namespace

Consider following C++ program.

```
// A program to demonstrate need of namespace
int main()
{
   int a=100;
   double a=8.8; // Error here
}
```

In the above program when we try to use same variable names an error will take place called as redefinition of variable ‘a’ in other words **name conflict or name collision** will occur. for avoiding these types of errors **namespace** is used.

Defining a Namespace:

A namespace definition begins with the keyword namespace followed by the namespace name as follows

```
namespace namespace_name
{
  // code declarations
}
```

To access either function or variable which belong to a particular namespace

Syntax:

```
nameofthenamespace::code;   // code could be variable or function.
```

EXAMPLE-1:

```
#include <iostream>
using namespace std;
// first name space
namespace one
 {
int a=100;
}
// second name space
namespace two
{
  double a=9.345;
}
```

```
int main () {
   // accessing variable from first name space.
   cout<<one::a;
   // accessing variable from second name space.
   cout<<two::a;
   return 0;
}
```

EXAMPLE-2:

```
#include <iostream>
using namespace std;
// first name space
namespace one {
   void Display() {
      cout <<"Inside one" << endl;
   }
}
// second name space
namespace two {
   void Display() {
      cout << "Inside two" << endl;
   }
}
int main () {
   // Calls function from first name space.
   one::Display();
   // Calls function from second name space.
   two::Display();
   return 0;
}
```

using directive

one can also avoid adding of namespaces with the **using namespace** directive. This directive tells the compiler that the subsequent code is making use of names in the specified namespace.

syntax:

```
using namespace nameofthenamespace;
```

EX:

```
#include <iostream>
using namespace std;
// first name space
namespace one {
  void Display() {
    cout <<"Inside one" << endl;
  }
}
// second name space
namespace two {
  void Display() {
    cout << "Inside two" << endl;
  }
}
using namespace one;
int main () {
  // Calls function from first name space.
  Display();
  // Calls function from second name space.
  two::Display();
  return 0;
}
```