

Shared Pointer

- To create an initialized instance, we call `make_shared`

```
auto p{make_shared<int>(36)};    // Available from C++11
```

- `make_shared` allocates the reference counter and the memory object in a single block of memory
 - Unlike `make_unique`, it is not a wrapper for `new()`
 - If we initialize a `shared_ptr` by calling `new()` explicitly, the memory object and reference counter are allocated separately

```
shared_ptr<int> p{new int(36)};    // Not recommended - less efficient
```

Copying shared pointers

- Unlike unique pointers, shared pointers can be copied and assigned
- Copying a shared pointer instance will cause the shared pointer's counter to increase by one

`p2(p1);`

- Before the copy, p1's reference counter had the value 1
- After the copy, p2 now shares p1's memory object
- The counter in the shared object will have the value 2

Assigning shared pointers

- Assignment

`p2 = p1` // Where p2 and p1 are pointing at different memory objects

- p2 and p1 now share the same memory object
- The counter in p1 is incremented and the counter for p2's old memory object is decremented
- If this counter becomes zero, p2's old memory object will be deleted