C++ Assignment 4

Garrett Keefe

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

UID: u1232643

Setting

-value : String

+ setting(string value)  
+ ~setting()  
+ change(string value)  
+ get()

Server Setting

+ server\_setting(string v)  
+ ~server\_setting()  
+ change(string v)  
+ get()  
# has\_server\_setting\_changed()  
# get\_server\_setting()  
# change\_server\_setting(string v)

Notify

+ notify(string init\_value)  
+ ~notify()  
+ change(string curr\_value)  
+ register\_observer(observer \*)  
# notify\_observers()

GUI Setting

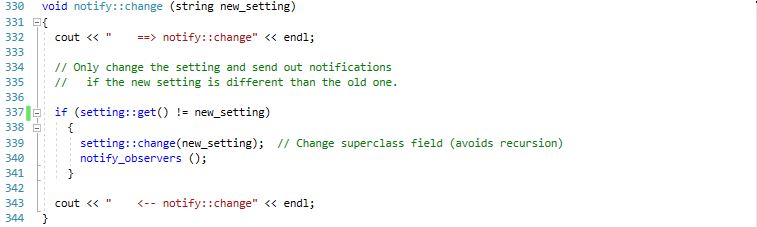
+ gui\_setting(string value)  
+ ~gui\_setting()  
+ change(string v)

Identified Problems

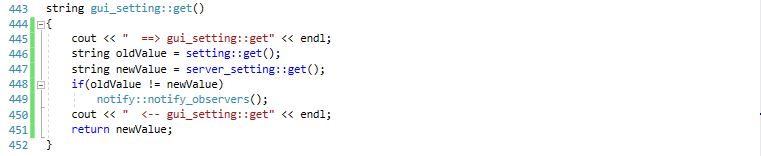
1. The first problem with the program occurs when gui\_setting::change() is called. Within the function, notify::change() is called as well. On its own, the notify::change() functions works as expected, but when it is called from a gui\_setting object, the behavior of one of the lines of codes is changed and breaks the program. Within notify::change(), there is a get() call that helps to evaluate if the object nees to notify its observers that the setting has been changed. Normally, this would call setting::get() and then check if the old value is equal to the new value. However, because our object is now a gui\_setting object and not a notify object, the get call that is evaluated as the base is now server\_setting::get(). This is true of the get call within notify::change() as well. This change will break notify::change() in the case scenario where the value of gui\_setting::change is the same as the current setting within the remote server. For example, if you call gui\_setting::change(“foggy”) and the remote value of the server\_setting has recently changed to “foggy”, the notify object will see that the setting value is now equal to “foggy” and assume that it doesn’t need to update. This is wrong because server\_setting::get() has just changed the value of the setting when it found that the remote value of the server was different than expected. The observers for the object still need to be notified of the fact that the gui\_setting object was modified.
2. The second problem is similar to the first. When the remote value of the server is edited in a gui\_setting object, it will update the setting::value string when get() is called from the gui\_setting object. This is fine, the problem occurs when you realize that the setting has been changed to a different value, but notify::notify\_observers() was never called, because notify::change() was never called. For example, the gui\_setting object intially has value = “clear” but then the server is updated remotely to be equal to “darkened”. The next time that the gui\_setting.get() is called, it will find out that the remote value has changed and will update the setting::value string to “darkened”. This process never even touches the notify side of the inheritance and therefore cannot call the notify\_observers() despite the fact that it is needed for the gui\_setting to achieve the expected behavior.

Fixes

The fix for the first problem is extremely easy to implement. All that needs to change is that the get() call within notify::change() simply needs to be specified as a setting::get() call. Example below.



The fix for the second problem is slightly more complicated, but isn’t too bad. We can simply override the get() call within the gui\_setting object to implement the additional functionality we need to notify the observers. Example below.



All we do here is check that the value before we run server\_setting::get() is the same as after, and if not, that means that the remote value was changed and server\_setting::get() just updated the string setting::value. This means that we need to notify the observers of our gui\_setting object that the setting value has changed.

The updated UML diagram is below.

Setting

-value : String

+ setting(string value)  
+ ~setting()  
+ change(string value)  
+ get()

Server Setting

+ server\_setting(string v)  
+ ~server\_setting()  
+ change(string v)  
+ get()  
# has\_server\_setting\_changed()  
# get\_server\_setting()  
# change\_server\_setting(string v)

Notify

+ notify(string init\_value)  
+ ~notify()  
+ change(string curr\_value)  
+ register\_observer(observer \*)  
# notify\_observers()

GUI Setting

+ gui\_setting(string value)  
+ ~gui\_setting()  
+ change(string v)  
+ get()