
Subject: Re: Proposed change to U++ to allow owning children.

Posted by [Lance](#) on Sun, 20 Mar 2011 13:28:08 GMT

[View Forum Message](#) <> [Reply to Message](#)

It just come to me that we can change the interface slightly to make it unnecessary/impossible for end users to use new directly.

Add a template member function to the Ctrl class

```
class Ctrl...
```

```
{
```

```
...
```

```
    bool owned : 1;
```

```
....
```

```
protected:
```

```
    bool IsOwned()const{ return owned; }
```

```
    Ctrl& Owned(bool v){ owned=v; return *this; }
```

```
public:
```

```
    template <typename ChildType>
```

```
    ChildType& AddOwned()
```

```
{
```

```
    ChildType* p=new ChildType();
```

```
    p->Owned();
```

```
    (void)Add(p);
```

```
    return *p;
```

```
}
```

```
template <typename ChildType, typename T>
```

```
ChildType& AddOwned(T& t)
```

```
{
```

```
    ChildType* p=new ChildType(t);
```

```
    p->Owned();
```

```
    (void)Add(p);
```

```
    return *p;
```

```
}
```

```
// return ChildType so that user can further set its
```

```
// properties.
```

```
// The IsOwned and Owned functions will be demoted to protected
```

```
// so that they are available only to library developers
```

```
//
```

```
// whether a child is owned is a one time decision.
```

```
// an owned child will not be able to be reverted to unowned
```

```
// by end user (without derive from the Ctrl class or its
```

```
// derivatives), but he/she is free to change parents for it
//
// Unless the end user uses some hackish practice,
// library developers can be assured the owned flag
// is reliable and predictable.
};
```
