Castor3D - Évolution #1092

Uniform Buffer Objects

12/21/2010 10:03 AM - dragonjoker59

Status:	Fermé	**Start date:**	12/21/2010
Priority:	Bas	**Due date:**	
Assignee:	dragonjoker59	**% Done:**	100%
Category:	OpenGL	**Estimated time:**	0.00 hour
Target version:	0.7.0	**Spent time:**	12.00 hours
Lien forum:			

Description

Ajouter la gestion des Uniform Buffer Objects

History

#1 - 12/21/2010 10:07 AM - dragonjoker59

La prise en compte est faite, mais apparemment seul un bloc est pris correctement en compte, les autres sont ignorés.
Selon moi, c'est dû aux drivers qui ne prennent pas encore ça en compte (GeForce 9500 GT, Driver v8.17.11.9621).
A tester sur d'autres configs.

#2 - 01/12/2012 04:11 PM - dragonjoker59

- Project changed from GLRenderSystem to Castor3D

- Parent task deleted (#1090)

#3 - 01/12/2012 04:12 PM - dragonjoker59

- Category set to OpenGL

#4 - 04/20/2013 01:59 PM - dragonjoker59

- Status changed from Assigné to Résolu

- % Done changed from 90 to 100

Grâce au à l'implémentation des FrameVariableBuffer, j'ai pu intégrer cette notion au niveau Castor3D, ce qui me permettra de le mettre en place pour Direct3D aussi, sachant que je vais abandonner Direct3D 9 au profit de Direct3D 11 (avec donc les constants buffers).
En fait tous les blocs sont pris en compte, mais il fallait renseigner le paramètre "index" des fonctions glUniformBinding et glBufferBase. Je le fais via un index statique que j'incrémente à chaque initialisation de variable buffer.
La partie plugin de FrameVariableBuffer maîtrise l'initialisation, elle ne fait appel à l'initialisation des variables que s'il est impossible de passer par l'initialisation du buffer (UBO indisponible, par exemple).
J'ai supprimé les fichier GlUniformBuffer.hpp/inl/cpp car au final c'est FrameVariableBuffer qui s'en charge, pas besoin de classe supplémentaire.

#5 - 12/12/2014 01:44 PM - dragonjoker59

- Status changed from Résolu to Fermé

#6 - 09/28/2015 02:08 AM - dragonjoker59

- Target version set to 0.7.0