

Known Bugs/Limitations in Release 2.1

I don't really have a bug tracking/feature tracking methodology, so there aren't numbers assigned or anything like that. If anybody has any good suggestions, I'll take them under advisement. These are the major bugs/unimplemented-features-that-look-like-they-should-be-there that I can think of...

WWTCLKit

- **I alt-drag a slider in IB, and the colors of the slider suddenly get weird.**
I know; I managed to break something when I added colors to my sliders. I've looked, and although I think it probably is my bug, I also think that the SliderCell stuff is rather undocumented (especially with regard to the automaticizing that happens in IB), and it's going to be hard to fix. I'm hoping it will annoy someone enough that they'll take the time and track it down for me.
- **WWTextFields seem to ignore justification - they're always left justified.**
Again, I'm not sure what broke here. It's not a huge deal for me, but I would

like this fixed...

WW3DKit

- **I read in a rib file with patches in it and the bounding box seems wrong.** Does the rib file have Patches in it? Do the patches have a Basis other than Bezier? Right now I only use the vertices of a patch to calculate the convex hull of the mesh. This is really way too large (usually) for ones like b-spline basis and just plain wrong for ones like catmull-rom. I still haven't completely implemented bounding boxes for patches with bases other than Bezier. As it is, I'm not totally sure it's right. It's complicated code, and right now, not terribly high on my list.
- **I read in a rib file and it looks wrong.** Could happen. Bring up the WW3DWell's Control Panel and bring up the RIB panel (press "Rib" on the far left). Now select "treat TransformBegin the same as AttributeBegin". This means that when the WVEveParser sees a TransformBegin, it does the same thing it does when it sees an AttributeBegin (duh!). In other words, instead of alloc'ing up a RIBTransformBegin object, it allocs up a new WW3DShape, makes it a child of the current shape, and makes that shape the current one. This also means that TransformEnd now does the same as AttributeEnd, which means that it will pop the current shape off the stack. If it still looks wrong, send me mail (and the RIB file). Also, make sure that

the RIB file is all ASCII. WavesWorld can't deal with binary RIB files. Another thing to check on is to see if

- **I generated a rib file in (some program) and when I dropped it into the WW3DWell, the camera was in a totally different position.** Right now, I throw away any camera location info and stick the object 5 units in front of the camera. I plan to have support of holding on to camera info, I just haven't needed it, and haven't worried about it.
- **What is the "interp" panel of the WWRIBWell's control panel for?** This lets you type tcl code directly at the two tcl interps embedded in the WWRIBWell. Type "info commands" in each of them and see what happens.
- **What are the files /tmp/foonnn.rib, /tmp/foonnn.tiff, and /tmp/tmpnnn.eve?** These are tmp files that the WW3DWell should clean up, but sometimes doesn't. Sorry about that... Need to switch them over to memory streams. Potentially, you could get two rib wells scribbling on the same files, but this hasn't been an issue for me (if you've got multiple rib wells doing similar things at the same time, I'm wicked envious of the fast machine you've got...). I made them write out to files instead of streams for debugging reasons (it's easier to look at a file than a stream if something gets borked...), and still I'm still debugging this stuff...
- **When I drop a buggy eve file in the WW3DWell, I get the error panel several times.** Yea, I know. I need to trace through all the points where an

error occur and decide what's the highest level of evaluation that things always go through and weed out the redundancies. I really should have an exception mechanism that lets me toss out when I get an error. In the meantime, I felt it was better to get too many error messages than miss one...