Static\_cast – это явное преобразование типов. (например int в float, double, или в любые другие типы данных). Также, можно преобразовать данные в указатель, или вообще в любой тип, в какой захотите.

Операция static\_cast доступна только в языке C++. static\_cast может быть использована для преобразования одного типа в другой, но она не должна быть использована для выполнения недопустимого преобразования, например, преобразование значения в указатель или наоборот.

Преобразования между числовыми типами. int, long, char, unsigned int — все их можно кастить друг в друга при помощи static\_cast.

Операция static\_cast является защищённой, в отличие от C-style cast, т.е. она не даст вам преобразовать одни данные в другие так, что бы программа сломалась.

Операция C-style cast – это основная функция преобразования типов, в языке С++. Она может преобразовать абсолютно любые типы данных в другие, и ей совершенно не важно что получится в результате, будет ли он иметь смысл, сломает ли он компьютер – нет, она об этом не думает, она просто берёт и преобразовывает, как образцовая функция. Будь как эта функция, делай не важно какую по сложности работу, вне зависимости от получаемого результата, и возможно ты закончишь ВУЗ с отличием!

Операция dynamic\_cast доступна только в C++. Динамическое приведение типов данных может быть использовано для безопасного приведения указателя (или ссылки) на суперкласс, в указатель (или ссылку) на подкласс в иерархии классов. Если динамическое приведение типов — недопустимо, так как реальный тип объекта, указывает не на тот тип подкласса, приведение типов не выполнится.

Динамическое приведение указателя

При приведении указателя, в случае неудачи, dynamic\_cast возвращает нулевой указатель NULL. Такое поведение обеспечивает быстрый способ определения, является ли данный объект частностью динамического типа.

(Я буду читать только 1ю страницу)

У static\_cast очень много различных применений. Большую часть времени, когда вы хотите сделать явное преобразование типов, вы хотите именно static\_cast.

Формальный список всего, что умеет static\_cast, [очень большой](http://en.cppreference.com/w/cpp/language/static_cast), я приведу лишь наиболее важные вещи, которые он умеет (а также которые он *не умеет*):

Что можно:

1. Преобразование указателя на родительский класс к указателю на дочерний класс. Объект по указателю **обязан** быть правильного дочернего класса, иначе undefined behaviour. Если вы не уверены и хотите проверить, тот ли подкласс у объекта, пользуйтесь dynamic\_cast (он специально для этого предназначен).
2. Преобразования между числовыми типами. int, long, char, unsigned int — все их можно кастить друг в друга при помощи static\_cast.
3. Можно закастить любое выражение в void. Результат будет вычислен и отброшен (но побочные эффекты, разумеется, выполнятся).
4. static\_cast может привести константу nullptr к любому типу-указателю. Обычно это не нужно и можно полагаться на неявное преобразование типов, но иногда (например, для выбора нужной перегрузки функции) такое может пригодится.

Что нельзя:

1. Преобразование между указателями на в принципе несовместимые типы. Например, указатель на double нельзя привести к указателю на int. Для трюков с нарушением type safety пользуйтесь reinterpret\_cast.
2. Указатели на типы, а также сами типы с несовместимыми атрибутами const и/или volatile. Если вам необходимо нарушить const-корректность, пользуйтесь const\_cast.
3. Разумеется, вы не сможете привести указатель на функцию-член к указателю на обычную функцию, или указатель на код к указателю на данные. Для подобных грязных хаков пользуйтесь reinterpret\_cast.

Ещё одной причиной использования static\_cast (как и других C++-специфических преобразований типов) является лёгкость его поиска в исходниках, как глазами, так и поисковыми утилитами. Сишный каст (особенно его функциональную разновидность) очень легко пропустить в коде.

static\_cast — унарная операция приведения типов данных в С++

Операция static\_cast доступна только в языке C++. static\_cast может быть использована для преобразования одного типа в другой, но она не должна быть использована для выполнения недопустимого преобразования, например, преобразование значения в указатель или наоборот. Рекомендуется пользоваться операцией static\_cast, нежели Cи-стилем приведения, потому что static\_cast ограничивает недопустимое приведение типов и, следовательно — безопаснее.

Операция static\_cast, грубо говоря, — это шаблон функции, в которой необходимо явно указать тип данных для преобразования, то есть задать параметр шаблона.

static\_cast<dataType>(value);

Итак, в треугольных скобочках указывается тип данных, к которому необходимо преобразовать значение value, которое стоит в круглых скобочках.

Операция dynamic\_cast доступна только в C++. Динамическое приведение типов данных может быть использовано для безопасного приведения указателя (или ссылки) на суперкласс, в указатель (или ссылку) на подкласс в иерархии классов. Если динамическое приведение типов — недопустимо, так как реальный тип объекта, указывает не на тот тип подкласса, приведение типов не выполнится.

Динамическое приведение указателя

При приведении указателя, в случае неудачи, dynamic\_cast возвращает нулевой указатель NULL. Такое поведение обеспечивает быстрый способ определения, является ли данный объект частностью динамического типа.

Динамическое приведение ссылки

При приведении ссылочной переменной, невозможно вернуть указатель, в случае неудачи. Поэтому будет вызвано исключение std::bad\_cast (из заголовочного файла <typeinfo> ).