Inheritance

Prof. Wade Fagen-Ulmschneider

ILLINOIS

Inheritance allows for a class to inherit all member functions and data from a **base class** into a **derived class**.

Generic to Specialized

A **base class** is a generic form of a specialized, **derived class**.

Shape ➔ Cube

cpp-inheritance/Shape.h

```cpp
#pragma once

class Shape {
  public:
    Shape();
    Shape(double width);
    double getWidth() const;

  private:
    double width_;
};
```

cpp-inheritance/Cube.h

```cpp
#pragma once

#include "Shape.h"
#include "HSLAPixel.h"

namespace uiuc {
  class Cube : public Shape {
    public:
      Cube(double width, uiuc::HSLAPixel color);
      double getVolume() const;

    private:
      uiuc::HSLAPixel color_;
  };
}
```

Initialization

When a derived class is initialized, the derived class **must** construct the base class:

- **Cube** must construct **Shape**
- By default, uses default constructor
- Custom constructor can be used with an initialization list

cpp-inheritance/Cube.cpp

```cpp
#include "Cube.h"
#include "Shape.h"

namespace uiuc {
  Cube::Cube(double width, uiuc::HSLAPixel color) : Shape(width) {
    color_ = color;
  }

  double Cube::getVolume() const {
    // Cannot access Shape::width_ due to it being `private`
    // ..instead we use the public Shape::getWidth(), a public function:

    return getWidth() * getWidth() * getWidth();
  }
}
```

Access Control

When a base class is inherited, the derived class:

- Can access all **public** members of the base class
- Can **not** access **private** members of the base class

Initializer List

The syntax to initialize the base class is called the initializer list and can be used for several purposes:

- Initialize a base class
- Initialize the current class using another constructor
- Initialize the default values of member variables

cpp-inheritance/Cube.cpp

```cpp
#include "Shape.h"

Shape::Shape() : Shape(1) {
  // Nothing.
}

Shape::Shape(double width) : width_(width) {
  // Nothing.
}

double Shape::getWidth() const {
  return width_;
}
```