static keyword is used to declares members and methods that are not bound to class instances.

Static members

Static members of a class are not associated with objects of the class. These are independent objects that are stored elsewhere, once, and these are accessible within class functions. We need to define the static member outside of class (see below). The static member is shared between all instances of the class. Meaning if its value is changed in one instance, the other instances will see it as well.

class A
{
    static int x; // declaration - use 'static'
};

int A::x = 0; // definition - doesn't use 'static'

The following piece of code where static member is used to count the total number of instances.

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

class X
{
    public:
        static int how_many;

        X() {
            ++how_many;
        }
};

int X::how_many = 0;

int main()
{
    X instance1;
    X instance2;
    X instance3;

    cout << "X has " << X::how_many << " instances." << endl;
```
When we run this program, the output is:

X has 3 instances

Static methods

Static functions of the class are not associated with class instances. Consequently, this pointer is not available within static methods. Static methods can only directly access static members and other static methods of the class.

In the following code, void an_example_static_function() is a static method of class X.

```cpp
using namespace std;

class X {
public:
    int x;
    static int how_many;

    X() {
        ++how_many;
    }

    static void an_example_static_function();
};

int X::how_many = 0;

void X::an_example_static_function() {
    cout << "how_many = " << how_many << endl;
}

int main() {
    X instance1;
    X instance2;
    X instance3;

cout << "X has " << X::how_many << " instances." << endl;

    X::an_example_static_function();

    return 0;
}
```

More information

For more info, please checkout http://en.cppreference.com/w/cpp/language/static