Subject: Re: Value: no deep copy support?
Posted by mirek on Wed, 20 Oct 2010 21:16:59 GMT
View Forum Message <> Reply to Message

kohait00 wrote on Wed, 20 October 2010 10:55back to the topic:

offering a somehow optional way to be able to deepcopy the content of a Value would make it an appealing refcount based container for polymorphic stuff as well. Array<T> still needs some info about type.
Value doesnt.

i dont mean copy-on-write or something, which would be implicit, so the user does not get an idea of behaviour. he still can expect Value to be one (of many) references to the content.

but wanting to internally clone it without knowing type or anything about it would be great as well. enabling the user to 'edit' later 2 distinct copies in an abstract handler.

Nonsense. You can NEVER 'edit' Value.

Quote:
it could maybe rely on DeepCopyNew somehow. but this might need to introduce a Copy function in the Value::Void interface, returning Void*.

virtual Void* Copy() const { return new Void(); }

the RawValueRep<T> would add sth like:

virtual RawValueRep* Copy() const { return new RawValueRep(*this); }

//the value could then be MoveableAndDeepCopyOption

```
Value(const Value& v, int)
{
 ptr = v.ptr->Copy();
 //ptr->Retain(); //we are first owner
}
```

this would enable the approach described above

```
Value v = (int)123;
Value v2(v,0); //indicate to use the deep copy to duplicate the values

//v and v2 are now unrelated anymore..can be edited/overwritten individually
//using const_cast..clone has been produced without knowing content type.
```

Value v3;
vs <<= v; //same as with constructor, more intuitive, as from Containers :)

these are only very basic ideas, maybe it is of interest. i'll try to play with it a bit..maybe i can find sth usefull.

Sorry, but this is just misunderstanding. There is zero value in deep copy for Value.

Please think about Value like it is an 'int' - with "assignment" time always O(1).

That implies Value is principally immutable - you can never change what is inside. Only assign another value.
