

Classes 101

What do I do with all this new syntax?

From where do Classes come?

- Read the project specification
- Find the nouns
 - Person, place, thing or idea
- Decide which ones are the most important
- Those are usually the classes.

Technicalities

- Scope and Access of Members
 - Data and Methods within the class have class scope, i.e. only data and methods within the class have direct access to the data and methods
 - Data and functions outside the class are at a file scope and do not have direct access to the data and methods inside the class

More on Scope

- Variables inside a method have method scope
- If a variable in a function has the same name as an already created class variable, then the function variable hides the class variable
- The hidden class variable can be used through the scope resolution operator ::

Accessing Data and Methods

- Just like structs, we use the dot (.) notation to access public data and methods of class
 - Example
 - Suppose we have an object of a Time class called Today and the class has a public data member Hour.
 - We can access hour like this:
 - `Today.Hour = 4;`

Constructors

- Constructors are called whenever an object is created.
- Used to initialize any data inside the object
- Can have multiple constructors for various purposes
- Must have at least one default constructor

Interface and Implementation

- Place class definition in a separate file
- Place class implementation in another file
- This hides implementation details from users
- ...or does it?

Controlling Access to Members

- Public
- Private
- Protected

Access and Utility Functions

- Public functions in the interface are called Access functions
- Private functions are called utility functions because other member functions use them

Constructors, again

- Default constructor called each time an object is created or instantiated
- Two ways to code default constructor
 - `MyClass::MyClass()`
 - `MyClass::MyClass(int x=0)`

Constructors

```
class MyClass{
```

```
public:
```

```
    MyClass();
```

```
private:
```

```
    int x;
```

```
};
```

```
MyClass::MyClass()
```

```
{
```

```
    x=0;
```

```
}
```

```
class MyClass{
```

```
public:
```

```
    MyClass(int y=0);
```

```
private:
```

```
    int x;
```

```
};
```

```
MyClass::MyClass(int y)
```

```
{
```

```
    x=y;
```

```
}
```

Destructors, again

- Called whenever the class object goes out of scope
- Has only one signature
- `MyClass::~~MyClass();`

Set and Get Functions

- Allow clients of the class to adjust or retrieve private data
- If every data member has a set and get member function, how is this different from a struct or public data?

Watch Out

- Returning a reference to a private member
- Not gonna do it
- Reference returning functions can be used as an *l-value* and therefore can be used to clobber the data member they are returning

What happens when you say =

- For simple classes, with no dynamic data, everything works fine.
- Memberwise assignment takes place.
- Get into trouble when we start looking at pointers

Quiz

- Write a default constructor for a time class that can receive up to 3 parameters; the hour (use 24 hour time), minute and seconds.
- Make sure your constructor checks for valid input.
- Use the following class name: Time
- Store the internal data as seconds.