Wrapped `std::weak_ptr`

77

```
class person
{
public:
    class weak_ref
    {
    public:
        weak_ref() = default;
        weak_ref (person p)
            : person_ (p.person_) {}

        std::optional<person> get() const
        {
            if (auto valid = person_.lock())
                return person (std::move (valid));

            return std::nullopt;
        }

    private:
        std::weak_ptr<__person> person_;
    };

    //... rest of class as before

private:
    person (std::shared_ptr<__person>&& other)
        : person_ (other) {}
};
```

```
person p1;
//... do stuff with p1

person::weak_ref p2; // create uninitialised
p2 = p1;             // assign from strong-ref

if (auto valid_person = p3.get())
    std::println ("p3 {}", valid_person->get_first_name());

p2.get().transform ([] (auto valid_person) {
                        valid_person.set_first_name (“John");
                        return valid_person;
                    });
```

https://godbolt.org/z/fd9oGWoTo

Wrapped std::weak_ptr

```
class person
{
public:
    class weak_ref
    {
    public:
        weak_ref() = default;
        weak_ref (person p)
            : person_ (p.person_) {}

        std::optional<person> get() const
        {
            if (auto valid = person_.lock())
                return person (std::move (valid));

            return std::nullopt;
        }

    private:
        std::weak_ptr<__person> person_;
    };

    //... rest of class as before

private:
    person (std::shared_ptr<__person>&& other)
        : person_ (other) {}
};
```

```
person p1;
//... do stuff with p1

person::weak_ref p2; // create uninitialised
p2 = p1;             // assign from strong-ref

if (auto valid_person = p3.get())
    std::println ("p3 {}", valid_person->get_first_name());

p2.get().transform ([] (auto valid_person) {
                        valid_person.set_first_name (“John");
                        return valid_person;
                    });
```

Swift `structs`