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%)

Subject: Re: Changing the pick semantics notation
Posted by zsolt on Fri, 07 Mar 2014 18:45:43 GMT
View Forum Message <> Reply to Message

The current pick behavior was very extreme and hard to explain thing for most of the programmers I know. I think, this change should help a lot in U++'s popularity.

Subject: Re: Changing the pick semantics notation
Posted by koldo on Sat, 08 Mar 2014 18:05:02 GMT
View Forum Message <> Reply to Message

Very reasonable. My vote is yes.

In addition (if possible) I would like to have some kind of #define to force errors when an = or a <<= is found, to help code updating.

Subject: Re: Changing the pick semantics notation
Posted by nlneilson on Sat, 08 Mar 2014 19:45:16 GMT
View Forum Message <> Reply to Message

Vote for the change.

The only down side to the change would be updating current code in existing apps. A few posts covering problems should solve things.

Picking has worked good for me even on menu pull downs in tablet mode with a pen.

Subject: Re: Changing the pick semantics notation
Posted by dolik.rce on Sat, 08 Mar 2014 20:41:13 GMT

I cautiously support this

The change seems to touch mostly everything in U++, so I'd be glad if it was very thoroughly tested before merging into trunk. Also some performance checks (on real life applications) would be a nice idea.

If I understand it correctly, it seems to smoothen some rough edges around current pick behavior, so it is a good thing. What I like most is the fact that it should prevent accidental picks, which happen to me quite often

Best regards,
Honza