MATERIAL AND TEXTURE ANIMATION WITH NIFSKOPE:

PART 4: CONTROLLER MANAGER, SEQUENCE AND SCRIPT

A tutorial from Pixelhate. 2014.

Contact Information: Pm Pixelhate at http://forums.nexusmods.com/index.php?showuser=1205226

This guide is the result of compiling information on the use of NifSkope by reading tutorials (some quite outdated) and Wikis, tracking help posts on forums or by trying to figure things out by myself. This work was possible thanks to too many people to name them all. Some of their words where just copy/pasted here, as things where explained better than I will ever be able to do.
I am forever grateful for their generous act of sharing knowledge.

So, if this guide is ever useful to you, it is thanks to them.

Please, see credits and links at the end of this document.

This part comes from a package with 4 parts in Pdf or Doc format and with a series of meshes and textures, referenced in the following pages. A series of flow charts images summarize the steps for each animation type.

If you received this tutorial without these resources, please visit http://www.nexusmods.com/fallout3/ and download the full package.

Tutorial is in four parts:
Part 1 an Introduction
Part 2 about Material Animations
Part 3 covering Textures Animations
Part 4 This part which deal with Controller Manager, Sequence and Script.

TABLE OF CONTENT

Thank you to TrickyVein for his inspiring tutorial
http://forums.nexusmods.com/index.php?/topic/984792-tutorial-working-with-the-nicontrollermanager/

This tutorial is focused on Static or Havoked object. No skins, armours or weapons have been tested.

All information shared here is meant for Fallout 3, but most probably will work for FNV as well.

Two meshes are provided: a working camera monitor (DemoCameraMonitor.nif) and a “blank” monitor (DemoTerminalnterface01.nif).
The camera monitor contains multiple Material and Texture animations in two sequences that can be triggered by script. It is given for information and inspiration purpose.

The blank monitor is made ready for you to build the animation while following the tutorial.

The nif has three parts that can be animated separately: the Power Button, the Screen and the Glare.

In this demonstration, we will set a simple animation Translate for the Screen and Emissive for the Power Button.
Hopefully, you will be able to animate the other parts by yourself later.

When OFF (Forward) the screen will display one part of the texture fixedly, the button will be unlit.

When ON (Backward) the screen will display a panning effect on another part of the texture, giving the illusion of a security camera recording, the button will be glowing.

Both sequences will use the same animation block structure. In order to save some work the following steps are planned:

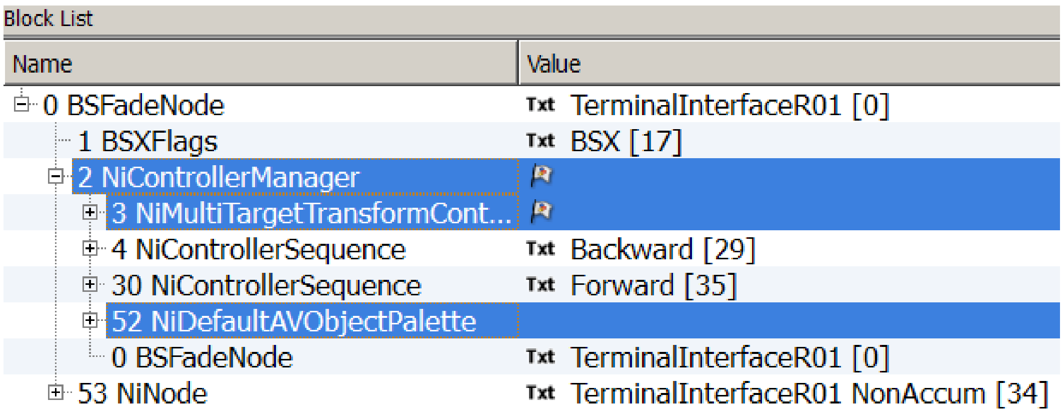
- Preparatory work.
- Building the general structure.
- Building **one** sequence structure.
- Duplicate it at a certain point and set the parameters of the two sequences to differentiate them.
- Integrate them in the general structure.

STRUCTURE

The architecture for a Material or a Texture Animation with a NiControllerManager is a bit more complex than for regular animations. *(Make sure you’ve read and understood part 2 & 3).*

Let’s have a brief look at the general structure first.

If we expand the Root Node, a BSXFlags, a NiControllerManager and a NiNode will appear.


Block List

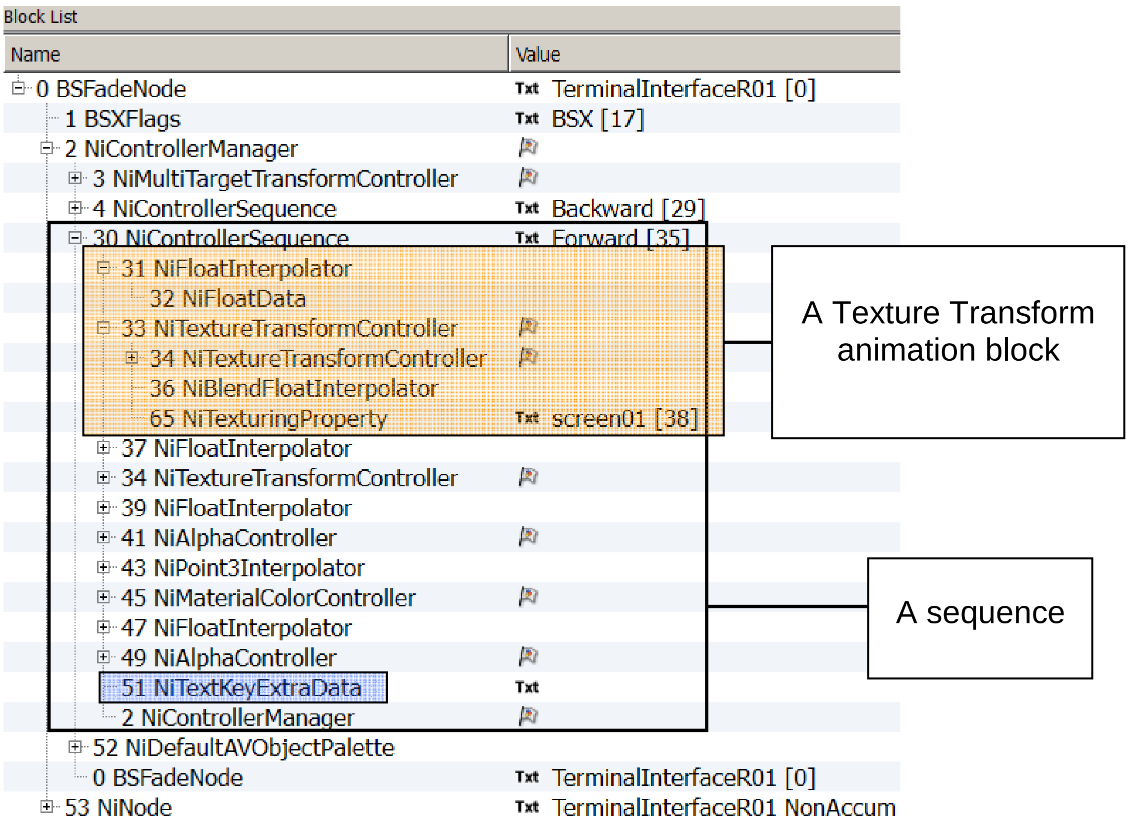
Name	Value
0 BSFadeNode	TerminalInterfaceR01 [0]
1 BSXFlags	BSX [17]
2 NiControllerManager	
3 NiMultiTargetTransformCont...	
4 NiControllerSequence	Backward [29]
30 NiControllerSequence	Forward [35]
52 NiDefaultAVObjectPalette	
0 BSFadeNode	TerminalInterfaceR01 [0]
53 NiNode	TerminalInterfaceR01 NonAccum [34]

The **NiControllerManager** will be in main control. Attached to it is a **NiDefaultAVObjectPalette**, referencing all the NiTriStrips and NiNodes contained in you nif.

As next Controller, a **NiMultiTargetTranformController** also referencing all the NiTriStrips and NiNodes contained in you nif.

Then comes the **NiControllerSequence**. From one (usually an idle animation) to several depending how many animations the object is supposed to play. The names of the NiControllerSequence are coded to be triggered by scripts.

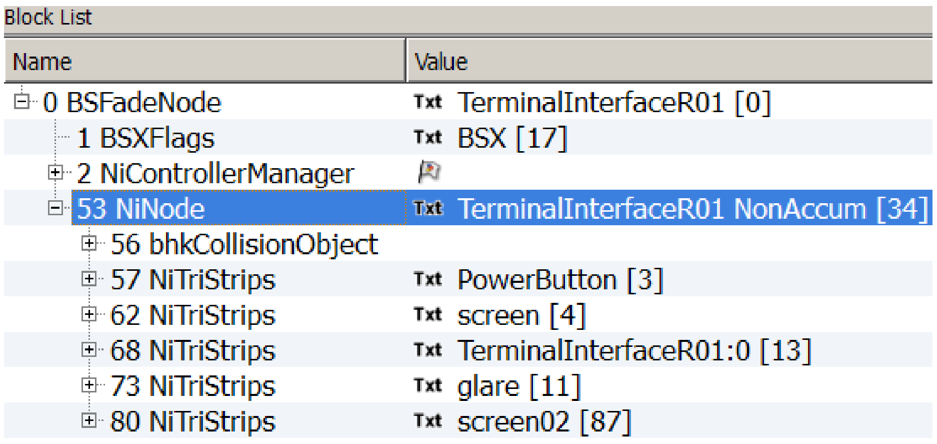
Each NiControllerSequence will have one ore more controlled blocks with the animation blocks build in. Each NiControllerSequence will have a **NiTextKeyExtraData** where Start and Stop time are defined and sound triggering is possibly set.



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterfaceR01 [0]
1 BSXFlags	Txt BSX [17]
2 NiControllerManager	
3 NiMultiTargetTransformController	
4 NiControllerSequence	Txt Backward [29]
30 NiControllerSequence	Txt Forward [35]
31 NiFloatInterpolator	
32 NiFloatData	
33 NiTextureTransformController	
34 NiTextureTransformController	
36 NiBlendFloatInterpolator	
65 NiTexturingProperty	Txt screen01 [38]
37 NiFloatInterpolator	
34 NiTextureTransformController	
39 NiFloatInterpolator	
41 NiAlphaController	
43 NiPoint3Interpolator	
45 NiMaterialColorController	
47 NiFloatInterpolator	
49 NiAlphaController	
51 NiTextKeyExtraData	Txt
2 NiControllerManager	
52 NiDefaultAVObjectPalette	
0 BSFadeNode	Txt TerminalInterfaceR01 [0]
53 NiNode	Txt TerminalInterfaceR01 NonAccum

And lastly, attached to the Root Node, the **NiNode** which will contain all the NiTriStrips and NiNodes from your nif.



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterfaceR01 [0]
1 BSXFlags	Txt BSX [17]
2 NiControllerManager	
53 NiNode	Txt TerminalInterfaceR01 NonAccum [34]
56 bhkCollisionObject	
57 NiTriStrips	Txt PowerButton [3]
62 NiTriStrips	Txt screen [4]
68 NiTriStrips	Txt TerminalInterfaceR01:0 [13]
73 NiTriStrips	Txt glare [11]
80 NiTriStrips	Txt screen02 [87]

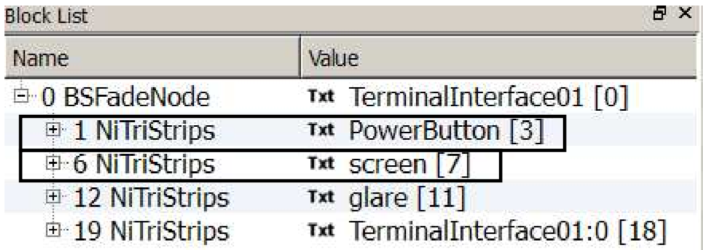
To set up this intricate structure, please follow the steps cautiously, as they are given after careful consideration.

Open DemoCameraMonitor.nif, and DemoTerminalInterFace01.nif in Geck preview.

Open both of them in NifSkope and compare them.

