

◀ Lesson 4:
Writing Multifile Programs

≡

SEND FEEDBACK

SEARCH

Q

RESOURCES

▲

CONCEPTS

▼

1. Intro

2. Header Files

3. Using Headers with Multiple Files

4. Bjarne on Build Systems

5. CMake and Make

6. References

7. Pointers

8. Pointers Continued

9. Bjarne on pointers

10. References vs Pointers

11. Bjarne on References

12. Maps

13. Classes and Object-Oriented Pro...

14. Classes and OOP Continued

15. This Pointer

16. How Long Does it Take to Learn ...

17. Outro

This Pointer

When working with classes it is often helpful to be able to refer to the current class instance or object. For example, given the following `Car` class from a previous lesson, the `IncrementDistance()` method implicitly refers to the current `Car` instance's `distance` attribute:

```
// The Car class
class Car {
public:
    // Method to print data.
    void PrintCarData() {
        cout << "The distance that the " << color << " car " << number << " has traveled is: " << distance;
    }

    // Method to increment the distance travelled.
    void IncrementDistance() {
        distance++;
    }

    // Class/object attributes
    string color;
    int distance = 0;
    int number;
};
```

It is possible to make this explicit in C++ by using the `this` pointer, which points to the current class instance. Using `this` can sometimes be helpful to add clarity to more complicated code:

```
// The Car class
class Car {
public:
    // Method to print data.
    void PrintCarData() {
        cout << "The distance that the " << this->color << " car " << this->number << " has traveled is: ";
    }

    // Method to increment the distance travelled.
    void IncrementDistance() {
        this->distance++;
    }

    // Class/object attributes
    string color;
    int distance = 0;
    int number;
};
```

Note: you may see `this` used in some code in the remainder of the course.