting and aliasing)

C Language Introduction

Introduction to Iterators in C++

Introduction to C++ Programming Language

Hygienic Macros : An Introduction

OpenMP | Introduction with Installation Guide

Understanding "volatile" qualifier in C | Set 1 (Introduction)

Pointers in C and C++ | Set 1 (Introduction, Arithmetic and Array)

Forward List in C++ | Set 1 (Introduction and Important Functions)

Virtual Functions and Runtime Polymorphism in C++ | Set 1 (Introduction)

GDB (Step by Step Introduction)

Improved By : MadhuriAgawane

Article Tags : C   C++   cpp-namespaces

Practice Tags : C   CPP

30

☐ To-do ☐ Done

Feedback/ Suggest Improvement   Add Notes   Improve Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

GeeksforGeeks
A computer science portal for geeks

5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305
feedback@geeksforgeeks.org

COMPANY

About Us
Careers
Privacy Policy
Contact Us

LEARN

Algorithms
Data Structures
Languages
CS Subjects
Video Tutorials

PRACTICE

Courses
Company-wise
Topic-wise
How to begin?

CONTRIBUTE

Write an Article
Write Interview Experience
Internships
Videos

@geeksforgeeks, Some rights reserved