Step by Step

Open DemoTerminalInterFace01.nif



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 NiTriStrips	Txt PowerButton [3]
6 NiTriStrips	Txt screen [7]
12 NiTriStrips	Txt glare [11]
19 NiTriStrips	Txt TerminalInterface01:0 [18]

We are going to animate the screen
And the Power Button

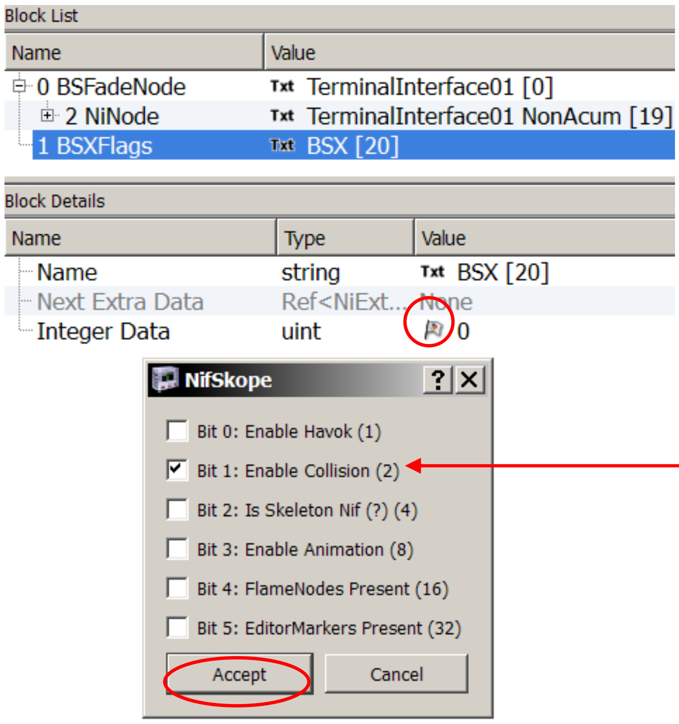
Let's check a few things:

- Is there a BSXFlags? (Collision needs to be activated for player activation detection.)
- Is there any Shader preparatory to do? Better to do it now, if needed. Textures animations need a BSShaderNoLightingProperty.
- Is my future animated NiTriStrips animation ready, is my texture correctly UV sized?
- Are the Material and the Texturing Properties well named?
- Do I have a Collision Object? (Needed for activate triggering.)

Preparatory Work

1. Inserting a BSXFlag (if not present).

- Block ➪ Insert ➪ Bethesda ➪ BSXFlag.
- Click on the flag to access the popup window settings.



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
2 NiNode	Txt TerminalInterface01 NonAcum [19]
1 BSXFlags	Txt BSX [20]

Block Details

Name	Type	Value
Name	string	Txt BSX [20]
Next Extra Data	Ref<NiExt...	None
Integer Data	uint	0

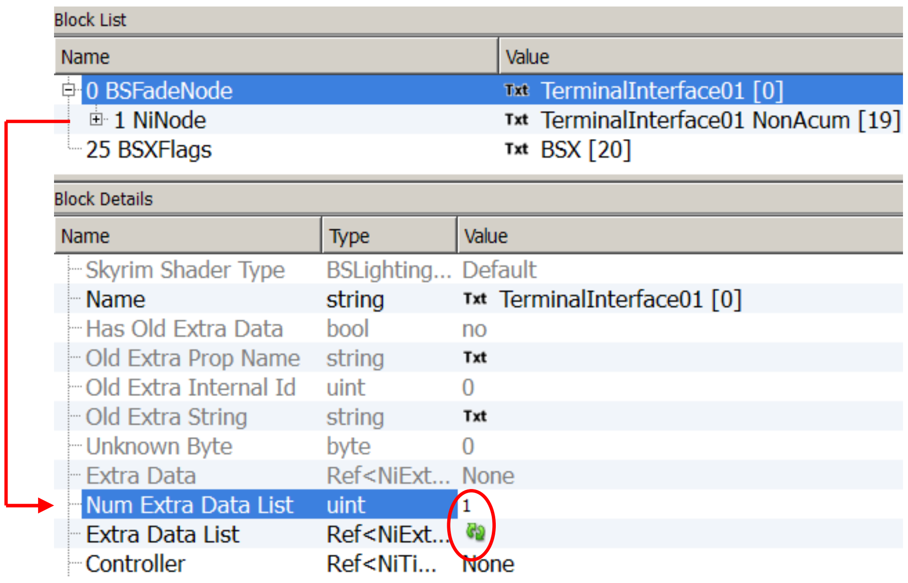
NifSkope

- [ ] Bit 0: Enable Havok (1)
- [x] Bit 1: Enable Collision (2)
- [ ] Bit 2: Is Skeleton Nif (?) (4)
- [ ] Bit 3: Enable Animation (8)
- [ ] Bit 4: FlameNodes Present (16)
- [ ] Bit 5: EditorMarkers Present (32)

Check Bit 1 to enable collision.

1A. Linking the BSXFlag with the Root Node: select the BSFadeNode.

- In Block Details, change the Num Extra Data List to 1, update.

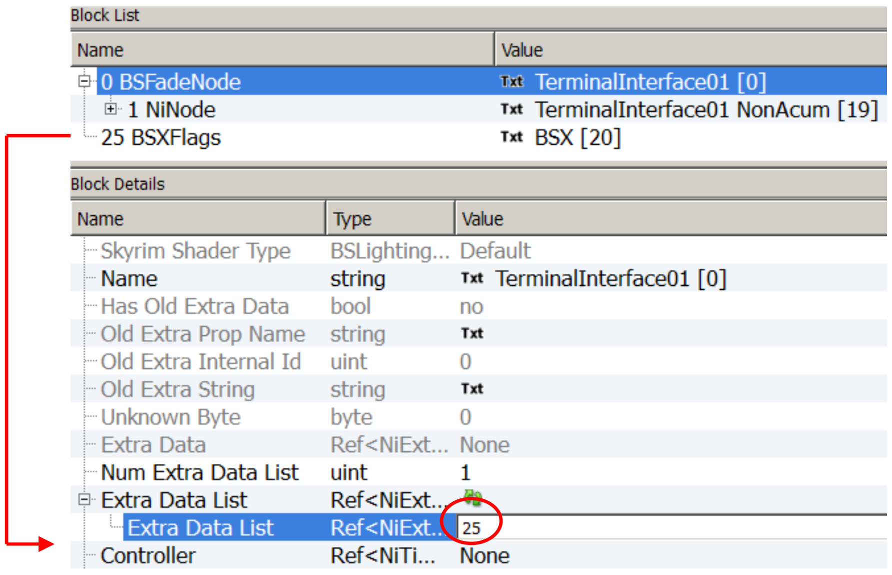

Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 NiNode	Txt TerminalInterface01 NonAcum [19]
25 BSXFlags	Txt BSX [20]

Block Details

Name	Type	Value
Skyrim Shader Type	BSLighting...	Default
Name	string	Txt TerminalInterface01 [0]
Has Old Extra Data	bool	no
Old Extra Prop Name	string	Txt
Old Extra Internal Id	uint	0
Old Extra String	string	Txt
Unknown Byte	byte	0
Extra Data	Ref<NiExt...	None
Num Extra Data List	uint	1
Extra Data List	Ref<NiExt...	
Controller	Ref<NiTi...	None

- In Extra Data List Value, enter the value of the BSXFlag.


Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 NiNode	Txt TerminalInterface01 NonAcum [19]
25 BSXFlags	Txt BSX [20]

Block Details

Name	Type	Value
Skyrim Shader Type	BSLighting...	Default
Name	string	Txt TerminalInterface01 [0]
Has Old Extra Data	bool	no
Old Extra Prop Name	string	Txt
Old Extra Internal Id	uint	0
Old Extra String	string	Txt
Unknown Byte	byte	0
Extra Data	Ref<NiExt...	None
Num Extra Data List	uint	1
Extra Data List	Ref<NiExt...	
Extra Data List	Ref<NiExt...	25
Controller	Ref<NiTi...	None

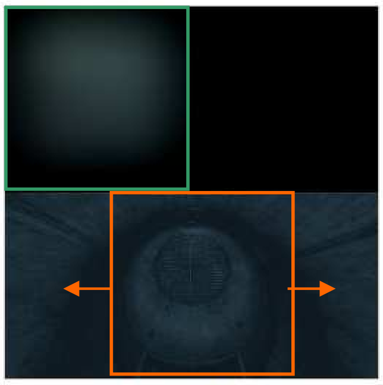
2. Shader preparatory work if needed. *(see Part 3, page 2 of the tutorial.)*
Luckily, not for this demo.

3. Texture choice and size setting.

DemoMonitor.dds.

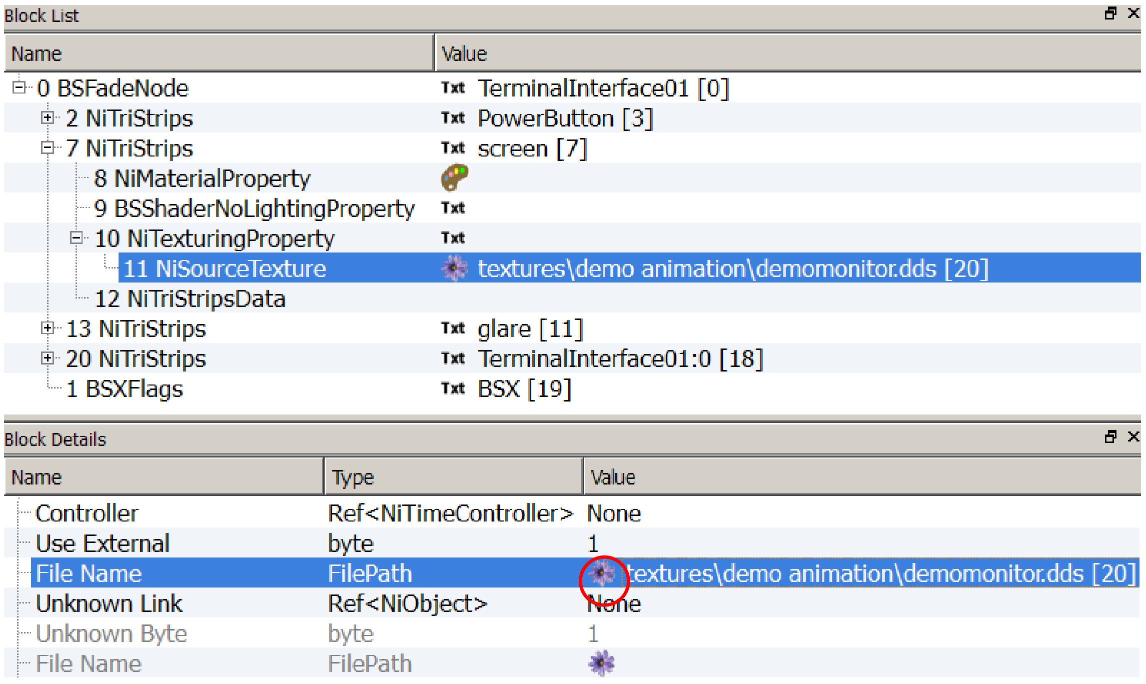
A quarter up left will be used for OFF/Forward

Half for Panning effect ON/Backward


DemoMonitor.dds

The UV map needs to be resized half of its size.

3A. Find the NiSourceTexture of the Screen NiTriStrips. Select the texture (click on mauve flower).



