C++

Inheritance: public, private and protected

Public inheritance

- All the example of inheritance seen so far are **public** inheritance:
  class D : public B {};
- There exist two other types of inheritance: **protected** and **private** inheritance
- They are very rarely used

Inheritance

- Public inheritance preserves the original accessibility of the base class public and protected members in the derived class
  - (base) public -> (derived) public
  - (base) protected -> (derived) protected
  - (base) private -> no access
- Protected inheritance causes public members to become protected (protected members are preserved) in the derived class
  - (base) public -> (derived) protected
  - (base) protected -> (derived) protected
  - (base) private -> no access
- Private inheritance causes all members to become private in the derived class
  - (base) public -> (derived) private
  - (base) protected -> (derived) private
  - (base) private -> no access

Syntax

- For private inheritance:
  class D : private B {};
- For protected inheritance:
  class D : protected B {};

Usage

- Protected and private inheritance are very rarely used
- One possible usage is as a syntactic variant of composition
- Example: “Car has a Engine” relationship can be expressed as “Car privately inherits from Engine”
- In some particular occasion, private inheritance may have advantages over composition (“has-a” relationship)