Dynamic Cast

This lesson highlights the key features of the dynamic_cast operator.

Features

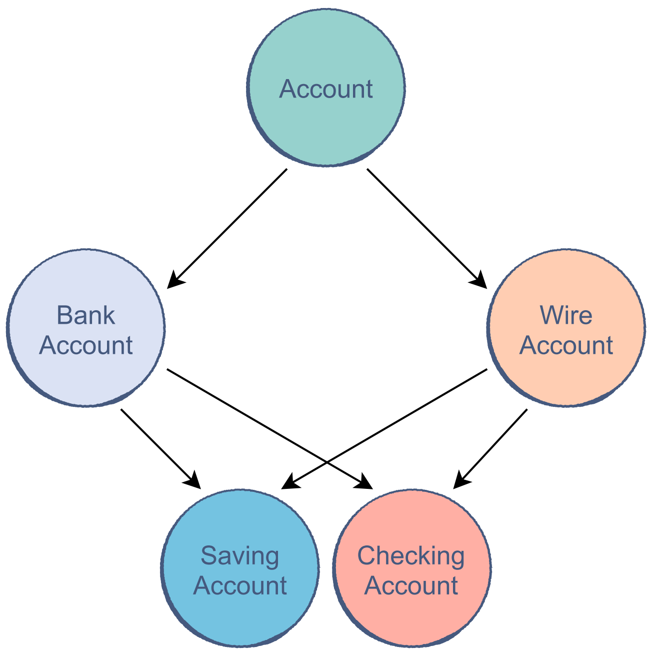
- `dynamic_cast` converts a pointer or reference of a class to a pointer or reference in the same inheritance hierarchy.
- It can only be used with polymorphic classes. With `dynamic_cast`, we cast **up**, **down**, and **across** the inheritance hierarchy.
- Type information at run time is used to determine if the cast is valid.
- If the cast is not possible, we will get a `nullptr` in case of a pointer, and an `std::bad_cast-exception` in case of a reference.
- `dynamic_cast` is mostly used when converting from a derived class to a base class, but can also work the opposite operation.

Example

```
class Account{
public:
  virtual ~Account() = default;
};

class BankAccount: virtual public Account{};

class WireAccount: virtual public Account{};

class CheckingAccount: public BankAccount, public WireAccount {};

class SavingAccount: public BankAccount, public WireAccount {};
```

```
int main(){

  Account * a = nullptr;
  BankAccount * b = nullptr;
  WireAccount * w = nullptr;
  SavingAccount * s = nullptr;

  CheckingAccount c;

  a = dynamic_cast<Account*> (&c);                   // upcast
  a = &c;                                            // upcast

  b = dynamic_cast<BankAccount*>(a);                 // downcast
  w = dynamic_cast<WireAccount*>(b);                 // crosscast
  s = dynamic_cast<SavingAccount*>(a);               // downcast

}
```

The classes in the code above form the following hierarchy:



From line 23 onwards, we can see how up, down, and cross casting is possible with `dynamic_cast`.

> Do keep in mind that `dynamic_cast` only deals with pointers and references.

In the next lesson, we'll explore `static_cast`.