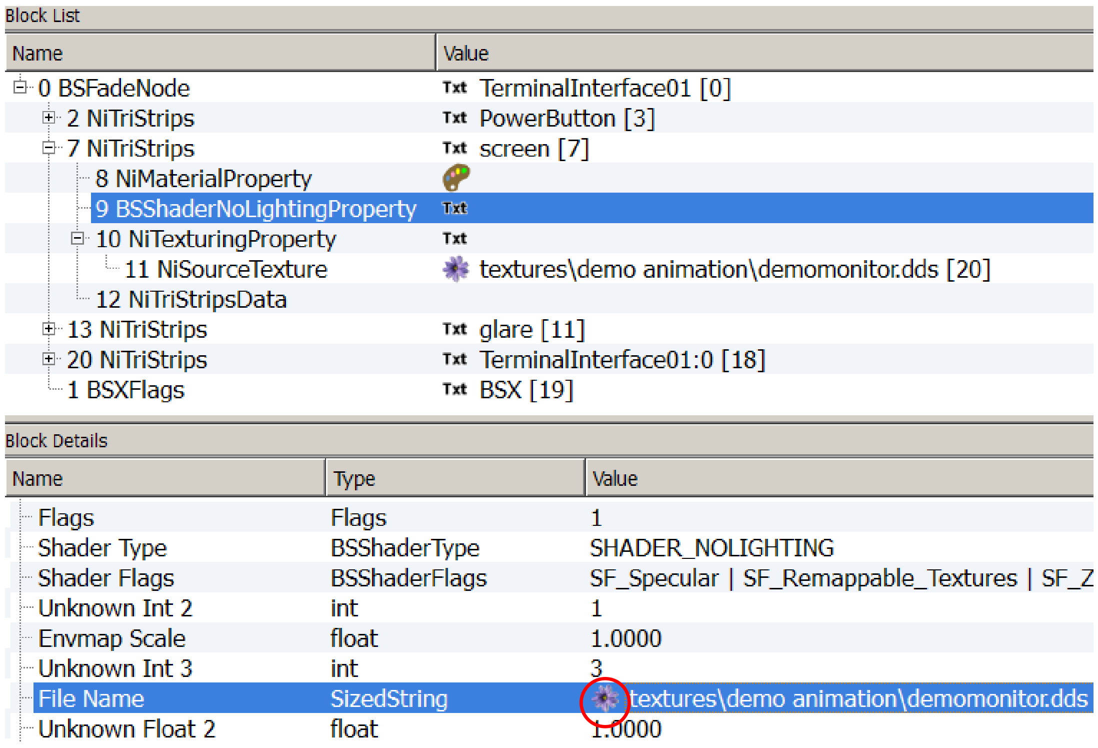
Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
2 NiTriStrips	Txt PowerButton [3]
7 NiTriStrips	Txt screen [7]
8 NiMaterialProperty	
9 BSShaderNoLightingProperty	Txt
10 NiTexturingProperty	Txt
11 NiSourceTexture	✿ textures\demo animation\demomonitor.dds [20]
12 NiTriStripsData	
13 NiTriStrips	Txt glare [11]
20 NiTriStrips	Txt TerminalInterface01:0 [18]
1 BSXFlags	Txt BSX [19]

Block Details

Name	Type	Value
Controller	Ref<NiTimeController>	None
Use External	byte	1
File Name	FilePath	✿ textures\demo animation\demomonitor.dds [20]
Unknown Link	Ref<NiObject>	None
Unknown Byte	byte	1
File Name	FilePath	✿

3B. Select the same texture in the BSShaderNoLightingProperty of the Screen NiTriStrips (mauve flower in File Name).



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
2 NiTriStrips	Txt PowerButton [3]
7 NiTriStrips	Txt screen [7]
8 NiMaterialProperty	
9 BSShaderNoLightingProperty	Txt
10 NiTexturingProperty	Txt
11 NiSourceTexture	✿ textures\demo animation\demomonitor.dds [20]
12 NiTriStripsData	
13 NiTriStrips	Txt glare [11]
20 NiTriStrips	Txt TerminalInterface01:0 [18]
1 BSXFlags	Txt BSX [19]

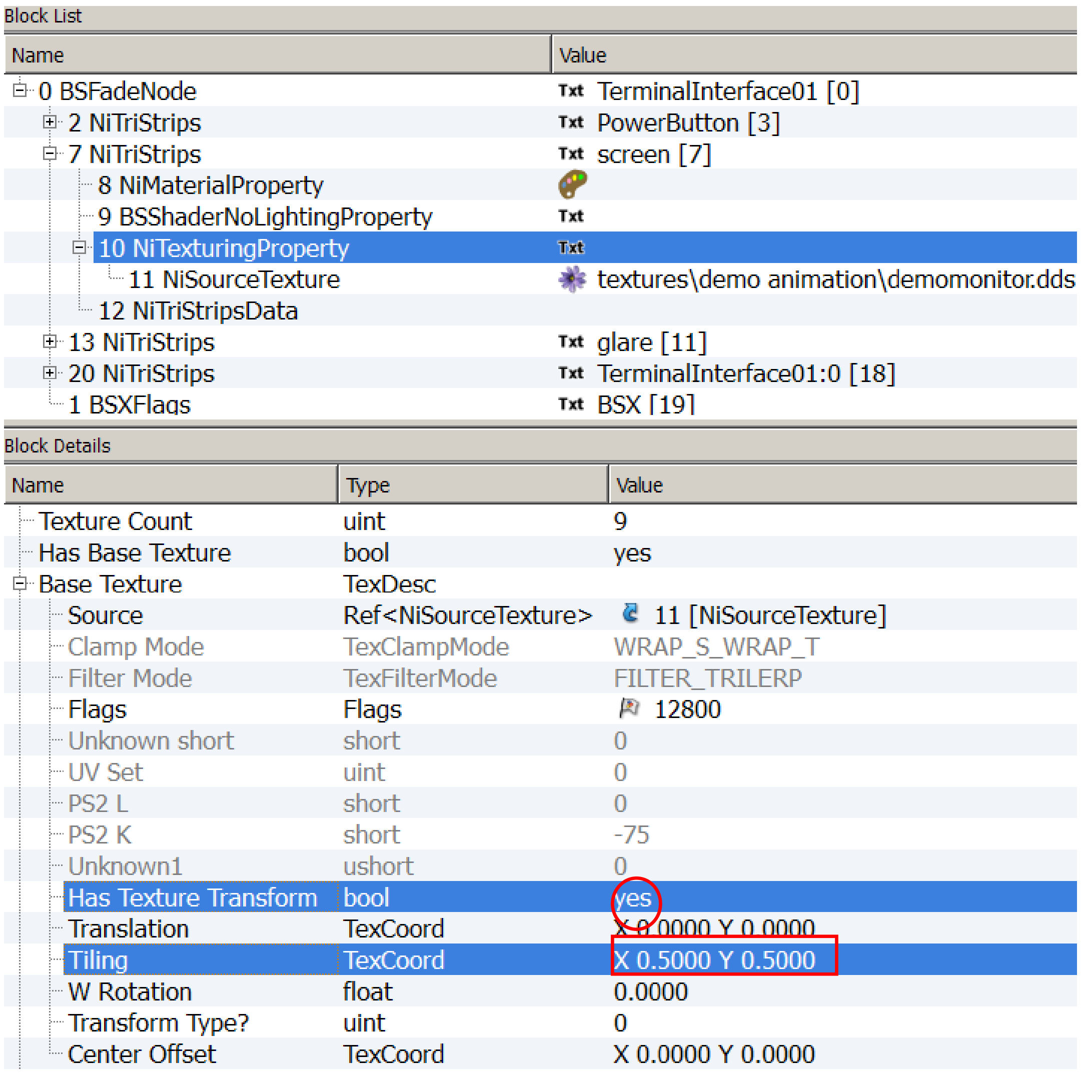
Block Details

Name	Type	Value
Flags	Flags	1
Shader Type	BSShaderType	SHADER_NOLIGHTING
Shader Flags	BSShaderFlags	SF_Specular | SF_Remappable_Textures | SF_Z
Unknown Int 2	int	1
Envmap Scale	float	1.0000
Unknown Int 3	int	3
File Name	SizedString	✿ textures\demo animation\demomonitor.dds
Unknown Float 2	float	1.0000

3C. Enabling textures animation and retiling the UV map half of its size.

In NiTexturingProperty, expand the Base Texture:

- Select Yes for the **Has Texture Transform** Value. This is a switch for animations.
- Enter 0.5000 to both Tiling Value.



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
2 NiTriStrips	Txt PowerButton [3]
7 NiTriStrips	Txt screen [7]
8 NiMaterialProperty	
9 BSShaderNoLightingProperty	Txt
10 NiTexturingProperty	Txt
11 NiSourceTexture	textures\demo animation\demomonitor.dds
12 NiTriStripsData	
13 NiTriStrips	Txt glare [11]
20 NiTriStrips	Txt TerminalInterface01:0 [18]
1 BSXFlags	Txt BSX [19]

Block Details

Name	Type	Value
Texture Count	uint	9
Has Base Texture	bool	yes
Base Texture	TexDesc	
Source	Ref<NiSourceTexture>	11 [NiSourceTexture]
Clamp Mode	TexClampMode	WRAP_S_WRAP_T
Filter Mode	TexFilterMode	FILTER_TRILERP
Flags	Flags	12800
Unknown short	short	0
UV Set	uint	0
PS2 L	short	0
PS2 K	short	-75
Unknown1	ushort	0
Has Texture Transform	bool	yes
Translation	TexCoord	X 0.0000 Y 0.0000
Tiling	TexCoord	X 0.5000 Y 0.5000
W Rotation	float	0.0000
Transform Type?	uint	0
Center Offset	TexCoord	X 0.0000 Y 0.0000

4. Give the NiMaterialProperty and the NiTexturingProperty of the NiTriStrips **screen** a proper name.

5. Do the same with the all Material and Texture Properties of the NiTriStrips you plan to animate.

6. Create a Collision Object if needed.

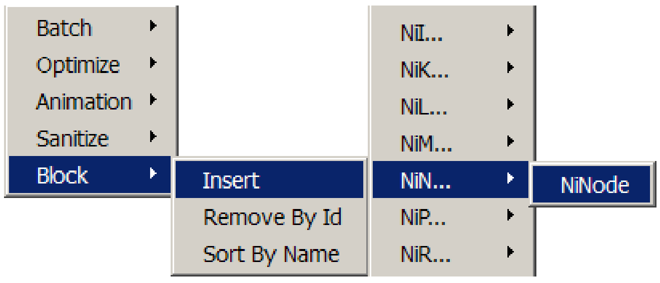
You can save at this point and see in Geck preview if everything is fine, so far.

This close the preparatory work, now, we can start the:

GENERAL STRUCTURE BUILDING

We want all the NiTriStrips to be gathered under one NiNode.

1. At the end, insert a new NiNode.

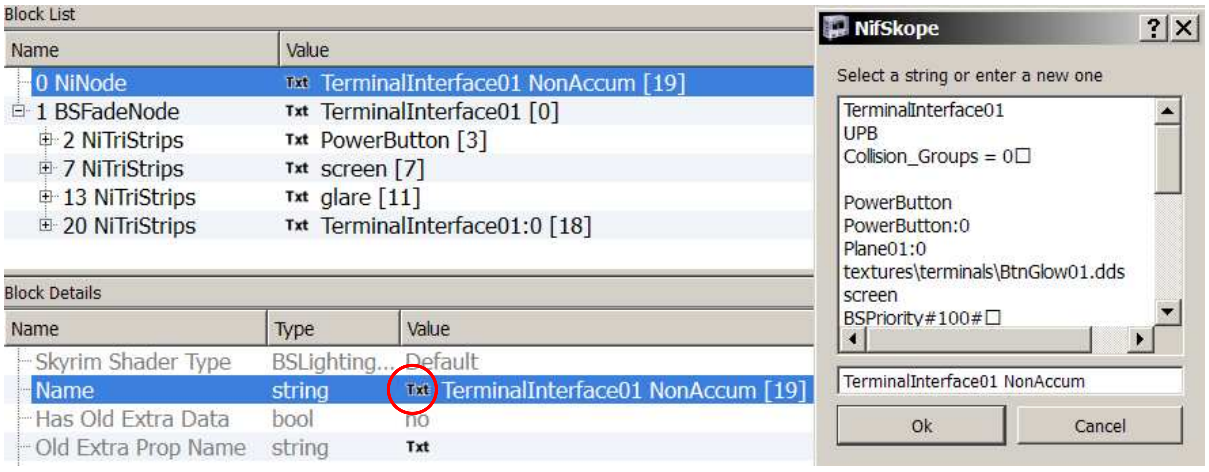


Name it as the Root Node (the BSFadeNode) followed by a space and the suffix NonAccum.

Here that would be: **TerminalInterface01 NonAccum**

(This seems to be a naming convention, not formally needed to make things work. Better to leave it this way and stay on the safe side).

- Click on the Txt Value to access the String pop up window.



