Code Review: ControlExec - February 11, 2016

Item	File	Description	Consensus	Who	Status	Notes
1	ControlExec.cpp	error checking: Return of pointers from various methods (e.g. GetParamsInstance(), GetVisualizer() not checked for NULL in numerous methods, resulting in abort. No error logged where return value is checked. Add error handling or asserts as appropriate()	y		open	This is still not done. Numerous instances where methods that return pointers are called and the return value is not checked against NULL.
2	ControlExec.cpp	Documentation for ActivateRender() unclear. What happens when parameter 'instance' is not known? Where does 'instance' come from? What are possible values for parameter 'type'? Make use of doxygen \sa (see also)	y		Done	
3	ControlExec.cpp	Shouldn't ControlExec be derived from MyBase to make use of error reporting?	y		Done	
3a	ControlExec.cpp	Many methods return error conditions, but don't log error messages. E.g. SetCurrentParamsInstance(), ActivateRender()	y		Done	
4	ControlExec.h	Documentation for RemoveParams() implies that all instance identifiers will become invalid after calling this method. This appears to have the potential for nasty side effects. Why not simply keep instance params unique and not force them to be contiguous.	y		Done	Such a change to instance identifiers would require lots of code changes. Instead, since the identifiers are only manipulated by the InstanceParams, the appropriate ParamsMgr methods were made private and the InstanceParams made a friend of ParamsMgr.
5	ControlExec.h	Params management interface seems unnecessarily complex. Hopefully can be simplified or more extensively documented. There are over 20 methods for Params access. Not clear when to use what.	y		closed	Needs further review and discussion. See example code. Params management methods have been moved to ParamsMgr class.
6	ControlExec.cpp	Memory leak in ControlExec::LoadData(). _dataStatus not freed if already allocated.	y		Fixed	an:_dataStatus is deleted in destructor jc: There is nothing to prevent LoadData() from being called twice on same instance.
7	ControlExec.cpp	Hardcoded window with and height in ControlExec::NewVisualizer()?	y		Done	Removed. width and height are set in resizeViz.
8	ControlExec.cpp	How is FindInstanceIndex() used? It takes a RenderParams ptr as a parameter, but there appears to be now way to obtain a RenderParams ptr from the ControlExec			Done	Review params api first. This is currently only used to perform enable /disable operations on sets of RenderParams, and the instance index is needed for that.
9	ControlExec.cpp	SetErrorHandler() is a no-op. Can it be deleted?	y		Done	
10	ControlExec.cpp	ValidateParams is a no-op. Can it be deleted?	y		Done	
11	ControlExec.cpp	Undo and Redo return Params ptrs. Which Params? Can anything useful be done these pointers if the caller doesn't know which Params instances or types that they refer to?			Closed	Need to evaluate further. May not be needed. This Params is currently only used to determine whether a render is required, so its type and instance is not needed. If we identify a need for more information during undo/redo then we should modify the API at that time.
12	ControlExec.h	Is there a reason that the ControlExec is derived from ParsedXml class? Would composition be more appropriate?	y		Done	do if easy
13	ControlExec.cpp	Are the log file facilities (OpenLogFile(), etc) needed? What are they for?	N/A			
14	ControlExec.cpp	Shouldn't be a static class (static members and methods) without justification.			Done	
15	Params	Explore making static params class a true class (aka ParamsMgr), a pointer of which can be handed to routers, etc. so that they don't need access to controlexec.			Done	

16	ControlExec.cpp	Resource mgt: • ParseMgr allocated from heap and never freed. • GetCommandText() allocates string that can never be freed • Test for NULL before calling delete (e.g. delete _sessionParser, delete in destroyParams())	y		Done	
17	ControlExec.cpp	ControlExec should be hidden from renderers, Visualizer, TextObject, etc.. If these classes need access to resources owned by the ControlExec (e.g DataMgr) provide only as much access as needed. I.e. Pass a handle to the DataMgr to the classes that need access.	y		Done	This has been removed from all the Render classes except still kept as a private variable in the Renderer class, as the control exec is needed when performing Undo/Redo of renderer enabling.
18	ControlExec.h	Use 'const' qualifiers for Get accessor methods as appropriate.	y		Done	