Subject: NEW: generic Toupel grouper
Posted by kohait00 on Thu, 12 Aug 2010 14:03:51 GMT
View Forum Message <> Reply to Message

sometimes its cool to group things together in an easy manner, especially when wroking with the template containers. to build a grouping container over and over again each time for different purposes, just for classes that would have the parts public anyway,is odd.

say you want to have a vector that simply contains a 2-tupel of values. maybe some measurements consisting of two values at once, or anything that logically belongs together but does not need any abstraction or scope hiding. a normal container cant do it.

Vector<float, float> vi; //would resemble VectorMap.

you just need to quickly define and access 2 things at once in a container and want to have sth like

```
Vector<Duo<int, int> > vi;
...
vi[i].t1 = 123;
vi[i].t2 = 234;
```

this looks good and is handy.

so here comes some helpers to do that

```
template<class T>
class Solo
{
public:
 typedef Solo<T> CLASSNAME;
 Solo(const T & _t) : t(_t) {}
 Solo() {}

 operator T & () { return t; }
 operator const T & () const { return t; }

 T t;
};

template<class T1, class T2>
class Duo
{
public:
 typedef Duo<T1, T2> CLASSNAME;
```

```
 Duo(const T1 & _t1, const T2 & _t2) : t1(_t1), t2(_t2) {}
 Duo() {}

 operator T1 & () { return t1; }
 operator const T1 & () const { return t1; }
 operator T2 & () { return t2; }
 operator const T2 & () const { return t2; }

 T1 t1;
 T2 t2;
};

template<class T1, class T2, class T3>
class Trio
{
public:
 typedef Trio<T1, T2, T3> CLASSNAME;
 Trio(const T1 & _t1, const T2 & _t2, const T3 & _t3) : t1(_t1), t2(_t2), t3(_t3) {}
 Trio() {}

 operator T1 & () { return t1; }
 operator const T1 & () const { return t1; }
 operator T2 & () { return t2; }
 operator const T2 & () const { return t2; }
 operator T3 & () { return t3; }
 operator const T3 & () const { return t3; }

 T1 t1;
 T2 t2;
 T3 t3;
};

template<class T1, class T2, class T3, class T4>
class Quartett
{
public:
 typedef Quartett<T1, T2, T3, T4> CLASSNAME;
 Quartett(const T1 & _t1, const T2 & _t2, const T3 & _t3, const T4 & _t4) : t1(_t1), t2(_t2), t3(_t3),
t4(_t4) {}
 Quartett() {}

 operator T1 & () { return t1; }
 operator const T1 & () const { return t1; }
 operator T2 & () { return t2; }
 operator const T2 & () const { return t2; }
 operator T3 & () { return t3; }
 operator const T3 & () const { return t3; }
 operator T4 & () { return t4; }
```

```
 operator const T4 & () const { return t4; }

 T1 t1;
 T2 t2;
 T3 t3;
 T4 t4;
};
```

maybe they can go to Others.h..
