Notes:

Please read the notes as well for other ideas

Interface between Engine core and Objects. Notice no Use of D3D11 in this class

Utility class used to use unsigned int instead of string

Engine Core
The connection between what mesh uses what animation set is known here.

Possible Upgrades:

*Animation set infomration wrote to a texture

*Interpolation between the bone offset information instead of the world

*Temp vertex buffer for models needing more than one pass

*Dual quaternions (industery standard)

*More than two interpolators

*Blend only half the skeleton with another animation
*Instanced animations

Engine Core

One per animated model type. An animation my be used for multiple models as well. Notice there is no mesh animation here. That should not be needed for actually animating the bones.

One per Rig

Better to call this a skeleton and use a transform class instead of just the inverse bind pose. Also names of all the bones would go in the skeleton

Think if you want to have Bones[key frame index][bone index] or Bones[bone_index][key frame index]

One per bone, per time stamp

The choice to do all bones marked for one keyframe was on purpose. See possible upgrade notes. It is acceptable for a keyframe to have the bone information for only the important key frames.

This updates the blender then sends the resulting bone offset matrix array to the render node

Can be a normal game object with an animation component

This classes job is to go from one animation to another animation and set the bone offset matrix

One per game object

A further abstraction from the blender can be made to use more blends, state machines or any other number of combations

One for every bone at every time.

This is assumed to tie the world. You may want to add Joint names, Local and bone offset information as well

Better to convert this to a quaternion, scale and position

Per Object data Allows multiple objects using the same aniation to look different.

This class's job is to get the world position of each bone for one animation

---

AnimatedRenderNode

Attributes
+ BoneOffset Matrix : XMFLOAT4X4 *
+ IndexBuffer : Integer
+ Material : Integer
+ PerTypeInfo : Integer
+ Vertex Buffer : Integer
+ WVP : XMFLOAT4X4

Operations
+ ...()

HashString

Attributes
+ ...
+ LookUp : std::map<std::string, Integer>

Operations
+ ...()
+ GetKey(std::string) : Integer

Renderer

Attributes
+ RenderNodes : Ani...

Operations

ResourceManager

Attributes
+ ...
+ Animations : std::map<Integer, AnimationSet>
+ Rigs : std::map<Integer, BindPose *>

Operations
+ ...()
+ GetAnimation(_setKey : Integer, _animKey : Integer, _defaultBlendTime : float &) : const ModelingPr...
+ GetBindPose(_setKey : Integer) : const XMFLOAT4X4 *
+ LoadAnimationSet(std::string _key) : Integer _key

Animation Set

Attributes
+ Animations : std::vector<Animation>
+ bindPose : const Bind Pose *
+ BlendTimes : std::vector<std::vector<BlendInfo> >
+ defaultAnim : Integer

Operations
+ AddAnimation(_file : FILE *) : Boolean
+ AddBindPose(const BindPose *) : Boolean
+ GetAnimation(_key : Integer) : const Animation *
+ GetBlendInfo(AnimationFrom : Integer, AnimationTo : Integer...
+ GetDefaultAnimation() : const Animation *

Bind Pose

Attributes
+ InvBindPose : XMFLOAT4X4[num_bones]
+ num_Bones : Integer

Operations
+ GetBindPose() : const XMFLOAT4X4 *
+ GetNumBones() : Integer
+ Init(_num : Integer, XMFLOAT4X4 *)

Animation

Attributes
+ AnimType : ANIM_TYPE
+ Bones : KeyFrames[num_KeyFrames]
+ num_KeyFrames : Integer
+ TotalTime : float

Operations
+ GetFrame(_index : Integer) : const Bone*
+ GetNumBones() : Integer
+ Init(_num : Integer, _type : ANIM_TYPE)

KeyFrame

Attributes
+ num_bones : Integer
+ Time : float
+ World : Bone[num_bones]

Operations

BlendInfo

Attributes
+ BlendStyle : BLEND_TYPE
+ TotalBlendTime : float

Operations

**«enumeration»
BLEND_TYPE**

Literals
CUR_GO_NEXT_GO
CUR_GO_NEXT_STOP
CUR_STOP_NEXT_GO
CUR_STOP_NEXT_STOP
DEFAULT
POP_NEXT

**«enumeration»
ANIM_TYPE**

Literals
LOOP
RETURN_DEFUALT
RETURN_LAST
RUN_ONCE

Blender

Attributes
+ AnimationContainer : ModelingProject1::Animation Set *
+ Blend : ModelingProject1::BlendInfo *
+ BoneOffsetArray : std::vector<XMFLOAT4X4>
+ currAnim : Interpolator *
+ nextAnim : Interpolator *

Operations
+ GetSkinningMatrix()
+ SetAnimSet(_key : std::string)
+ SetNextAnim(_type : BLEND_TYPE, _animKey : Integer) : Boolean
+ Update(float _time)

Interpolator

Attributes
+ animation : const Animation *
+ CurrBlendTime
+ currFrame
+ TotalBlendTime
+ World : std::vector<Bones>

Operations
+ GetCurrentWorld(_num : int &) : Bone *
+ GetTime() : float
+ SetAnimation(const Animation *)
+ SetTime(_time : float)
+ Update(float _time) : ANIM_TYPE

Bone

Attributes
+ World : Float4x4

Operations

AnimatedGameObject

Attributes
+ ...
+ animation : Blender
+ RenderNode : RenderNode *

Operations
+ ...()