Block List

Name	Value
0 NiNode	Txt TerminalInterface01 NonAccum [19]
1 BSFadeNode	Txt TerminalInterface01 [0]
2 NiTriStrips	Txt PowerButton [3]
7 NiTriStrips	Txt screen [7]
13 NiTriStrips	Txt glare [11]
20 NiTriStrips	Txt TerminalInterface01:0 [18]

Block Details

Name	Type	Value
Skyrim Shader Type	BSLighting...	Default
Name	string	Txt TerminalInterface01 NonAccum [19]
Has Old Extra Data	bool	no
Old Extra Prop Name	string	Txt

NifSkope

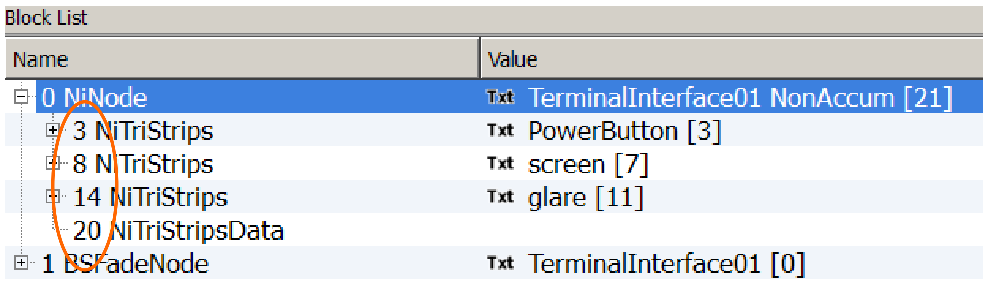
Select a string or enter a new one

```
TerminalInterface01
UPB
Collision_Groups = 0□

PowerButton
PowerButton:0
Plane01:0
textures\terminals\BtnGlow01.dds
screen
BSPriority#100#□
```

TerminalInterface01 NonAccum

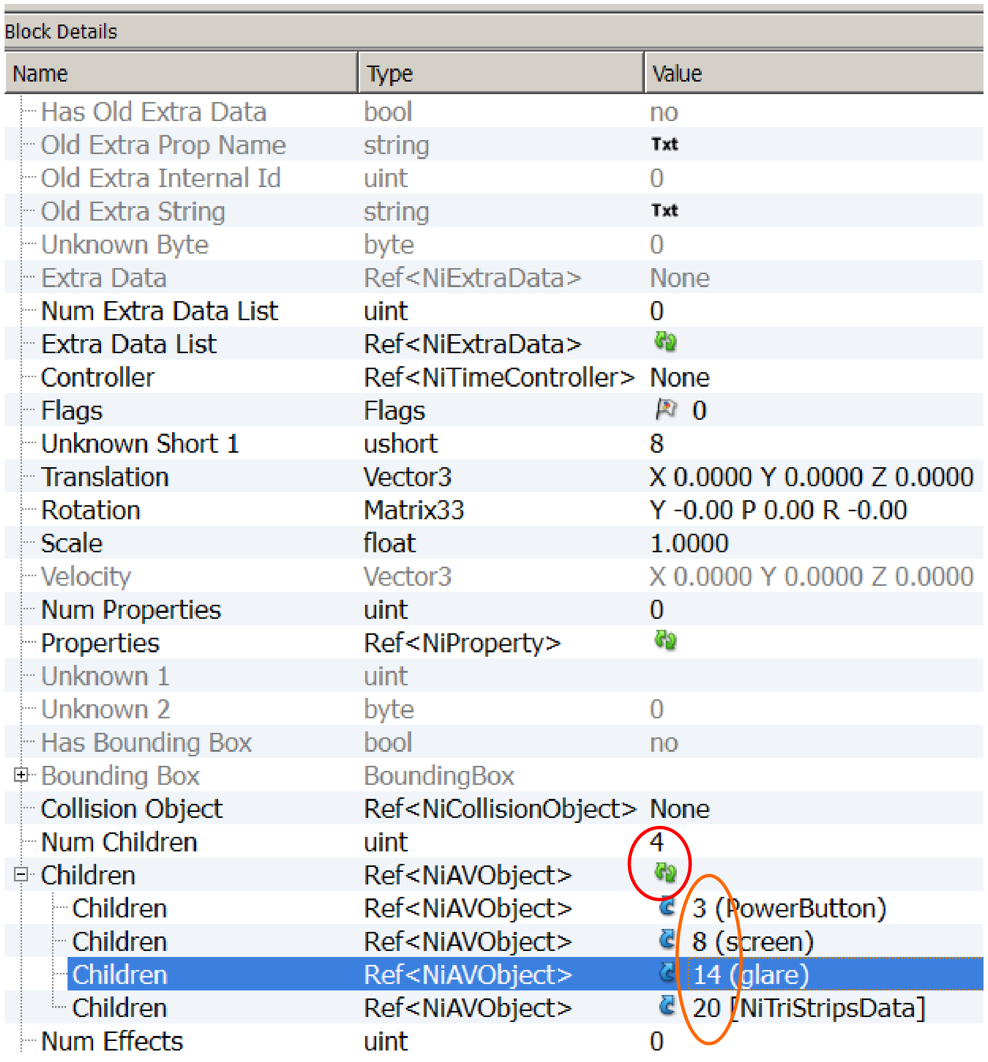
1A. Making all the NiTriStrips a child of the new NiNode.

- Choose the number of child you want in **Num Children**, update.
- Insert the Value of the four NiTriStrips.



Block List

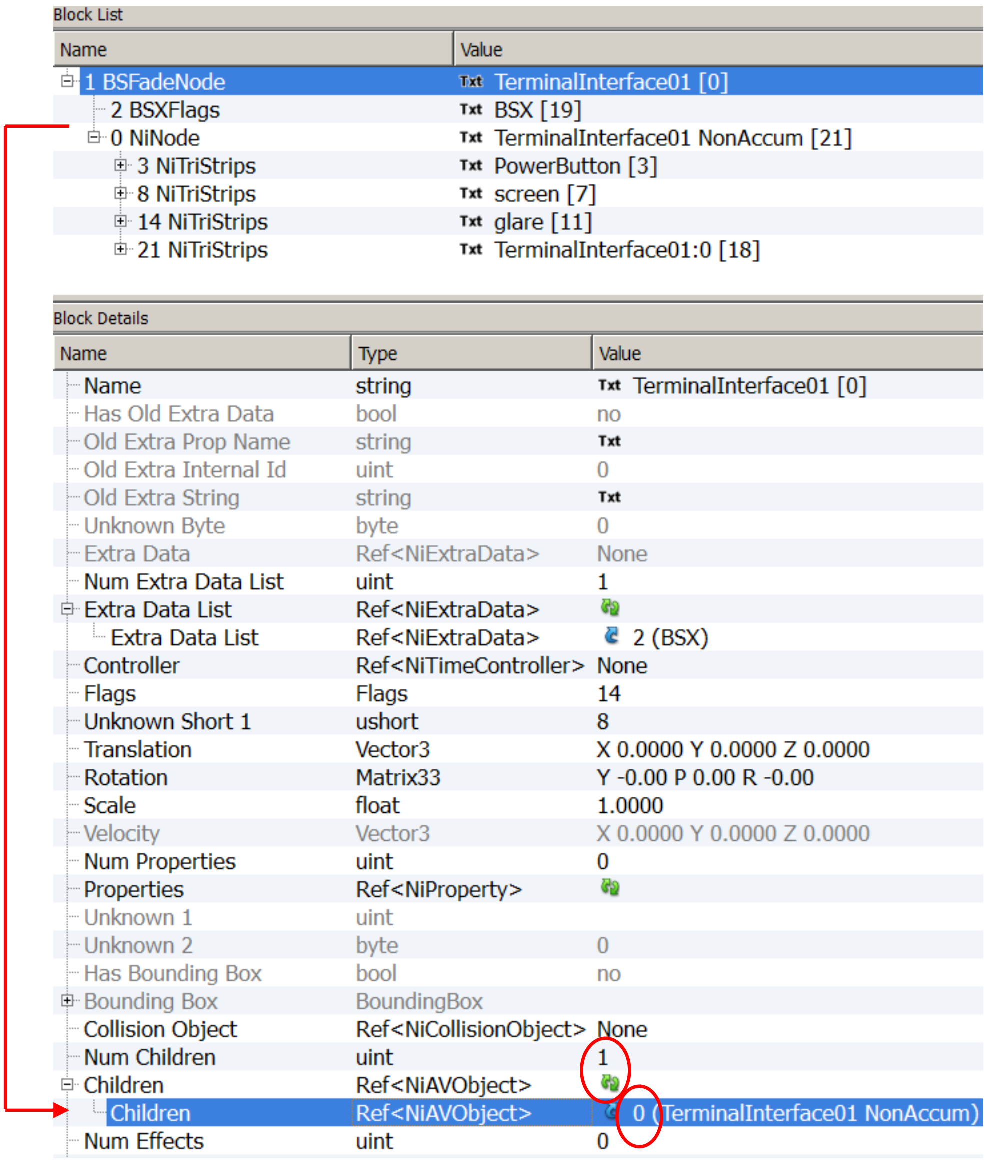
Name	Value
0 NiNode	Txt TerminalInterface01 NonAccum [21]
3 NiTriStrips	Txt PowerButton [3]
8 NiTriStrips	Txt screen [7]
14 NiTriStrips	Txt glare [11]
20 NiTriStripsData	
1 BSFadeNode	Txt TerminalInterface01 [0]



Block Details

Name	Type	Value
Has Old Extra Data	bool	no
Old Extra Prop Name	string	Txt
Old Extra Internal Id	uint	0
Old Extra String	string	Txt
Unknown Byte	byte	0
Extra Data	Ref<NiExtraData>	None
Num Extra Data List	uint	0
Extra Data List	Ref<NiExtraData>	
Controller	Ref<NiTimeController>	None
Flags	Flags	0
Unknown Short 1	ushort	8
Translation	Vector3	X 0.0000 Y 0.0000 Z 0.0000
Rotation	Matrix33	Y -0.00 P 0.00 R -0.00
Scale	float	1.0000
Velocity	Vector3	X 0.0000 Y 0.0000 Z 0.0000
Num Properties	uint	0
Properties	Ref<NiProperty>	
Unknown 1	uint	
Unknown 2	byte	0
Has Bounding Box	bool	no
Bounding Box	BoundingBox	
Collision Object	Ref<NiCollisionObject>	None
Num Children	uint	4
Children	Ref<NiAVObject>	
Children	Ref<NiAVObject>	3 (PowerButton)
Children	Ref<NiAVObject>	8 (screen)
Children	Ref<NiAVObject>	14 (glare)
Children	Ref<NiAVObject>	20 [NiTriStripsData]
Num Effects	uint	0

- **1B**. Making the NiNode the only child of the Root Node.
  Select the Root Node.
- Change the **Num Children** to 1, update.
- Insert the Value of the NiNode in the Children Value field.



Block List

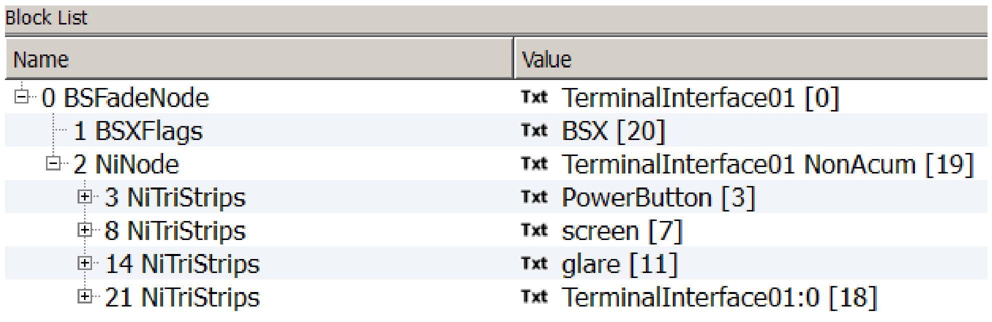
Name	Value
1 BSFadeNode	Txt TerminalInterface01 [0]
2 BSXFlags	Txt BSX [19]
0 NiNode	Txt TerminalInterface01 NonAccum [21]
3 NiTriStrips	Txt PowerButton [3]
8 NiTriStrips	Txt screen [7]
14 NiTriStrips	Txt glare [11]
21 NiTriStrips	Txt TerminalInterface01:0 [18]

Block Details

Name	Type	Value
Name	string	Txt TerminalInterface01 [0]
Has Old Extra Data	bool	no
Old Extra Prop Name	string	Txt
Old Extra Internal Id	uint	0
Old Extra String	string	Txt
Unknown Byte	byte	0
Extra Data	Ref<NiExtraData>	None
Num Extra Data List	uint	1
Extra Data List	Ref<NiExtraData>	
Extra Data List	Ref<NiExtraData>	2 (BSX)
Controller	Ref<NiTimeController>	None
Flags	Flags	14
Unknown Short 1	ushort	8
Translation	Vector3	X 0.0000 Y 0.0000 Z 0.0000
Rotation	Matrix33	Y -0.00 P 0.00 R -0.00
Scale	float	1.0000
Velocity	Vector3	X 0.0000 Y 0.0000 Z 0.0000
Num Properties	uint	0
Properties	Ref<NiProperty>	
Unknown 1	uint	
Unknown 2	byte	0
Has Bounding Box	bool	no
Bounding Box	BoundingBox	
Collision Object	Ref<NiCollisionObject>	None
Num Children	uint	1
Children	Ref<NiAVObject>	
Children	Ref<NiAVObject>	0 (TerminalInterface01 NonAccum)
Num Effects	uint	0

Time to save your work to allow blocks reorganisation.

