Subject: Re: Arrys vs Vectors
Posted by kohait00 on Tue, 21 Dec 2010 11:36:22 GMT
View Forum Message <> Reply to Message

i'd recommend to use Vector here, see Core/Gtypes.h for Size and Point implementations, they also are templates, and are only Moveable<>, no need for you to MoveableAndDeepCopyOption<> them. it is a lightweight pointer class as i can see. it is best copied. i suppose T is double or float..

picked is misleading... Upp has own usage for pick stuff, think of a different name for it.

also, use the initializer list in constructors, and the implicit logic of classes (aka implicit copy constructor).

think of what you really need to have getters/setters to. better make the members public in this case.

```
template<class T>
class Point3D : Moveable<Point3D<T> >
{
public:
 T x,y,z,w;
 unsigned int id,layer;
 bool taken;
public:
 Point3D ()
  : x(T())
  , y(T())
  , z(T())
  , w((T)1.0)
  , id(0)
  , layer(-1)
  , taken(0)
 {}

 Point3D (const T& a, const T& b, const T& c, unsigned int e, bool f, unsigned int g, const T& d=(T)1.0)
  : x(a)
  , y(b)
  , z(c)
  , w(d)
  , id(e)
  , layer(g)
  , taken(f)
 {}

 //use implicit copy constructor
 //this one should probaly be Point3D<T> operator*(const T& o) const, since yours is modifying
object
 void operator*(const T& o){
  x*=o;
  y*=o;
  z*=o;
  w*=o;
 }

 Point3D<T> operator+(const Point3D<T> &o){
  return Point3D<T>(x+o.x(),y+o.y(),z+o.z(),w+o.w());
 }

 void Serialize(Stream& s){
  s%x%y%z%w%id%taken%layer;
 }

 String ToString()const{
  String s;
  s<<"x:"<<x<<","<<"y:"<<y<<","<<"z:"<<z<<","<<"w:"<<w<<","<<"id:"<<id<<","<<"taken:"<<taken<
<", "<<"layer:"<<layer;
  return s;
 }
};


GUI_APP_MAIN
{
 Vector<Point3D<double> > v;
 v.Add();

 Vector<Point3D<double> > vw;
 vw <<= v;

 Point3D<double> p;
 p = v[0];
}
```