Code review: Isoline Renderer - January 25, 2016

Status: Open

Item	File	Description	Consensus	Who	Status	Notes
1	isolineparams.cpp	IsolineParams::validateIsoControls() accesses every 2D and 3D variable in the data set whether they are needed or not	y	Alan	Done	Variable bounds are obtained as needed.
2	isolineparams.cpp	replace new3DIsoControls[i]->setVarNum(i) with setVarName(). Don't use variable indices when their meaning can change from data set to data set, session to session.	y	Alan	Done	Alan did evaluate.
3	isolineparams.cpp	Does there need to be a distinction made between 2D and 3D IsoControls? A lot of redundant code could be collapsed if they were not treated separately.	y	Alan	Done	
4	isolineparams.cpp	Type 1 validation in IsolineParams::validateIsoControls() appears to be reconciling different variables by using their integer offset in the data set. Shouldn't we be using the variables names, not the offset? I.e. The variable "temp" probably has the same meaning in two different data sets, but may have different offsets.	?	Alan	Done	Alan will evaluate.
5	isolinerenderer.cpp	Move responsibility for cache management inside of isolinerender.cpp. Clients of the renderer classes shouldn't have to know what changes necessitate cache invalidation. They shouldn't even have to know that a renderer uses a cache. Renderers that maintain a cache should retain parameter values for parameters that would force an update of the cache.	y	Alan	Done	
6	isolinerenderer.cpp	Renderer's should be subordinate to ControlExec. I.e. They shouldn't have access to it.	y	Alan	open	discuss ion context of control exec review
7	isolineeventrouter.cpp	Memory leaks: GLIsolineWindow, Histo,	y	Alan	Done	Qt is freeing the GLIsolineWindow memory, but not Histo.
8	isolineeventrouter.cpp	Can _dataMgr ever be NULL? Should the render event routers be enabled when data aren't loaded?	y	Alan	Done	Change tests for existence to asserts, or remove unnecessary tests.
9	isolineeventrouter.cpp	isolineeventrouter is really long and complex (~1200 lines), but the GUI it supports is fairly typical of all of the tabs. What can be done to simplify so that other render event routers aren't as long and complex?	y	Alan	Done	Reorganize the cpp file to place common elements together (e.g. by subtabs) and document them a such
10	isovalueeditor.cpp	Lots of memory leaks.	n			All the QObjects in the editor get deleted when their parent is deleted. Read the Qt documentation on QObjects.
11	isolinerender.cpp	Unnecessary ControlExec access. E.g. uses ControlExec to get DataMgr inside buildLineCache(), which is passed DataMgr as an argument.	y		Done	
12	isolinerender.cpp	Make invalidateLineCache(), buildLineCache(), etc. private	y		Done	
13	isolineeventrouter.cpp, isolinerenderer.cpp	Check for NULL before deleting pointers (e.g. in destructor)	y		Done	

14	isolineeventrouter.cpp	Remove ForceRenderer requirement. Specialize QWidget::Event()??	y		Done	ForceRender has been removed. Now various Qt events will trigger rerender. However transfer function edits do not necessarily trigger a Qt event so these now cause a rerender upon completion.
15	glisolinewindow.cpp	Rendering code duplicates isolinerenderer. Have isolinerenderer perform rendering for tab and eliminate/reduce glisolinewindow.cpp	?		postpone	
16	isolineparams.cpp	Resource mgt: memory allocated in validateIsoControls is never freed	n			These are ParamsBase instances that get destroyed with the Root ParamNode.