

Virtual Functions Part Two

Exercises

- What effect does the `override` keyword have?
- What do you expect to happen when the code on the next page is compiled?
- Compile the code and verify your answer
- What effect does the `final` keyword have?

```
class Drawable {  
public:  
    virtual void draw() { cout << "Drawable::draw()\n"; }  
};
```

```
class Circle : public Drawable {  
public:  
    void draw() override { cout << "Circle::draw()\n"; }  
    void draw(int radius) override { cout << "Circle::draw() with radius\n"; }  
};
```

- What is meant by a "pure virtual member function"?
- Why is it useful?
- What effect does making a member function pure virtual have?

- What is an "abstract base class"?
- What is unusual about an abstract base class?
- What is an abstract base class used for?

- In the program you wrote in the previous lesson, change `Drawable::draw()` to be a pure virtual function
- Compile and run your program again
- What happens if you try to create an instance of `Drawable`?
- Create a function which takes a `Drawable` instance by value
- What happens? Does this help programmers?