# Конспект №8: rvalue, lvalue.

## *Теоретический материал.*

Каждое выражение принадлежит ровно одной категории значений: lvalue, xvalue или prvalue. В процессе вычисления, значения могут переходить из одной категории в другую, но все выражения, в конечном итоге, преобразовываются к категории значений rvalue. В менее широком смысле, все категории можно разделить на левосторонние (изменяемые) и правосторонние (неизменяемые). На примере определение категорий выглядит так:

Операнд1 = Операнд2;

Операнд1 - левостороннее значение (glvalue, lvalue, xvalue)

Операнд2 - левостороннее или правостороннее значение (rvalue, prvalue, xvalue)

LVALUE - данные, которым можно присвоить какое-либо значение и адрес расположения этих данных в памяти можно получить, используя оператор "&". К левосторонним значениям относятся: переменные, ссылки, раскрытые и нераскрытые указатели, возвращаемое функцией значение по ссылке или указателю.

RVALUE - данные, которые можно присвоить левосторонним данным и адрес расположения в памяти которых нельзя получить, используя оператор "&". К правосторонним значениям относятся: результат применения "&", приведение значения, исключения, возвращаемое функцией значение не по ссылке или указателю.

* XVALUE ссылается на объект, как правило, ближе к концу его жизни.
* GLVALUE может быть lvalue или xvalue, можно применить оператор "&".
* PRVALUE представляет собой значения констант.

## Пример использования.

int f0() { int i = 23; return i; }

int& f1() { int i = 23; return i; }

int main()

{

int p = 5;

//&5; // Ошибка. Выражение должно быть lvalue.

&p; // Ок.

int \*t = &p; // Ок.

&t; // Ок.

//&(double)p; // Ошибка. Выражение должно быть lvalue.

&(double&)p; // Ок.

double \*x = &\*(double\*)&p; // Ок.

cout << \*(int\*)x << endl;

//&f0(); // Ошибка. Выражение должно быть lvalue.

cout << &f1() << endl; // Ок.

return 0;

}

Источники:

* [lvalue и rvalue](http://enotcpp.blogspot.com/2012/06/l-valuer-value.html)