

---

Subject: Re: Modal vs non-modal window  
Posted by [mirek](#) on Sat, 06 Oct 2007 16:07:56 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

mrjt wrote on Fri, 05 October 2007 06:09

```
class DataEntryWindow : public TopWindow
```

```
{
```

```
public:
```

```
    typedef DataEntryWindow CLASSNAME;
```

```
    // Sets the table name to update
```

```
    DataEntryWindow &SetTableRecord(String table, int recordid);
```

```
    // Links a DB field with a Ctrl
```

```
    DataEntryWindow &SetDataSource(Ctrl &source, String field, Value initial_value, int datatype);
```

```
    OnOK()          { if (Accept()) { Commit(); OnCancelClose(); } }
```

```
    OnCancelClose() { delete this; }
```

```
private:
```

```
    Commit(); // Builds SQL string using datasources and commits it to DB
```

```
};
```

could form the basis of a data entry window that can be launched from a single function and then forgotten about:

```
void NewEmployee()
```

```
{
```

```
    WithEmployeeLayout<DataEntryWindow> *wnd = new
```

```
WithEmployeeLayout<DataEntryWindow>();
```

```
    //Set tablename, datasources etc
```

```
    wnd->Open(this);
```

```
}
```

Not that this is only a good idea if you have unlimited number of peer windows (like in UWord), preferably even main peer windows.

For modeless things, usually it is better to just make those subdialogs a member variables (no pointers); to open and close them as needed, no news and deletes involved. This also has the advantage that the content of dialog is accessible in the containing class, so there is often no need to transfer widget data to variables and back...

Mirek

---