You should have now a structure like this:



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [20]
2 NiNode	Txt TerminalInterface01 NonAcum [19]
3 NiTriStrips	Txt PowerButton [3]
8 NiTriStrips	Txt screen [7]
14 NiTriStrips	Txt glare [11]
21 NiTriStrips	Txt TerminalInterface01:0 [18]

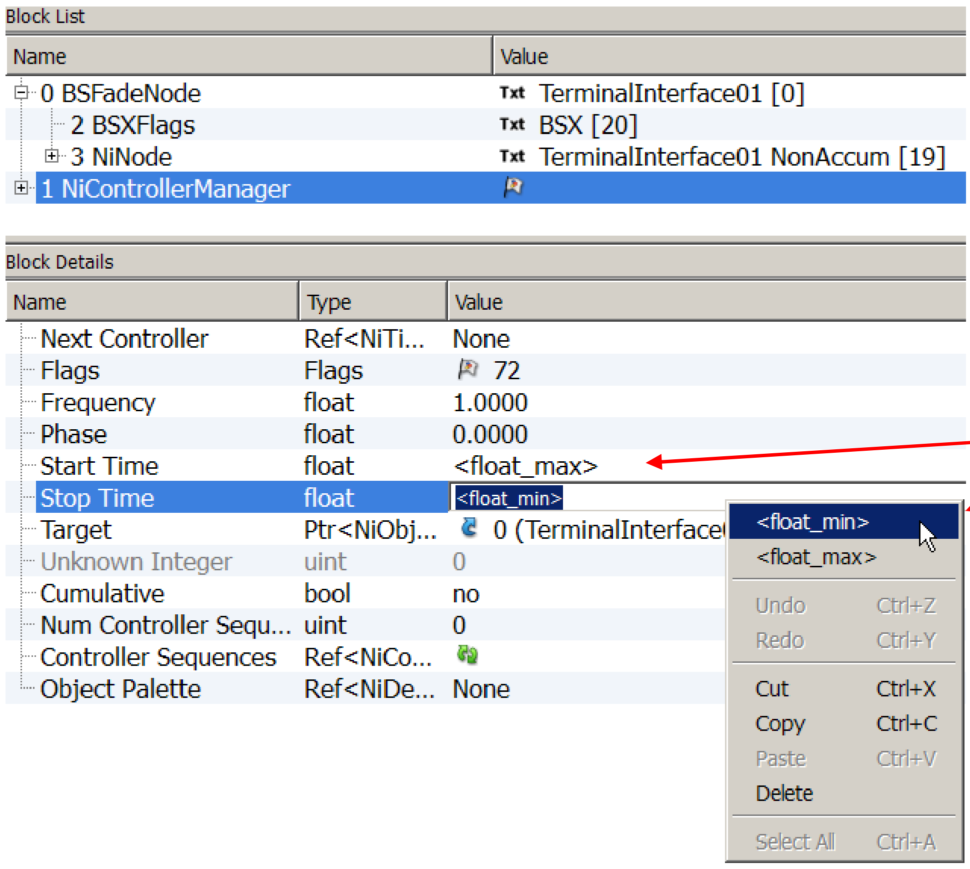
If you open it in Geck now, you won't notice any difference with the starting one.

It is just an internal organisation. However, this base structure is needed to build the animation.

2. Inserting a NiControllerManager.

- Select the Root Node, and then Block ➪ Insert ➪ NiC ➪ NiControllerManager.
- Adjust settings as following:

```
Flags: 72 for looping, 76 for clamp animation
Frequency: 1.0000
Start time: <Float_Max>
Stop Time: <Float_Min>
Target: should be 0 as it references your Root Node
```



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
2 BSXFlags	Txt BSX [20]
3 NiNode	Txt TerminalInterface01 NonAccum [19]
1 NiControllerManager	

Block Details

Name	Type	Value
Next Controller	Ref<NiTi...	None
Flags	Flags	72
Frequency	float	1.0000
Phase	float	0.0000
Start Time	float	<float_max>
Stop Time	float	<float_min>
Target	Ptr<NiObj...	0 (TerminalInterface
Unknown Integer	uint	0
Cumulative	bool	no
Num Controller Sequ...	uint	0
Controller Sequences	Ref<NiCo...	
Object Palette	Ref<NiDe...	None

Right-click the value field to access these settings

The actual length of the animation will be set later on other nodes.

We will take care of the Controller Sequences on a later step.

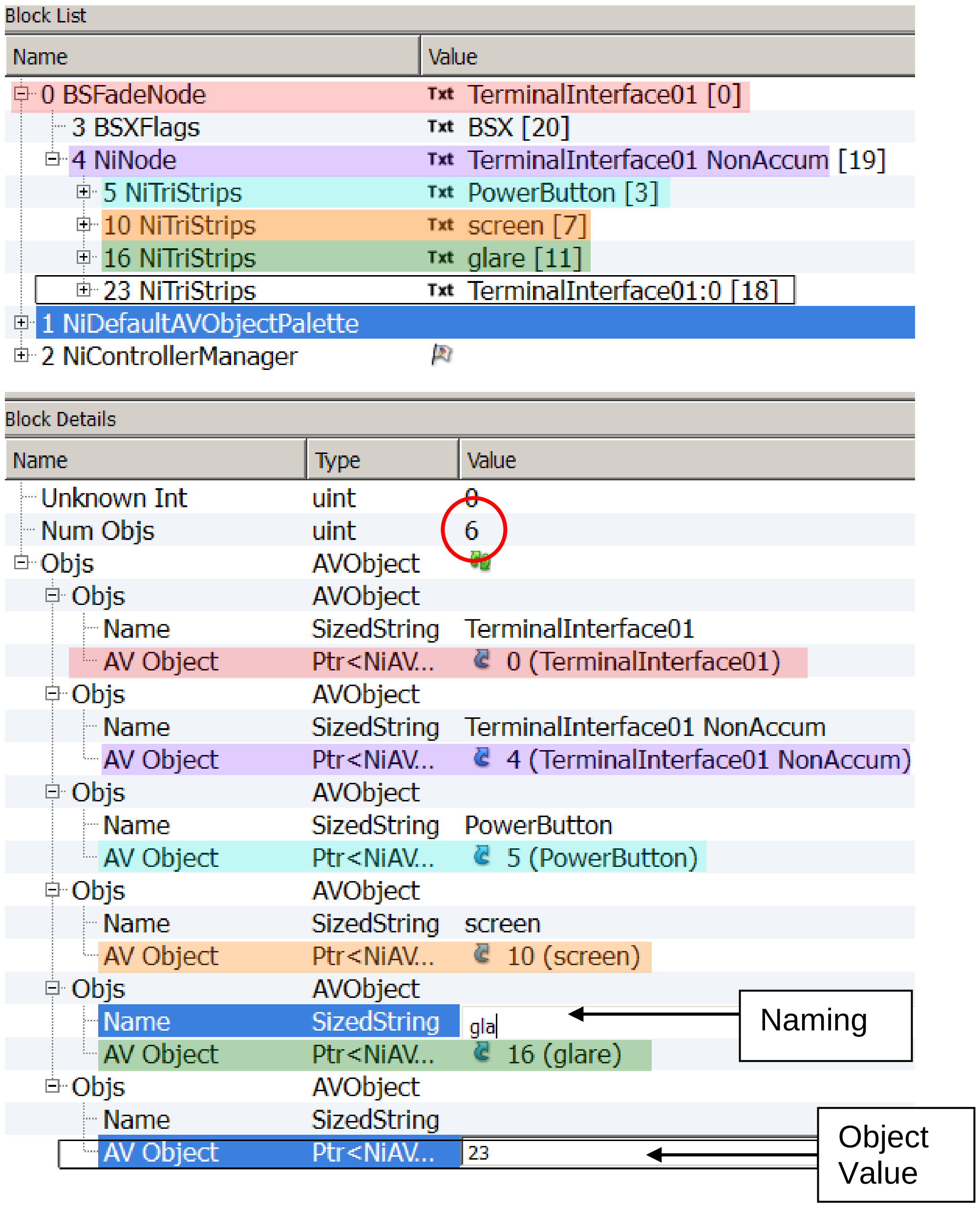
The next step will be to supply an Object Palette to the NiControllerManager.

The Object Palette will list all the NiNodes, NiTriStrips and other blocks contained in your Nif, even the Root Node.

Except for Collision Objects, only those are left out the Object Palette.

3. Inserting a NiDefaultAVObjectPalette.

- Select the Controller then, Block ➪ Insert ➪ NiD ➪ NiDefaultAVObjectPalette.
- Figure out how many objects you'll need to add to the list, insert that number in the Num Objs Value field, update array.
- Reference each of your blocks by inserting their value in the AV Object Value field.
- Insert the string-name in the Name Value field. No typo allowed!



Block List

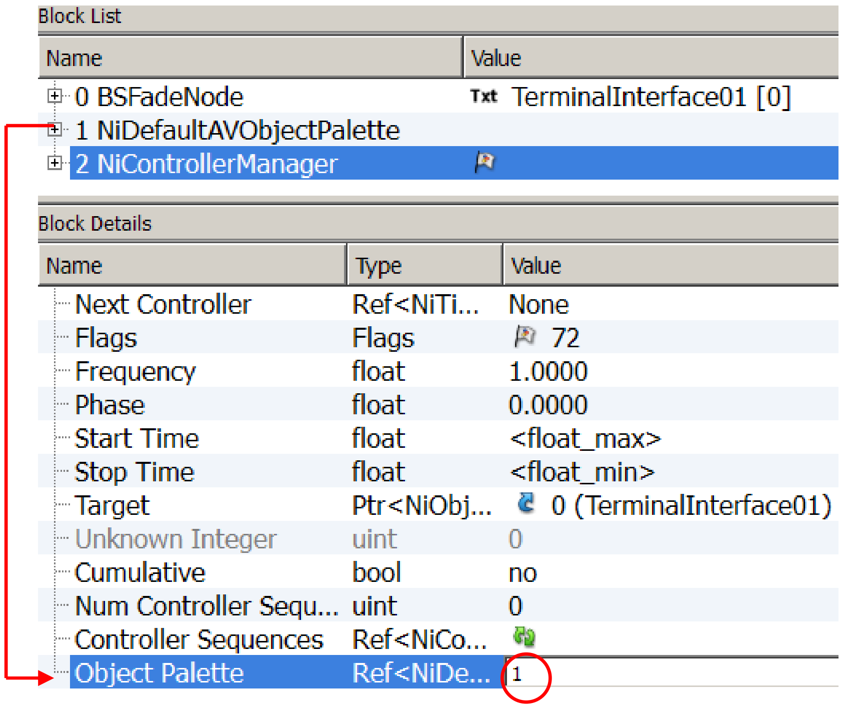
Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
3 BSXFlags	Txt BSX [20]
4 NiNode	Txt TerminalInterface01 NonAccum [19]
5 NiTriStrips	Txt PowerButton [3]
10 NiTriStrips	Txt screen [7]
16 NiTriStrips	Txt glare [11]
23 NiTriStrips	Txt TerminalInterface01:0 [18]
1 NiDefaultAVObjectPalette	
2 NiControllerManager	

Block Details

Name	Type	Value
Unknown Int	uint	0
Num Objs	uint	6
Objs	AVObject	
Objs	AVObject	
Name	SizedString	TerminalInterface01
AV Object	Ptr<NiAV...	0 (TerminalInterface01)
Objs	AVObject	
Name	SizedString	TerminalInterface01 NonAccum
AV Object	Ptr<NiAV...	4 (TerminalInterface01 NonAccum)
Objs	AVObject	
Name	SizedString	PowerButton
AV Object	Ptr<NiAV...	5 (PowerButton)
Objs	AVObject	
Name	SizedString	screen
AV Object	Ptr<NiAV...	10 (screen)
Objs	AVObject	
Name	SizedString	gla
AV Object	Ptr<NiAV...	16 (glare)
Objs	AVObject	
Name	SizedString	
AV Object	Ptr<NiAV...	23

This list can be extended later if need be.

