

# Explicit Casts

- C++ has four different cast operators
  - `dynamic_cast`, `static_cast`, `const_cast`, and `reinterpret_cast`
- Each cast has a special use-case.
- The syntax:


```
double myDouble{5.5};  
int i = static_cast<int>(myDouble);
```



**C-Casts** (`int i = (int)myDouble;` )

- should not be used.
- apply eventually a series of casts `static_cast` → `const_cast` → `reinterpret_cast`

# dynamic\_cast

- Converts a pointer or a reference of a class to a pointer or a reference in the same inheritance hierarchy.
- Can only be used on polymorphic types.  Inheritance
- Allows to cast up, down, and cross the inheritance hierarchy
- Type information at run time is used to determine if the cast is valid.
- If the cast is not possible, you will get a `nullptr` in case of a pointer or an `std::bad_cast`-exception in case of a reference.

# static\_cast

- Allows the conversion between related types
  - Pointer types in class hierarchies
  - Integral types into enumerations
  - Floating-point types into integrals types
- Difference to the `dynamic_cast`
  - Will be performed during compile time
  - Can not be applied to polymorphic types



Inheritance

# const\_cast

- `const_cast` allows it to remove or add the `const` or `volatile` qualifier.



Modifying a `const` or `volatile` declared object by removing its constness is undefined behavior.



[modifyingConst.cpp](#)

# reinterpret\_cast

- `reinterpret_cast` **allows it to convert**
  - a pointer type into another pointer.
  - an integral type into a pointer type and vice versa.
  
- Guarantees that the conversion to and from a pointer returns the same value.