# Copy on Write `structs`

80

```cpp
struct(cow) person
{
    std::string get_first_name() const
    {
        return first_name;
    }

    void set_first_name (std::string_view new_first)
    {
        first_name = new_first;
    }

    // Repeat for last_name

private:
    std::string first_name, last_name;
};
```

c.o.w. metaclass

↓

https://godbolt.org/z/Kd57jW7Wz

```
struct person
{
    std::string get_first_name() const {
        return person_->get_first_name();
    }

    void set_first_name (std::string_view new_first) {
        copy_if_shared();
        person_->set_first_name (new_first);
    }

    // Repeat for last_name

private:
    struct __person;
    static_assert (std::is_copy_constructible_v<__person>);

    std::shared_ptr<__person> person_
        = std::make_shared<__person>();

    void copy_if_shared() {
        if (person_.use_count() > 1)
            person_ = std::make_shared<__person> (*person_);
    }
};

template<>
struct is_send<person> : is_send_v<__person>{};
```

https://godbolt.org/z/h7T764n6a

# Copy on Write

```
struct(cow) person
{
    std::string get_first_name() const
    {
        return first_name;
    }

    void set_first_name (std::string_view
    {
        first_name = new_first;
    }

    // Repeat for last_name

private:
    std::string first_name, last_name;
};
```

```
struct person
{
    std::string get_first_name() const {
        return person_->get_first_name();
    }

    void set_first_name (std::string_view new_first) {
        copy_if_shared();
        person_->set_first_name (new_first);
    }

    // Repeat for last_name

private:
    struct __person;
    static_assert (std::is_copy_constructible_v<__person>);

    std::shared_ptr<__person> person_
        = std::make_shared<__person>();

    void copy_if_shared() {
        if (person_.use_count() > 1)
            person_ = std::make_shared<__person> (*person_);
    }
};

template<>
struct is_send<person> : is_send_v<__person>{};
```

## Copy on Write `struct`s