Module 26: Programming in C++

Dynamic Binding (Polymorphism): Part 1

Sourangshu Bhattacharya

Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

sourangshu@cse.iitkgp.ac.in

Slides taken from NPTEL course on Programming in C++

by **Prof. Partha Pratim Das**

Module Objectives

- Understand Casting in a class hierarchy
- Understand Static and Dynamic Binding

Module Outline

- Casting
  - Upcast & Downcast
- Static and Dynamic Binding

Casting: Basic Rules

- Casting is performed when a value (variable) of one type is used in place of some other type

```
int i = 3;
double d = 2.5;

double result = d / i; // i is cast to double and used
```

- Casting can be implicit or explicit

```
int i = 3;
double d = 2.5;

double *p = &d;

d = i;      // implicit

i = d;      // implicit -- // warning C4244: '=' : conversion from 'double' to 'int',
            // possible loss of data
i = (int)d; // explicit

i = p;      // error C2440: '=' : cannot convert from 'double *' to 'int'
i = (int)p; // explicit
```

Casting: Basic Rules

- (Implicit) Casting between unrelated classes is not permitted

```
class A { int i; };
class B { double d; };

A a;
B b;

A *p = &a;
B *q = &b;

a = b;      // error C2679: binary '=' : no operator found
            // which takes a right-hand operand of type 'main::B'

a = (A)b;   // error C2440: 'type cast' : cannot convert from 'main::B' to 'main::A'

b = a;      // error C2679: binary '=' : no operator found
            // which takes a right-hand operand of type 'main::A'

b = (B)a;   // error C2440: 'type cast' : cannot convert from 'main::A' to 'main::B'

p = q;      // error C2440: '=' : cannot convert from 'main::B *' to 'main::A *'

q = p;      // error C2440: '=' : cannot convert from 'main::A *' to 'main::B *'

p = (A*)&b; // Forced -- Okay
q = (B*)&a; // Forced -- Okay
```

Casting: Basic Rules

- Forced Casting between unrelated classes is dangerous

```
class A { public: int i; };
class B { public: double d; };

A a;
B b;

a.i = 5;
b.d = 7.2;

A *p = &a;
B *q = &b;

cout << p->i << endl; // prints 5
cout << q->d << endl; // prints 7.2

p = (A*)&b;
q = (B*)&a;

cout << p->i << endl; // prints -858993459    ------------- GARBAGE
cout << q->d << endl; // prints -9.25596e+061 ------------- GARBAGE
```

Casting on a Hierarchy

- Casting on a hierarchy is permitted in a limited sense

```
class A {};
class B : public A {};

A *pa = 0;
B *pb = 0;
void *pv = 0;

pa = pb; // okay ------------------------------------------------ // UPCAST

pb = pa; // error C2440: '=' : cannot convert from 'A *' to 'B *'  // DOWNCAST

pv = pa; // okay ------------------------------------------------ // Lose the type
pv = pb; // okay ------------------------------------------------ // Lose the type

pa = pv; // error C2440: '=' : cannot convert from 'void *' to 'A *'
pb = pv; // error C2440: '=' : cannot convert from 'void *' to 'B *'
```

Casting on a Hierarchy

- Up-Casting is safe

```
class A { public: int dataA_; };
class B : public A { public: int dataB_; };

    A a;
    B b;

    a.dataA_ = 2;
    b.dataA_ = 3;
    b.dataB_ = 5;

    A *pa = &a;
    B *pb = &b;

    cout << pa->dataA_ << endl;                       // prints 2
    cout << pb->dataA_ << " " << pb->dataB_ << endl; // prints 3 5

    pa = &b;

    cout << pa->dataA_ << endl;                       // prints 3
    // cout << pa->dataB_ << endl; // error C2039: 'dataB_' : is not a member of 'A'
```

Static and Dynamic Binding


```
#include <iostream>
using namespace std;

class B {
public:
    void f() { cout << "B::f()" << endl; }
    virtual void g() { cout << "B::g()" << endl; }
};

class D: public B {
public:
    void f() { cout << "D::f()" << endl; }
    virtual void g() { cout << "D::g()" << endl; }
};

  int main() {
      B b;
      D d;

      B *pb = &b;
      B *pd = &d; // UPCAST

      B &rb = b;
      B &rd = d;  // UPCAST

      b.f(); // B::f()
      b.g(); // B::g()
      d.f(); // D::f()
      d.g(); // D::g()

      pb->f(); // B::f() -- Static Binding
      pb->g(); // B::g() -- Dynamic Binding
      pd->f(); // B::f() -- Static Binding
      pd->g(); // D::g() -- Dynamic Binding

      rb.f();  // B::f() -- Static Binding
      rb.g();  // B::g() -- Dynamic Binding
      rd.f();  // B::f() -- Static Binding
      rd.g();  // D::g() -- Dynamic Binding

      return 0;
  }
```

Module Summary

- Introduced casting and discussed the notions of upcast and downcast
- Introduced Static and Dynamic Binding