Subject: Re: NEW: generic Toupel grouper
Posted by kohait00 on Sun, 15 Aug 2010 07:16:00 GMT
View Forum Message <> Reply to Message

Quote:
Otherwise I would prefer Single, Double, etc.

there, similarities to data types wouled be to close (Double) and misleading. Solo is not usefull anyway (wraping one T is .. nonsense), it should start with Duo, Trio, etc.. to set Apart from the One (and maybe later Two, Three)

Quote:
after criticizing kohaits proposals

dont worry, it's part of development, and didnt feel criticised . i am always looking forward to meeting better ideas..

to the above implementation:

the Get<1>() option (idea from boost?) is a cool trick, but IMHO of little use because it's not a runtime check, but a compile time definition, thus u.a, u.b, u.c is much simpler and clearer in that sense, and less to type anyway. i mean, in terms of compile time specialisation u.Get<1> is same as u.a, you have to provide the index at compiletime, so you know which type.

the Value operator[](int i) is a good idea though. to wrap / unwrap in value (boxing / unboxing is used in C# and others, though there in different context, as base class object).

having Duo, Trio, etc is, as you pointed out, more or less useless, even if it's better to read so i added a 5th T and deaulted past second T. (i'd rater use EmptyClass, but there is no Value(const EmptyClass &) for it, so i used Nuller. might be usefull to have an EmptyClass Value as well?)

so here comes another option.

```
template<class T1, class T2, class T3=Nuller, class T4=Nuller, class T5=Nuller>
class Tupel
{
public:
 Tupel(const T1 & _a, const T2 & _b, const T3 & _c, const T4 & _d, const T5 & _e)
  : a(_a), b(_b), c(_c), d(_d), e(_e) {}
 Tupel() {}

 Value operator[](int i) {
  switch(i) {
   case 1: return Value(a);
   case 2: return Value(b);
   case 3: return Value(c);
   case 4: return Value(d);
   case 5: return Value(e);
   default:
   case 0: ASSERT(0); return Value();
  }
  return Value(); //dummy
 }

 T1 a; T2 b; T3 c; T4 d; T5 e;
};
```

what about this one? it provides the simplicity desired and has the wrapper.