Indirection and Inheritance

Kate Gregory

@gregcons www.gregcons.com/kateblog

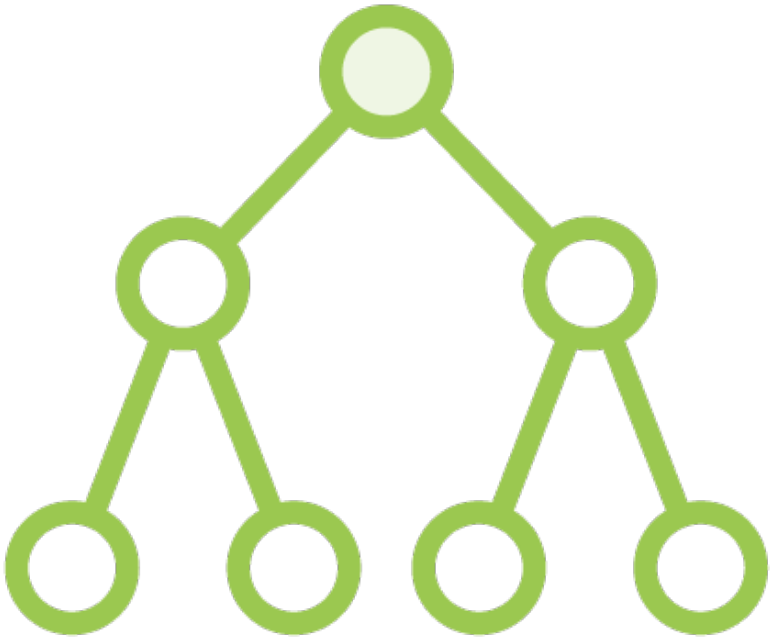
References and Inheritance

"Base class reference" can actually refer to a derived class instance

Respects the "is a" relationship

Vital to Liskov substitutability

References and Inheritance



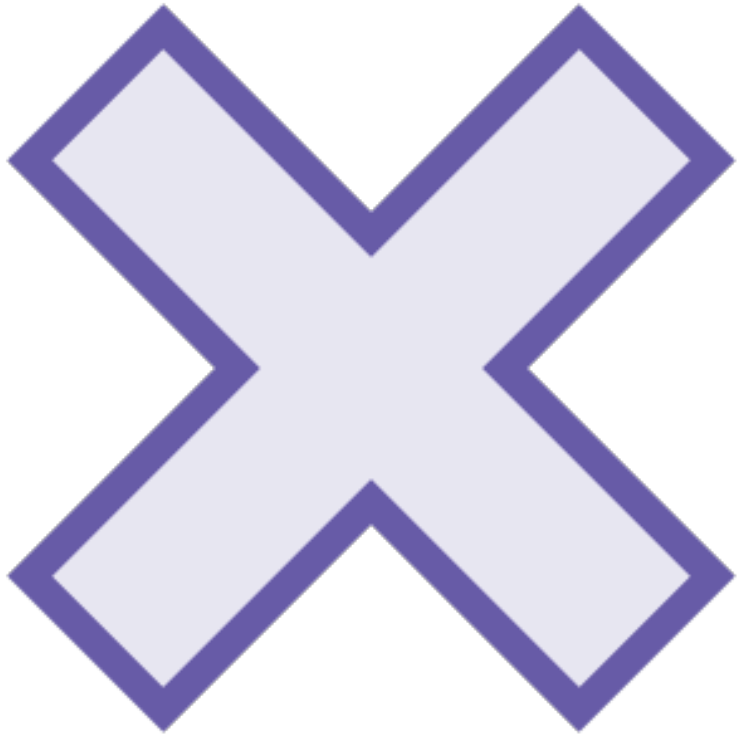
Any base class function can be called through a base reference to a derived instance

- Virtual function – derived class function executes
  - polymorphism
- Nonvirtual function – base class function executes
- This is C++ - you get to choose