4. Once your Object Palette is completed, attach it to the NiControllerManager.



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 NiDefaultAVObjectPalette	
2 NiControllerManager	

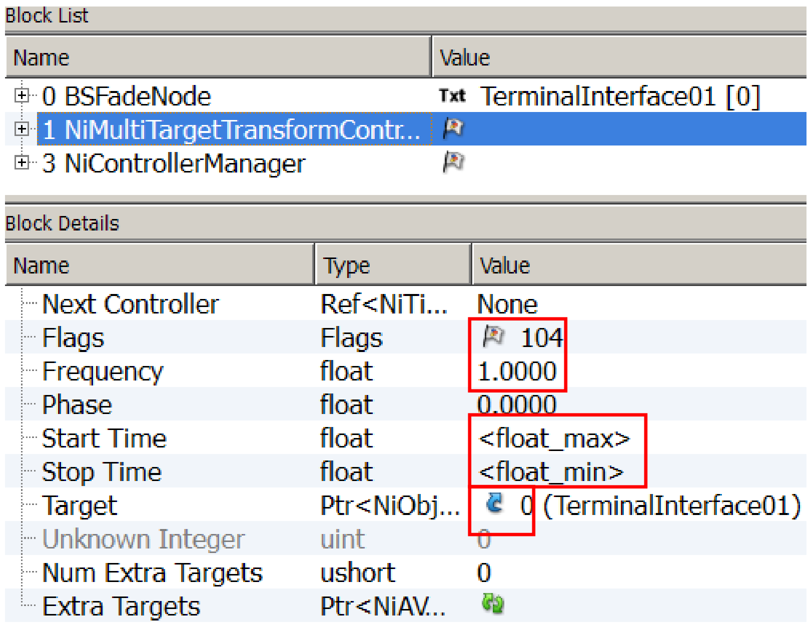
Block Details

Name	Type	Value
Next Controller	Ref<NiTi...	None
Flags	Flags	72
Frequency	float	1.0000
Phase	float	0.0000
Start Time	float	<float_max>
Stop Time	float	<float_min>
Target	Ptr<NiObj...	0 (TerminalInterface01)
Unknown Integer	uint	0
Cumulative	bool	no
Num Controller Sequ...	uint	0
Controller Sequences	Ref<NiCo...	
Object Palette	Ref<NiDe...	1

A new controller will be added now, the NiMultiTargetTranformController.

- Block ➪ Insert ➪ NiM ➪ NiMultiTargetTranformController.
- Adjust settings as following:

Flags: 104
Frequency: 1.0000
Start time: <Float_Max>
Stop Time: <Float_Min>
Target: should be 0 as it references you Root Node



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 NiMultiTargetTransformContr...	
3 NiControllerManager	

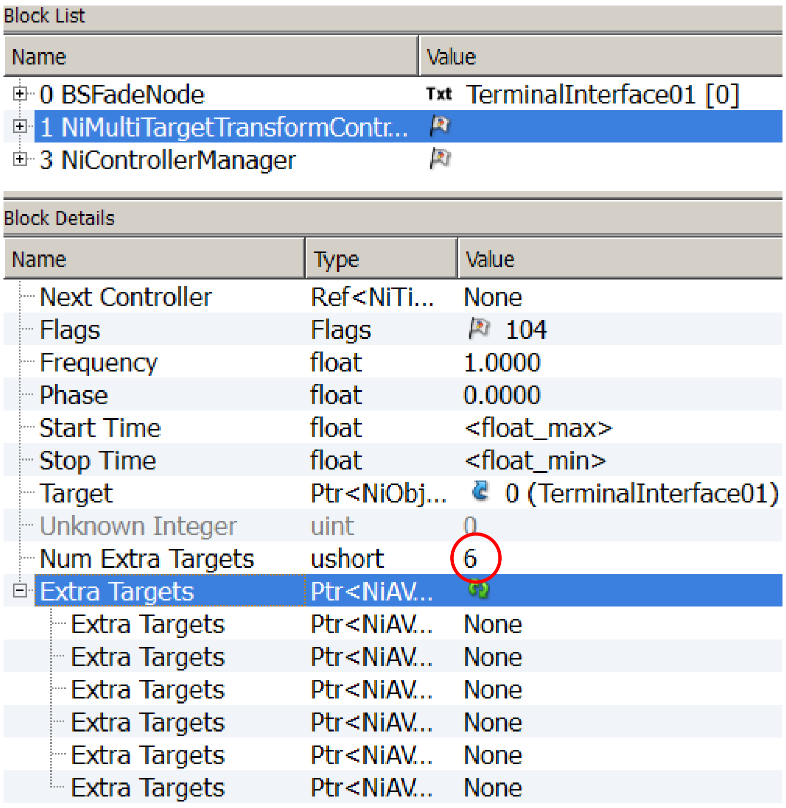
Block Details

Name	Type	Value
Next Controller	Ref<NiTi...	None
Flags	Flags	104
Frequency	float	1.0000
Phase	float	0.0000
Start Time	float	<float_max>
Stop Time	float	<float_min>
Target	Ptr<NiObj...	0 (TerminalInterface01)
Unknown Integer	uint	0
Num Extra Targets	ushort	0
Extra Targets	Ptr<NiAV...	

The following instructions differ from TrickyVein's tutorial (about object animation). They are given according my experience, based on the inspection of quite a few nif with texture animations.
They are, at least, valid as far as Material and Textures animation are concerned.

- In Num Extra Targets Value field, insert the number of NiNodes and NiTriStrips contained in your nif. That should be the same as in the NiDefaultAVObjectPalette.

There is no need to name them here (although, it is safe to do it). As long that the declared Objects/Extra Targets numbers are corresponding.

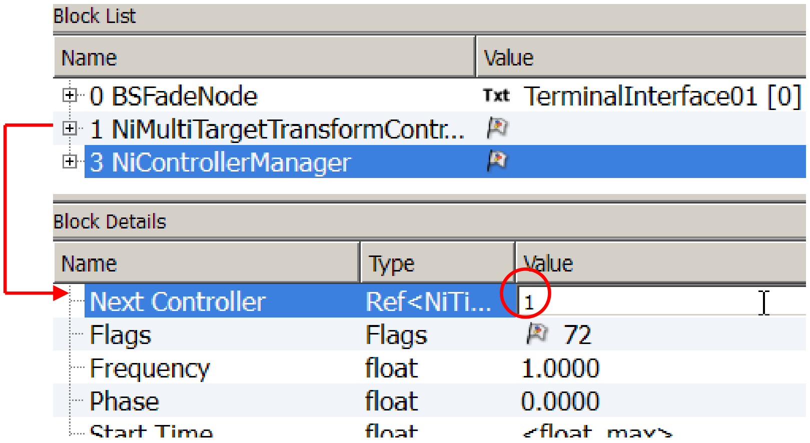

Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 NiMultiTargetTransformContr...	
3 NiControllerManager	

Block Details

Name	Type	Value
Next Controller	Ref<NiTi...	None
Flags	Flags	104
Frequency	float	1.0000
Phase	float	0.0000
Start Time	float	<float_max>
Stop Time	float	<float_min>
Target	Ptr<NiObj...	0 (TerminalInterface01)
Unknown Integer	uint	0
Num Extra Targets	ushort	6
Extra Targets	Ptr<NiAV...	
Extra Targets	Ptr<NiAV...	None
Extra Targets	Ptr<NiAV...	None
Extra Targets	Ptr<NiAV...	None
Extra Targets	Ptr<NiAV...	None
Extra Targets	Ptr<NiAV...	None
Extra Targets	Ptr<NiAV...	None

5. Linking the NiMultiTargetTranformController to NiControllerManager.


Block List

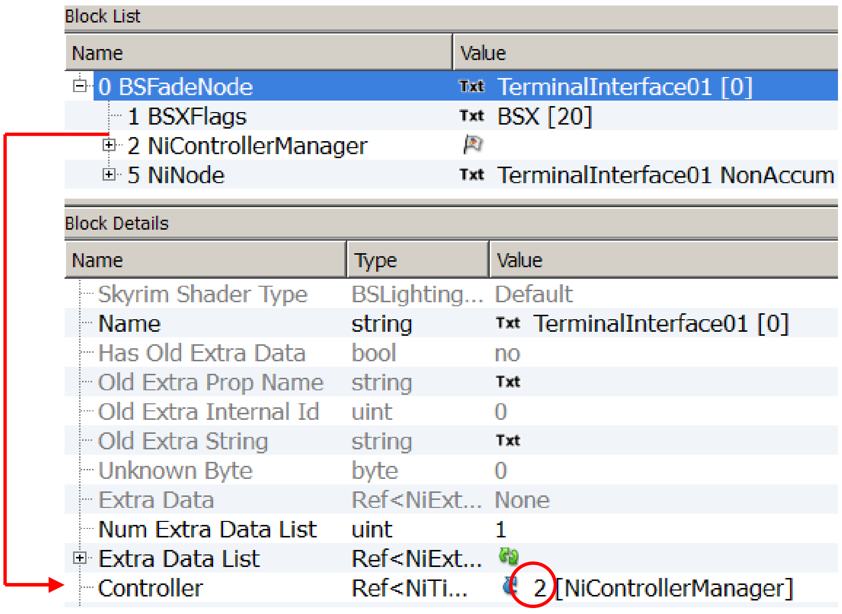
Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 NiMultiTargetTransformContr...	
3 NiControllerManager	

Block Details

Name	Type	Value
Next Controller	Ref<NiTi...	1
Flags	Flags	72
Frequency	float	1.0000
Phase	float	0.0000
Start Time	float	<float_max>

6. Linking the NiControllerManager to the Root Node.

- In the Root Node, set the NiControllerManager as the Controller.



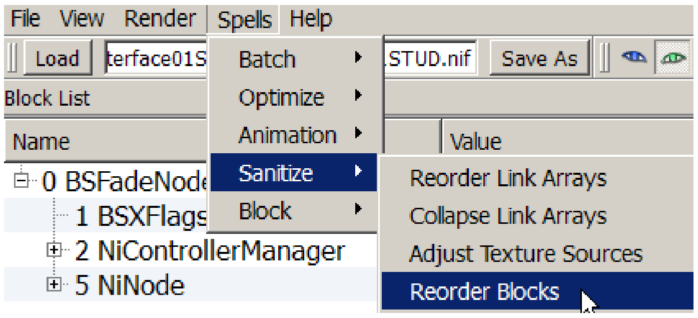
Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [20]
2 NiControllerManager	
5 NiNode	Txt TerminalInterface01 NonAccum

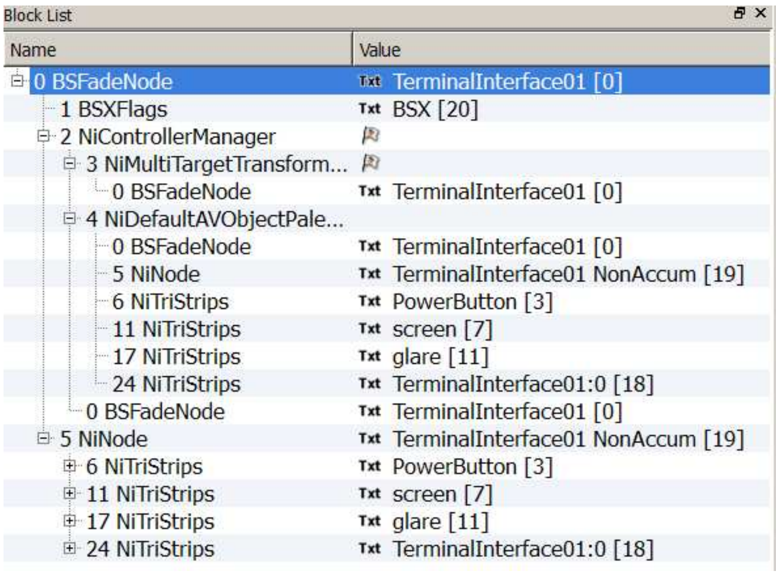
Block Details

Name	Type	Value
Skyrim Shader Type	BSLighting...	Default
Name	string	Txt TerminalInterface01 [0]
Has Old Extra Data	bool	no
Old Extra Prop Name	string	Txt
Old Extra Internal Id	uint	0
Old Extra String	string	Txt
Unknown Byte	byte	0
Extra Data	Ref<NiExt...	None
Num Extra Data List	uint	1
Extra Data List	Ref<NiExt...	
Controller	Ref<NiTi...	2 [NiControllerManager]

Save your work to let blocks reorganize or use Spell ➪ Sanitize ➪ Reorder Blocks.



This should be the result at this stage. Don’t open in Geck as it will crash.



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [20]
2 NiControllerManager	
3 NiMultiTargetTransform...	
0 BSFadeNode	Txt TerminalInterface01 [0]
4 NiDefaultAVObjectPale...	
0 BSFadeNode	Txt TerminalInterface01 [0]
5 NiNode	Txt TerminalInterface01 NonAccum [19]
6 NiTriStrips	Txt PowerButton [3]
11 NiTriStrips	Txt screen [7]
17 NiTriStrips	Txt glare [11]
24 NiTriStrips	Txt TerminalInterface01:0 [18]
0 BSFadeNode	Txt TerminalInterface01 [0]
5 NiNode	Txt TerminalInterface01 NonAccum [19]
6 NiTriStrips	Txt PowerButton [3]
11 NiTriStrips	Txt screen [7]
17 NiTriStrips	Txt glare [11]
24 NiTriStrips	Txt TerminalInterface01:0 [18]

Actually, you won’t be able to check your progress in Geck before a long series of manipulations. So, get coffee supplies, take a deep breath and let’s start the Sequence building process:

SEQUENCE BUILDING

We will need two sequences, both similar in their structure and containing the same type of animation. It means that we can create one sequence and duplicate it at a certain point to save some work.

