Subject: Re: NEW: Dispatcher (templateable dispatcher helper for MVC pattern and more)
Posted by kohait00 on Mon, 12 Jul 2010 07:51:59 GMT
View Forum Message <> Reply to Message

hey koldo..

it's a veeeery simple 1-to-many messages forwarder (dispatcher). the code doesnt do much more than

```
template<class T>
void Dispatcher<T>::DoDispatch(const T & o, unsigned param) const
{
 for(int i = 0; i < B::GetCount(); i++)
 {
  Dispatchable<T> * dest = B::operator[](i);
  if(dest)
   dest->Dispatch(o, param);
 }
}
```

where B:: is the base class container

the MVC pattern often times uses Dispatchers to pass around data, i.e. when a mode has many views, some genrated data needs to be displayed in each of the views, so it needs to be forwarded / dispatched.

imagine i.e your app generates data, that you want to be forwarded or displayed in more than one place. on multiple tabs or sth.. or some controller classes that listen for some kind of info to come in and generate more action.

it's actually a 'data link', 1-to-many.

for sure, the examples provided could be solved by just hooking up all the Ctrl's using callbacks. but this is too static, if zou imageine, you need to specify somehow dynamicly, who is about to receive stuff, or add or remove receivers, than your out. the hook up in code remains unchangable.

genrally, i think this Dispatcher thing also could be done using callbacks (a Vector<Callback>), but they tend not to enforce you to stick to some API, so using an interface class (Dispatchable<T>) makes sure you dont forget to implement the right function to receive stuff.

But i might try to provide a callback implementation as well, just for completeness.

i hope this was a bit more clear.