Can't call derived class functions

- Compiler only knows this is a base class reference

References and Inheritance



Can't create a "derived class reference" that refers to a base class instance

- Some derived-class member variables would be missing

Pointers and Inheritance

A “pointer to a base class” can actually point to a derived class instance

Any base class function can then be called through the pointer

- Virtual function – derived class function executes
- Nonvirtual function – base class function executes
- This is C++ - you get to choose

Can’t aim to “pointer to derived class” at a base class instance

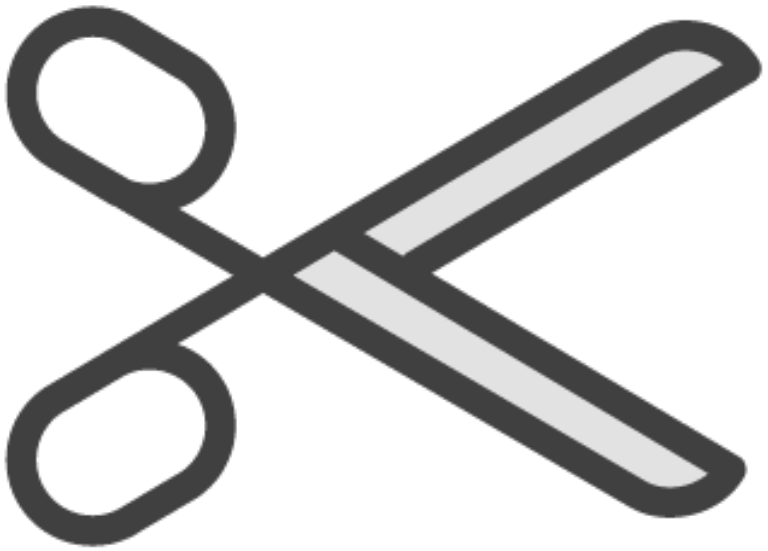
- Some derived-class member variables would be missing

Smart Pointers and Inheritance

Same rules when using smart pointers

Smart pointers act like regular pointers and that includes polymorphism

Slicing



If you copy objects around, slicing can occur

- Copy a derived object into a base object – extra member variables fall away
- Can't copy a base object into a derived object

Same rules apply when passing to a function by value

- A copy is made
- Slicing will happen

Use references or pointers to avoid slicing

- References use same syntax as solid objects

(type)

C style cast

Super dangerous

Doesn't tell humans much when they read your code

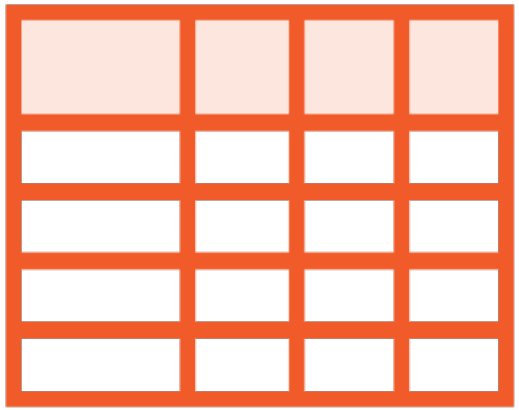
static_cast<type>

Compile time only

Up to you to be sure it’s safe

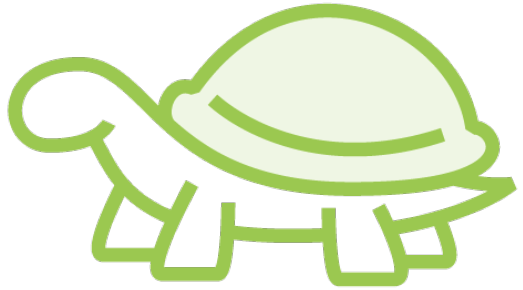
dynamic_cast<type>

Runtime check



Works only when casting to pointer to a class with a virtual table

Returns null if cast fails



Slower but safer

Other Cast Operators

const_cast

For casting away const
(not a beginner technique)

reinterpret_cast

For bit twiddling

Summary

Polymorphism lets you write general code that relies on specific implementations

- Update all the accounts, ship all the orders, pay all the employees

Raw pointers, smart pointers, and references all support polymorphism

Copying solid objects derived to base can cause slicing

Cast operators give you safety and expressiveness