

# Unique Pointer

- To create an initialized instance, we call `make_unique()`

*// Create a unique\_ptr to an int which has initial value 36*

```
auto p{ make_unique<int>(36) };
```

- This will call `new` and bind the result to the pointer member of `p`
- `make_unique` was added in C++14. In C++11, we have to call `new()` explicitly

```
unique_ptr<int> p{ new int(36) };
```

## unique\_ptr Example

```
// Data structure representing a point on the screen
```

```
struct point {  
    int x;  
    int y;  
};
```

```
// Create a unique_ptr to an point which has initial value {3,6}
```

```
auto p{ make_unique<point>( point{3, 6} ) };
```

```
unique_ptr<point> p{ new point{3, 6} };      // C++11
```

```
cout << p->x << ", " << p->y << endl;
```

```
auto pcopy = p;                             // Error
```

## Returning an unique\_ptr instance (contd)

- The ownership of the memory will be transferred from the local instance to the returned instance

```
unique_ptr<int> create_ptr() {  
    int x{0};  
    ...  
    unique_ptr<int> p = make_unique<int>(x);  
    return p;  
}  
  
unique_ptr<int> upi { create_ptr() };
```

// Calculate value of x  
// Create the local instance  
// The pointer member is transferred  
// from p to the returned instance