Objects in C++

See also: Chapters 1, 8

Class Declarations Definitions
In C++, the prototype for a class is called a class definition, even though it more closely resembles a function declaration. The class definition follows this form:

```cpp
class SpreadsheetCell
{
    public:
        void setValue(double inValue);  // method
        double getValue();

    protected:
        double mValue;  // member
};
```

Access Levels
Remember that access levels define how clients and subclasses of your class can use methods and members. They do not apply to other objects of the same class. Also, remember access levels have nothing to do with subclass overriding privileges. In C++, a subclass can override a private method even though it can't call the original method.

<table>
<thead>
<tr>
<th>Access Specification</th>
<th>Meaning</th>
<th>When to Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>public</td>
<td>Any code can call a public method or access a public member of an object.</td>
<td>Methods that you want clients to use. Accessor methods for data members.</td>
</tr>
<tr>
<td>protected</td>
<td>Methods of the class can call protected methods &amp; use protected members. Methods of a subclass can as well.</td>
<td>“Helper” methods that you do not want clients to use. Most data members.</td>
</tr>
<tr>
<td>private</td>
<td>Only methods of the class can call a private method and access a private member. Methods in subclasses cannot access private methods or members.</td>
<td>Only if you want to restrict access from subclasses.</td>
</tr>
</tbody>
</table>
Class Implementation

```cpp
// SpreadsheetCell.cpp
#include "SpreadsheetCell.h"

void SpreadsheetCell::setValue(double inValue)
{
    mValue = inValue;
}

double SpreadsheetCell::getValue()
{
    return (mValue);
}
```

Stack Based Object Creation and Usage

```cpp
SpreadsheetCell myCell, anotherCell;
myCell.setValue(6);
anotherCell.setValue(myCell.getValue());

cout << "cell 1: " << myCell.getValue() << endl;
cout << "cell 2: " << anotherCell.getValue() << endl;
```

Heap Based Objects

```cpp
SpreadsheetCell* myCellp = new SpreadsheetCell();

myCellp->setValue(3.7);
cout << "cell 1: " << myCellp->getValue() << " " << myCellp->getString() << endl;
delete myCellp;
```