How to use lambda expression in C++

Table of Contents

Author: dong1.nguyen@lge.com

2019.06.17

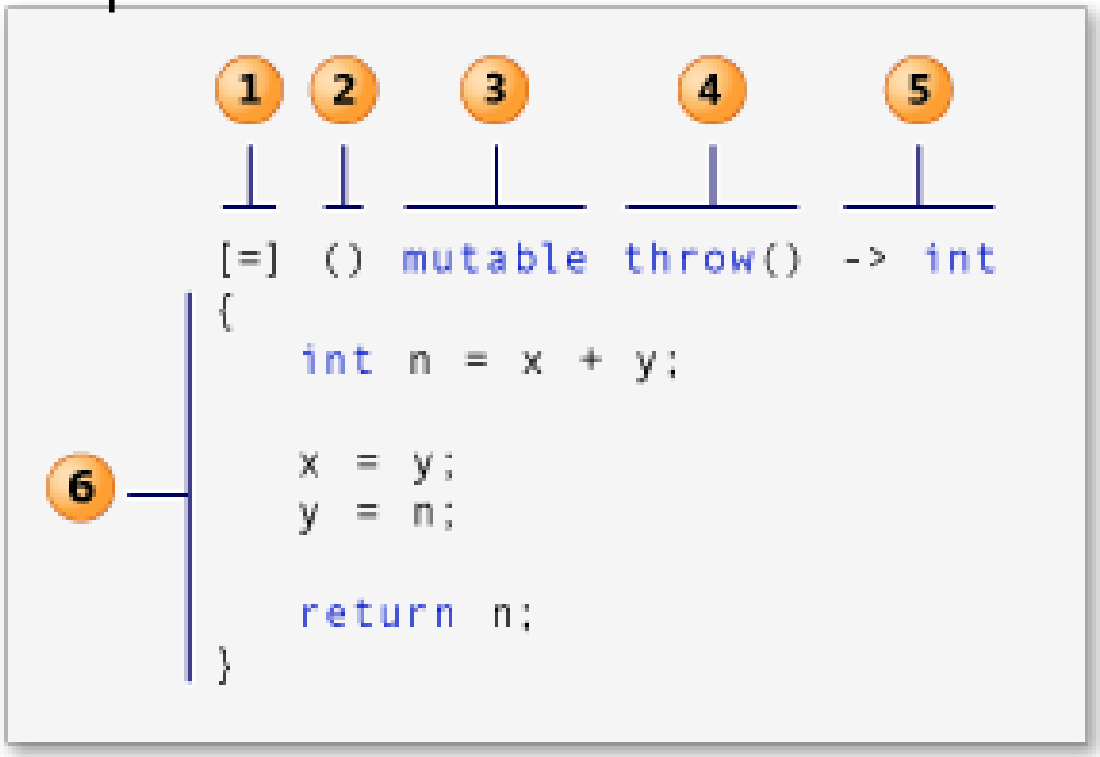
VC DCV

1. Lambda Expression - definition

- In C++11 and later, a lambda expression - often called a *lambda*—is a convenient way of defining an anonymous function object (a *closure*) right at the location where it is invoked or passed as an argument to a function.

2. Parts of a Lambda Expression

1. *Captured parameter list*
2. *Parameter list* Optional.
3. *Mutable specification* Optional.
4. *Exception-specification* Optional.
5. *Return-type* Optional.
6. *Lambda body*.



2. Parts of a Lambda Expression

Example:

```
#include<iostream>

int main() {
    auto x = [](int a) -> int { return a + 5; } (10);
    std::cout << x;
    return 0;
}
```

2. Parts of a Lambda Expression

Capture clause

- By reference
- By value
- Both reference and value

```
#include<iostream>

int main() {
    int a = 5;
    [&] { a = a + 1; std::cout << a << std::endl; }();   // --> In 6
    return 0;
}
```

2. Parts of a Lambda Expression

All capture reference

```
1  #include<iostream>
2  using namespace std;
3  int main() {
4      int a = 5;
5      int b = 7;
6      [&]() { a = a + 1; b = b - 1; cout << a << " " << b << endl; }();   // --> In 6 6
7      cout << a;  // --> In 6
8      cout << b;  // --> In 6
9      return 0;
10 }
```

2. Parts of a Lambda Expression

All capture value

```
1  #include<iostream>
2  using namespace std;
3  int main() {
4      int a = 5;
5      int b = 7;
6      auto x = [=]() { cout << a << " " << b << endl; return a + b;}();   // --> In 5 7
7      cout << x;  // --> In 12
8      cout << a;  // --> In 5
9      cout << b;  // --> In 7
10     return 0;
11 }
```

All capture value -> Build error

```
1  #include<iostream>
2  using namespace std;
3  int main() {
4      int a = 5;
5      int b = 7;
6      [=]() { a = a + 1; cout << a << " " << b << endl;}();   // --> Build error
7      cout << a;  // --> In 5
8      cout << b;  // --> In 7
9      return 0;
10 }
```

2. Parts of a Lambda Expression

Both capture reference and value

```
1  #include<iostream>
2
3  int main() {
4      int a = 0, b = 0;
5      [a, &b]() { a = 1; b = 1; } ();
6      std::cout << a << std::endl; // in 0
7      std::cout << b << std::endl; // in 1
8  }
```

2. Parts of a Lambda Expression

Using mutable, all capture value

```
1  #include<iostream>
2  using namespace std;
3  int main() {
4      int a = 5;
5      [=]() mutable { a = a + 1; cout << a << endl; }();    // --> In 6
6      cout << a;  // --> In 5
7      return 0;
8  }
```

2. Parts of a Lambda Expression

Using mutable, a capture value, b capture reference

```
#include<iostream>

int main() {
     int a = 0, b = 0;
    [a, &b]() mutable { a = 1; b = 1; } ();
    std::cout << a << std::endl; // in 0
    std::cout << b << std::endl; // in 1
}
```

2. Parts of a Lambda Expression

Nesting Lambda Expressions

```
// nesting_lambda_expressions.cpp
// compile with: /EHsc /W4
#include <iostream>

int main()
{
    using namespace std;

    // The following lambda expression contains a nested lambda
    // expression.
    int timestwoplusthree = [](int x) { return [](int y) { return y * 2; }(x) + 3; }(5);

    // Print the result.
    cout << timestwoplusthree << endl;
}
```

3. QUIZ question

1. Is this valid C++ 11 program & get compiled without any error?

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

int main() {

vector<int> v;
// vector gets filled

for_each(v.begin(), v.end(), [](int i) { cout << i*i << endl; } );

return 0;
}
```

3. QUIZ question

2. What is the correct syntax for lambda expression in C++11?

a) [capture](parameters)->return-type {body}
b) [parameters](capture)->return-type {body}
c) [capture][parameters]->return-type {body}
d) (capture)(parameters)->return-type {body}

3. Which of the following operator is used to capture all the external variable by reference?

a) &
b) =
c) *
d) &&

4. Which is the correct syntax of capturing a variable ‘X’ by reference and other variable ‘Y’ by value in lambda expression?

a) [&X, Y]
b) [X, &y]
c) [X, Y]
d) [&x, &Y]

3. QUIZ question

5. What is the output of the following C++ code?

```
#include<iostream>
using namespace std;
int main()
{
        int x = 5;
        auto check = []() -> bool
        {
                if(x == 0)
                        return false;
                else
                        return true;
        };
        cout<<check()<<endl;
        return 0;
}
```

a) 1
b) 0
c) Error
d) Segmentation fault

4. Q&A

Q&A

End…

Thank you!