

سلسلة تعلم البرمجة بلغة C++ الحديثة

Learn Modern C++ Programming Course

إعداد المهندس أحمد الديب



#6: Value Categories

lvalue & rvalue

lvalue & rvalue

- an **lvalue** is an expression that refers to an object (contiguous region of storage).
- **rvalue** means a value that is not an lvalue.
- lvalue originally meant “something that can be on the left-hand side of an assignment.” Not every lvalue may be used on the left-hand side of an assignment; an lvalue can refer to a constant
- An lvalue that has not been declared `const` is often called a **modifiable lvalue**.
- Address of an lvalue may be taken by built-in address-of operator `&`.

Example 1

```
int var1 = 4;  
int *pvar = &var1;  
int var2 = var1 + 4;
```

```
&var1 = 10; // error: lvalue required as left operand of assignment
```

```
(var1 = 4) = 10;  
(var1 + 4) = 10; // error: lvalue required as left operand of assignment
```

Example 2

```
int var = 20;

int& fun1() { return var; }
int fun2() { return var; }

int main() {
    int x1 = fun1(); // fun1() is lvalue
    fun1() = 10;

    int x2 = fun2(); // fun2() is rvalue
    fun2() = 10; // error: lvalue required as left operand of assignment

    return 0;
}
```

C++11 General rule

- Two properties that matter for an object when it comes to addressing, copying, and moving.
- **Has identity:** The program has the name of, pointer to, or reference to the object.
- **Movable:** The object may be moved from; move semantics e.g. pointer to dynamically allocated memory.

More categories

Has identity	Movable	category	category
Y	N	lvalue	generalized lvalue
Y	Y	xvalue	
N	Y	prvalue	rvalue

```
42 // movable (x = 42) and not identifiable -> rvalue, prvalue
a + b // movable (x = a + b) and not identifiable -> rvalue, prvalue
++a // ++a = 20 is OK -> identifiable and not movable -> glvalue, lvalue
a++ // a++ = 20 is ERROR -> movable and not identifiable -> rvalue, prvalue
```

Pre-increment and pre-decrement operators increments or decrements the value of the object and returns a reference to the result.

Post-increment and post-decrement creates a copy of the object, increments or decrements the value of the object and returns the copy from before the increment or decrement.

Thank you