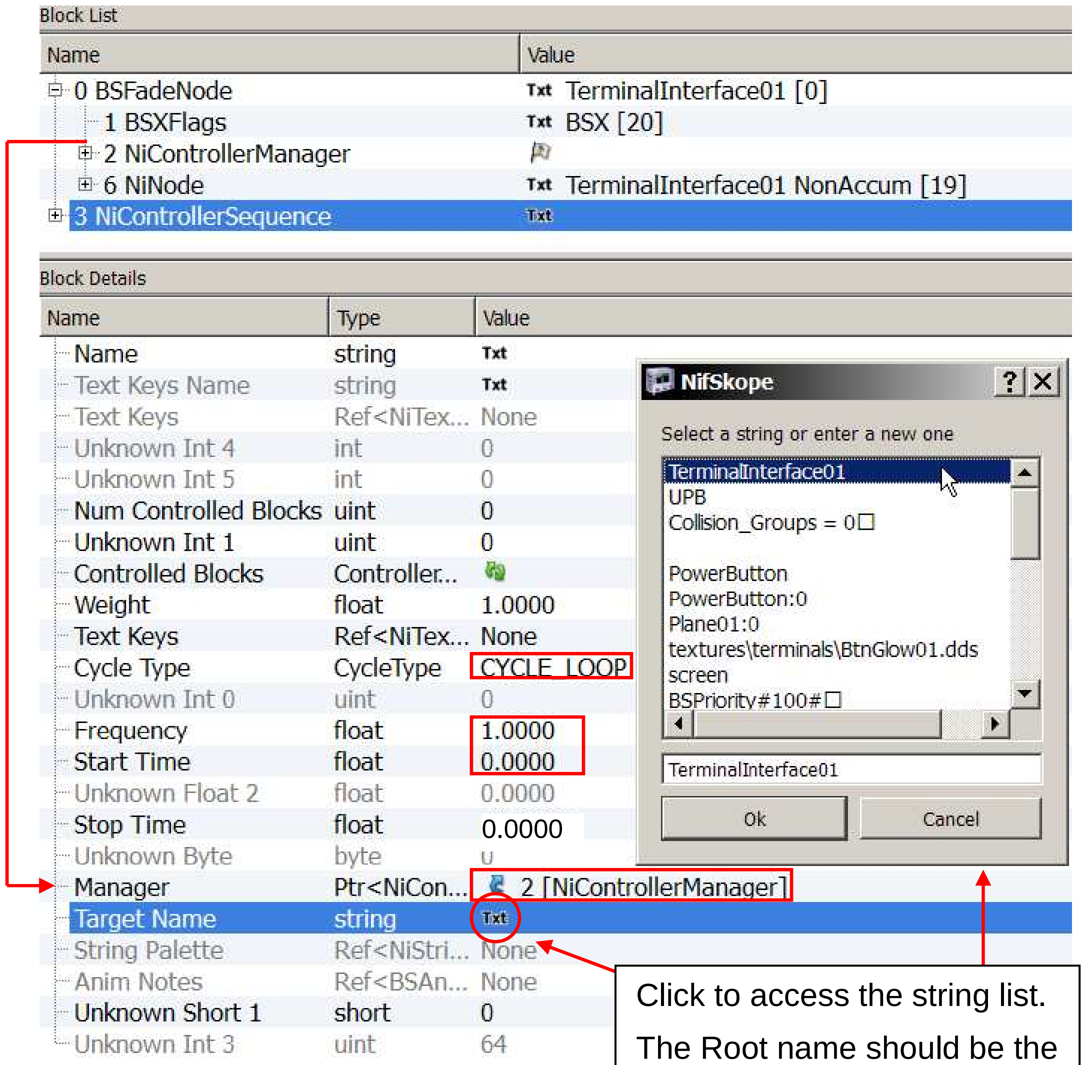
1. Inserting Sequence.

   Select your NiControllerManager, then
   Block ⇨ Insert ⇨ NiC ⇨ NiControllerSequence.

2. Configuring NiControllerSequence.

   We will set the parameters common to both sequences.

Cycle Type: CYCLE-LOOP Frequency: 1.0000 Start time: 0.000 Manager: Value of the NiControllerManager Target Name: Name of the Root Node



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [20]
2 NiControllerManager	
6 NiNode	Txt TerminalInterface01 NonAccum [19]
3 NiControllerSequence	Txt

Block Details

Name	Type	Value
Name	string	Txt
Text Keys Name	string	Txt
Text Keys	Ref<NiTex...	None
Unknown Int 4	int	0
Unknown Int 5	int	0
Num Controlled Blocks	uint	0
Unknown Int 1	uint	0
Controlled Blocks	Controller...	
Weight	float	1.0000
Text Keys	Ref<NiTex...	None
Cycle Type	CycleType	CYCLE_LOOP
Unknown Int 0	uint	0
Frequency	float	1.0000
Start Time	float	0.0000
Unknown Float 2	float	0.0000
Stop Time	float	0.0000
Unknown Byte	byte	0
Manager	Ptr<NiCon...	2 [NiControllerManager]
Target Name	string	Txt
String Palette	Ref<NiStri...	None
Anim Notes	Ref<BSAn...	None
Unknown Short 1	short	0
Unknown Int 3	uint	64

NifSkope

Select a string or enter a new one

TerminalInterface01
UPB
Collision_Groups = 0□

PowerButton
PowerButton:0
Plane01:0
textures\terminals\BtnGlow01.dds
screen
BSPriority#100#□

TerminalInterface01

Ok Cancel

Click to access the string list.

The Root name should be the first on the list.

Each Controller Sequence has a Text Data that defines Start and Stop times, along with the possibility of triggering sounds at certain point.

3. Inserting a NiTextKeyExtraData.

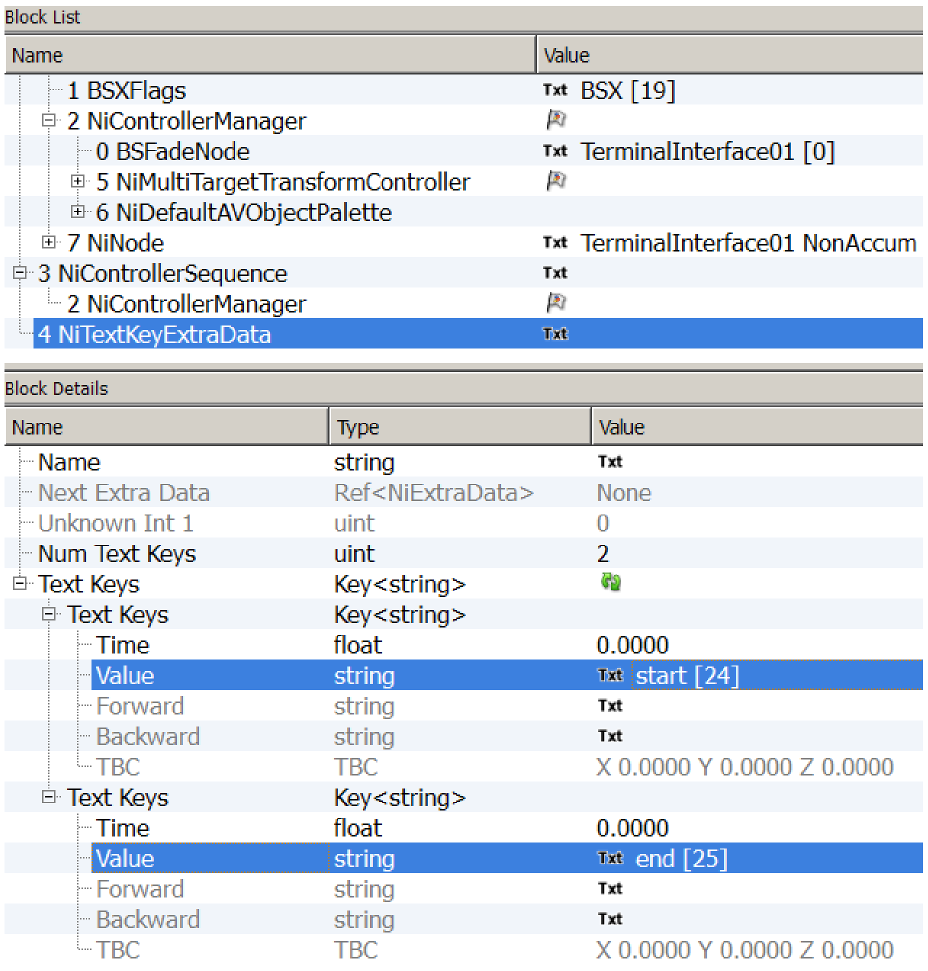
- Select the NiControllerSequence, then:

- Block ⇨ Insert ⇨ NiT ⇨ NiTextKeyExtraData.

4. Setting the NiTextKeyExtraData number of keys.

- Select the NiTextKeyExtraData. Choose the number of Text key you want. (Here it would be 2). Update.
- Enter the string "**start**" without quotes in the first **Value** Value field. (click on Txt to access to String pop up window.)
- Enter the string "**end**" without quotes in the last **Value** Value field.

The Time Values be set later individually.



Block List

Name	Value
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
0 BSFadeNode	Txt TerminalInterface01 [0]
5 NiMultiTargetTransformController	
6 NiDefaultAVObjectPalette	
7 NiNode	Txt TerminalInterface01 NonAccum
3 NiControllerSequence	Txt
2 NiControllerManager	
4 NiTextKeyExtraData	Txt

Block Details

Name	Type	Value
Name	string	Txt
Next Extra Data	Ref<NiExtraData>	None
Unknown Int 1	uint	0
Num Text Keys	uint	2
Text Keys	Key<string>	
Text Keys	Key<string>	
Time	float	0.0000
Value	string	Txt start [24]
Forward	string	Txt
Backward	string	Txt
TBC	TBC	X 0.0000 Y 0.0000 Z 0.0000
Text Keys	Key<string>	
Time	float	0.0000
Value	string	Txt end [25]
Forward	string	Txt
Backward	string	Txt
TBC	TBC	X 0.0000 Y 0.0000 Z 0.0000

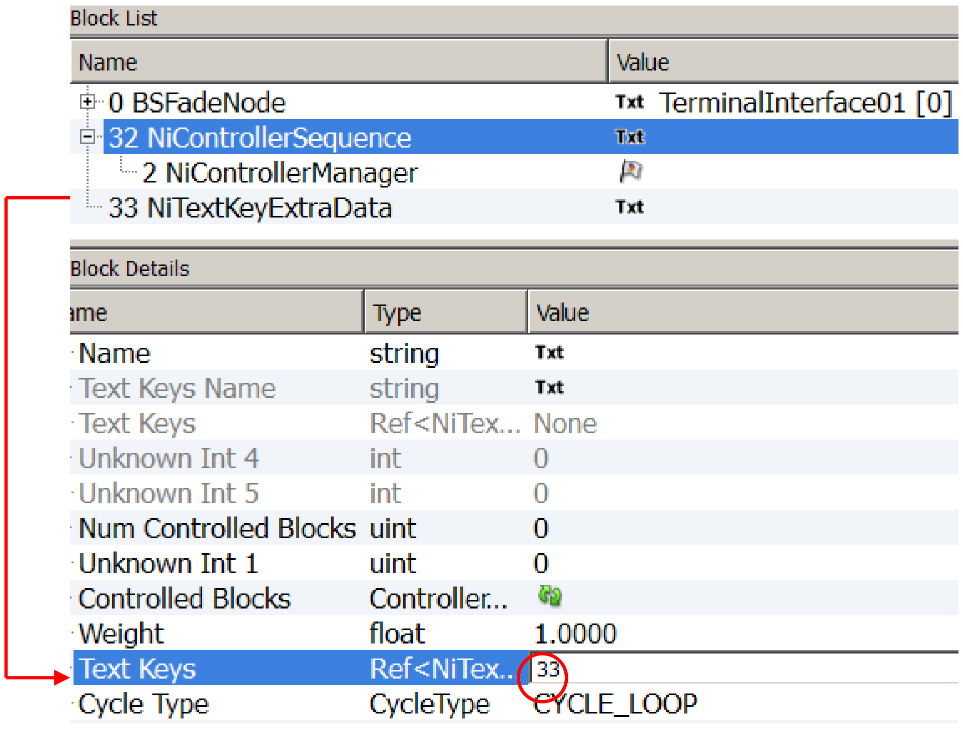
This is also the place where you can trigger sound in your animation. You can start a sound but you cannot stop it. So, if you trigger a looped sound, it will play forever and, each time you activate the sequence a new occurrence of the sound will start and stack up.
Use non looping sound.

