Содержание

пространства имён и using 3
namespace 3
Определение пространства имен 3
Вложенные пространства имен 4
using 4

пространства имён и using

namespace

Пространство имен позволяет сгруппировать функционал в отдельные контейнеры. Пространство имен представляет блок кода, который содержит набор компонентов (функций, классов и т.д.) и имеет некоторое имя, которое прикрепляется к каждому компоненту из этого пространства имен. Полное имя каждого компонента — это имя пространства имен, за которым следует оператор :: (оператор области видимости или scope operator) и имя компонента. Примером может служить оператор cout, который предназначен для вывода строки на консоль и который определен в пространстве имен std. Соответственно чтобы обратиться к этому оператору, применяется выражение std::cout.

Глобальное пространство имен - пространство имен по-умолчанию, используемое если не указанно иное пространство имен.

пример

```
#include <iostream>

void print(const std::string&);

const std::string message{"hello"};

int main() {
  print(message);
}

void print(const std::string& text) {
  std::cout << text << std::endl;
}
```

Здесь определены функции print и main и константа message и не используется никакого пространства имен. Поэтому фактически функции print и main и константа message определены в глобальном пространстве имен. В принципе для обращения к ним также можно использовать оператор ::, только без названия пространства имен, хотя это и избыточно:

```
int main() {
  ::print(::message);
}
```

Определение пространства имен

```
#include <iostream>

namespace hello {
```

```
const std::string message{"hello work"};
void print(const std::string& text) {
  std::cout << text << std::endl;
}

} // namespace hello

int main() {
  hello::print(hello::message); // hello work
}
```

Вложенные пространства имен

Одно пространство имен может содержать в себе другое

```
#include <iostream>

namespace console {
namespace messages {

const std::string hello{"hello"};
const std::string welcome{"Welcome"};
const std::string goodbye{"Good bye"};

} // namespace messages

void print(const std::string& text) {
  std::cout << text << std::endl;
}
void print_default() {
  std::cout << messages::hello << std::endl;
}

} // namespace console

int main() {
  console::print(console::messages::hello);
}
```

для обращения к члену вложенного пространства имен нужно использовать цепочку обращений.

using

используеться для задания псевдонимов[1)]

Например:

```
#include <iostream>

using ullong = unsigned long long;

int main() {
  ullong n {10234};
  std::cout << n << std::endl;
}
```

стоит отметить что для задания псевдонимов можно исопльзовать и старых подход из Си через typedef:

```
#include <iostream>

typedef unsigned long long ullong;

int main() {
    ullong n {10234};
    std::cout << n << std::endl;
}
```

так же using можно использовать для задания членов пространств имен, которые мы сможем использовать не указывая пренадлежность к пространству имен

`using пространство_имен::объект`

```
#include <iostream>
using std::cin;
using std::cout;
using std::endl;

int main() {
  int age;
  cout << "Input age: ";
  cin >> age;
  cout << "Your age: " << age << endl;
}
```

или и вовсе охватить все члены пространства имен так: `using namespace пространство_имен`

1)

пседонимов, а не новых типов данных!

From:
https://wiki.radi0.cc/ - **radi0wiki**

Permanent link:
https://wiki.radi0.cc/cpp:cpp_ultimate_guide:namespace

Last update: **2025/11/09 12:07**