# The sum Member Function

**Let's examine the behavior of `this` and `const` a little more closely, by** considering the `sum()` member function from `Time`:

```
class Time
{
public:
    Time sum(const Time& rhs) const;
    . . .
};
```

When you add two `Time` objects (`a` + `b`) together like this:

```
Time after = a.sum(b);
```

The **caller** (the implicit parameter) is the left-hand-side of the expression `a` + `b`. Thus, the effective implicit prototype for the function is similar to this:

```
Time sum(const Time* lhs, const Time& rhs);
```

In the implementation, however, instead of the **explicit `lhs` parameter** shown here, you'd use the keyword `this` to access the data members.

```
Time Time::sum(const Time& rhs) const
{
    auto tMinutes = this->m_hours * 60 + this->m_minutes;
    auto dMinutes = rhs.m_hours * 60 + rhs.m_minutes;
    . . .
}
```

If you leave off the keyword `this`, it is assumed. Notice that when you implement a `const` member function, you **repeat** the word `const` in the implementation.