Cyclic References

In this lesson, we'll examine how the use of shared pointers can create a reference cycle and why this could be harmful.

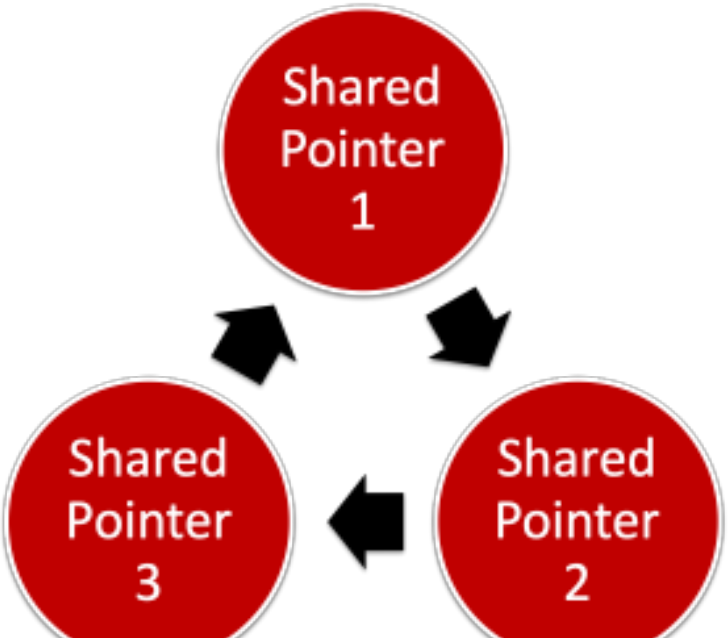
WE'LL COVER THE FOLLOWING

We get **cyclic references** of `std::shared_ptr` if they refer to each other.

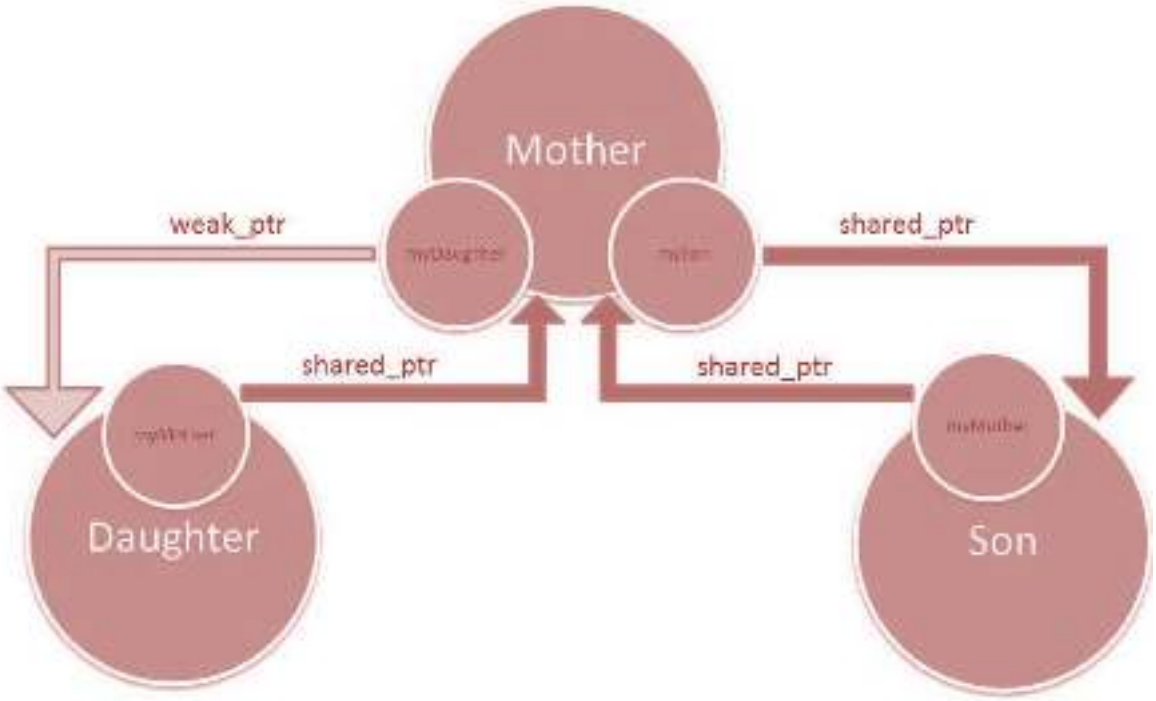
The issue

If we have a cyclic reference of `std::shared_ptr`, the reference counter will never become 0. We can break this cycle if by embedding an `std::weak_ptr` in the cycle. `std::weak_ptr` does not modify the reference counter.

> Theoretically, we can use a raw pointer to break the cycle of `std::shared_ptr`'s, but a raw pointer has two disadvantages. First, they don't have a well-defined interface. Second, they don't support an interface that can create an `std::shared_ptr` out of it.



There are two cycles in the graphic below: first, between the mother and her daughter; second, between the mother and her son. The subtle difference is that the mother references her daughter with an `std::weak_ptr`. Therefore, the `std::shared_ptr` cycle is broken.



To better understand, take a look at the corresponding source code in the example in the following lesson.