Subject: Re: [PROPOSAL]: VarArgs class for U++ (va_ macros replacement, in U++ stlye)
Posted by Oblivion on Sun, 21 Feb 2016 16:23:44 GMT
View Forum Message <> Reply to Message

Novo wrote on Sun, 21 February 2016 17:11Hi Oblivion,

You've added virtual destructor, which is IMHO not necessary. It will just add an extra pointer to your data structure.
I'm just curious, why you cannot use plain Vector<Any>?

Hello Novo,

Thank you for your comment.

Yes you are right, it is unnecessary in this case; a relic from its first version I've forgotten to delete. Thanks for pointing out! :)

As to your question, well, there are mainly two reasons why I do not want to use plain Vector<Any>:

First, as you can see, VarArgs is a convenience class/wrapper. It simply hides the unnecessary interface/clutter of Vector, which is huge, and allows a rather goal oriented, limited interface.
This brings me to the second reason: semantic considerations. Think of VarArgs as something like the Id class, which is actually a String, in the Core package. But unlike Id class,
VarArgs has a specific set of methods, and announces its purpose: It stores arguments for methods, or functions in a dynamic manner. Nothing less, nothing more.

From a newcomers perspective, when all you want to do is to use variadic functions, whose macros are not very safe and trying to find an equivalent for them in U++, where would you look at first?
Would you read through the whole documentation, and try to figure out which combinations will work? (This is what ought to be, but we are living in a world where time is precious.) It would be time consuming to read through the whole documentation of Vector, which is rather huge, and Any. Having specific classes under your reach for widely used programming techniques is, I believe, the goal of rapid application development frameworks, and U++ usually excels at this. Aside from that, it improves the source code readablity.

Regards,
Oblivion