Subject: Changing the pick semantics notation
Posted by mirek on Fri, 07 Mar 2014 18:01:27 GMT
View Forum Message <> Reply to Message

For 12+ years, '=' for pick-types meant 'pick' (destroying source), with implementation that relies on

```
#define pick_ const
```

to make it usable for function return values. <<= and T(const T&, int) were used to provide optional 'deep copy' values.

This served well, but it was not really optimal and might have scared some at first. And while in practice much less error-prone that it might look, there was still chance of occasional errors.

Meanwhile, we got C++11 with its rvalues. It would really be nice to replace pick_ with &&. But things are not that simple, OTOH they provide chances for improving things even more.

One thing is clear: we still do not want containers to have implicit deep copy (like STL has).

Now, after a day of experimenting I would like to present/propose new pick notation:

'=' alone is now disallowed for pick-types. Instead, we will write:

```
a = pick(b); // pick copy
```

or

```
a = clone(b); // deep copy
```

(except when we are assigning temporary value - then by the logic it is clear that we want 'pick').

operator<<= is thus deprecated and deep copy constructor can be written as

```
A a = clone(b)
```

When compiling with C++11, this notation becomes enforced.

When compiling with C++03, old things still work without a change and this new notation is optional.

I have already completely changed U++ to support new notation and C++11 compilation (to the point of compiling theide). This is committed in upp/branches/cpp11.

Please comment and/or vote on this change...

New pick notation(total votes: 12)

I support this 12/(100%)
I do not like this 0/(0%)
I do not care 0/(0%)