Add a key between start and end, choose your time of triggering and set the Value as:
Sound: NameOfTheSound

Where NameOfTheSound is either an existing name of a sound referenced in Geck or the name of a sound you've created in Geck.

Linking the NiTextKeyExtraData to the NiControllerSequence.

- In the NiControllerSequence, set the Value of the NiTextKeyExtraData in the Text Keys Value.



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
32 NiControllerSequence	Txt
2 NiControllerManager	
33 NiTextKeyExtraData	Txt

Block Details

ame	Type	Value
Name	string	Txt
Text Keys Name	string	Txt
Text Keys	Ref<NiTex...	None
Unknown Int 4	int	0
Unknown Int 5	int	0
Num Controlled Blocks	uint	0
Unknown Int 1	uint	0
Controlled Blocks	Controller...	
Weight	float	1.0000
Text Keys	Ref<NiTex...	33
Cycle Type	CycleType	CYCLE_LOOP

This is the basics of the sequence structure. You might want to save your work now as it can be a starting point for variations. (Although mesh is not usable as it is and will crash in the Geck.)

We will go further with this process, by building animation block inside the sequence.

Four blocks will be built:

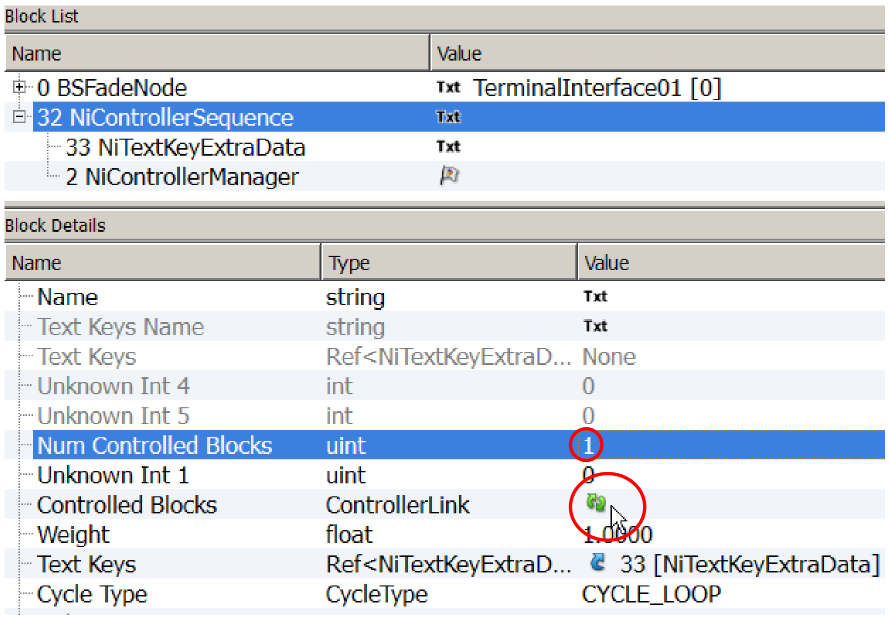
- One Alpha animation for future (in)visibility,
- One Emissive animation
- Two translate animations: U and V (for panning when ON and fixed when OFF).

BUILDING ANIMATION BLOCKS WITHIN A SEQUENCE

It is easier to define the blocks before being building them. You will need four Controlled Blocks in total, but for the clarity of the process, we will build each one by one and add them to the sequence one at the time.

1. Defining the animation blocks.

1A. Select the NiControllerSequence, in block details: Set the **Num Controlled Blocks** to 1. Update.



Block List

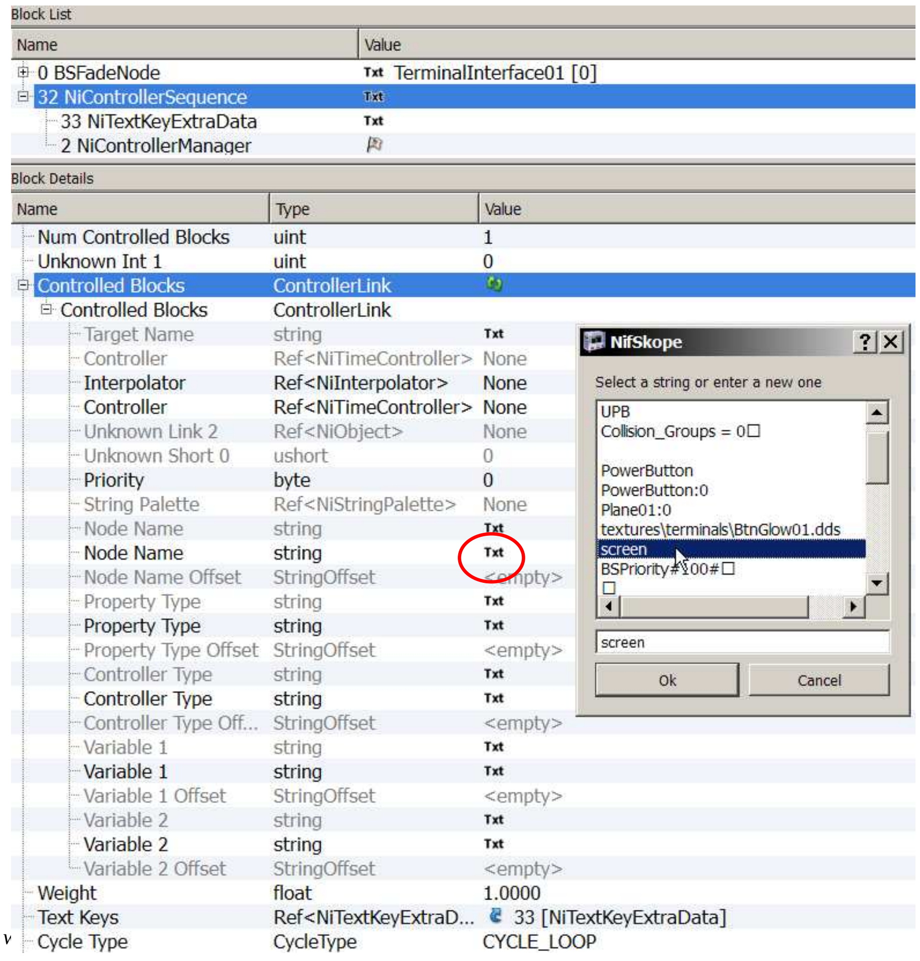
Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
32 NiControllerSequence	Txt
33 NiTextKeyExtraData	Txt
2 NiControllerManager	

Block Details

Name	Type	Value
Name	string	Txt
Text Keys Name	string	Txt
Text Keys	Ref<NiTextKeyExtraD...	None
Unknown Int 4	int	0
Unknown Int 5	int	0
Num Controlled Blocks	uint	1
Unknown Int 1	uint	0
Controlled Blocks	ControllerLink	
Weight	float	1.0000
Text Keys	Ref<NiTextKeyExtraD...	33 [NiTextKeyExtraData]
Cycle Type	CycleType	CYCLE_LOOP

1B. Expand Controlled Blocks.
In **Node Name**: Select the name of the NiTriStrips you're planning to animate.

Click on the Txt to access the string window.



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
32 NiControllerSequence	Txt
33 NiTextKeyExtraData	Txt
2 NiControllerManager	

Block Details

Name	Type	Value
Num Controlled Blocks	uint	1
Unknown Int 1	uint	0
Controlled Blocks	ControllerLink	
Controlled Blocks	ControllerLink	
Target Name	string	Txt
Controller	Ref<NiTimeController>	None
Interpolator	Ref<NiInterpolator>	None
Controller	Ref<NiTimeController>	None
Unknown Link 2	Ref<NiObject>	None
Unknown Short 0	ushort	0
Priority	byte	0
String Palette	Ref<NiStringPalette>	None
Node Name	string	Txt
Node Name	string	Txt
Node Name Offset	StringOffset	<empty>
Property Type	string	Txt
Property Type	string	Txt
Property Type Offset	StringOffset	<empty>
Controller Type	string	Txt
Controller Type	string	Txt
Controller Type Off...	StringOffset	<empty>
Variable 1	string	Txt
Variable 1	string	Txt
Variable 1 Offset	StringOffset	<empty>
Variable 2	string	Txt
Variable 2	string	Txt
Variable 2 Offset	StringOffset	<empty>
Weight	float	1.0000
Text Keys	Ref<NiTextKeyExtraD...	33 [NiTextKeyExtraData]
Cycle Type	CycleType	CYCLE_LOOP

NifSkope

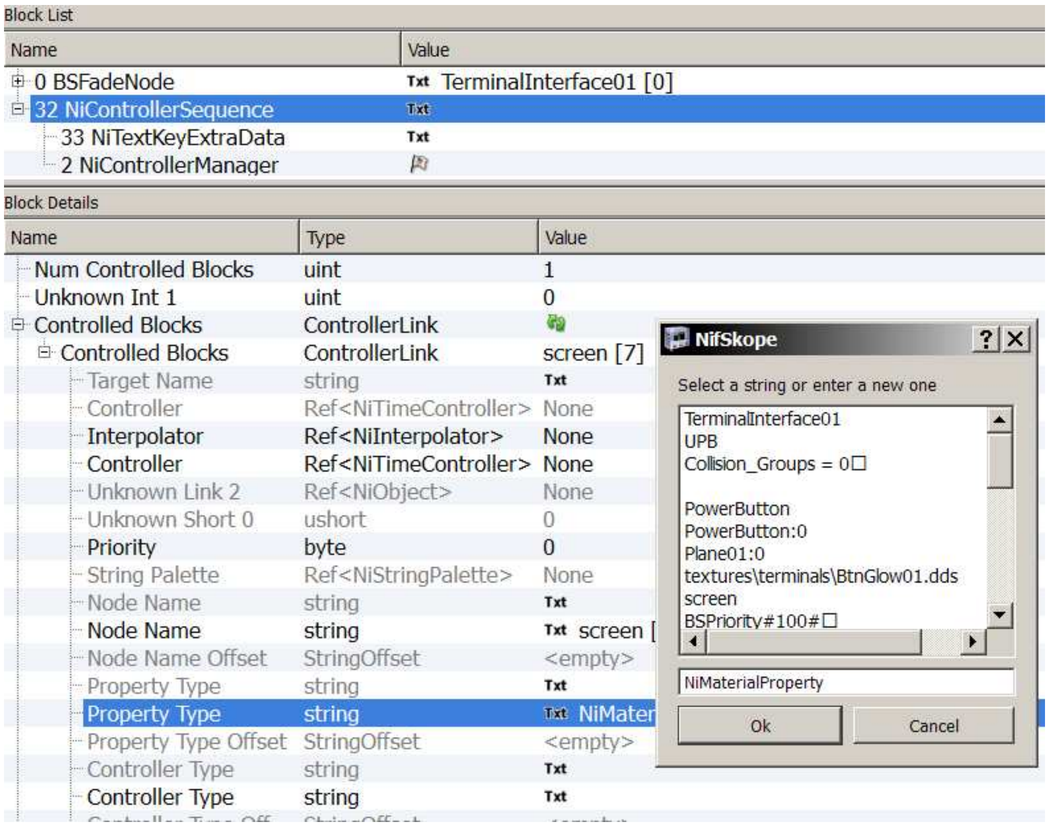
Select a string or enter a new one

```
UPB
Collision_Groups = 0□

PowerButton
PowerButton:0
Plane01:0
textures\terminals\BtnGlow01.dds
screen
BSPriority#100#□
□
```

screen

Ok Cancel

1C. Define the property you're going to animate.
In **Property Type** Value insert the following string: NiMaterialProperty



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
32 NiControllerSequence	Txt
33 NiTextKeyExtraData	Txt
2 NiControllerManager	

Block Details

Name	Type	Value
Num Controlled Blocks	uint	1
Unknown Int 1	uint	0
Controlled Blocks	ControllerLink	
Controlled Blocks	ControllerLink	screen [7]
Target Name	string	Txt
Controller	Ref<NiTimeController>	None
Interpolator	Ref<NiInterpolator>	None
Controller	Ref<NiTimeController>	None
Unknown Link 2	Ref<NiObject>	None
Unknown Short 0	ushort	0
Priority	byte	0
String Palette	Ref<NiStringPalette>	None
Node Name	string	Txt
Node Name	string	Txt screen [
Node Name Offset	StringOffset	<empty>
Property Type	string	Txt
Property Type	string	Txt NiMater
Property Type Offset	StringOffset	<empty>
Controller Type	string	Txt
Controller Type	string	Txt

NifSkope

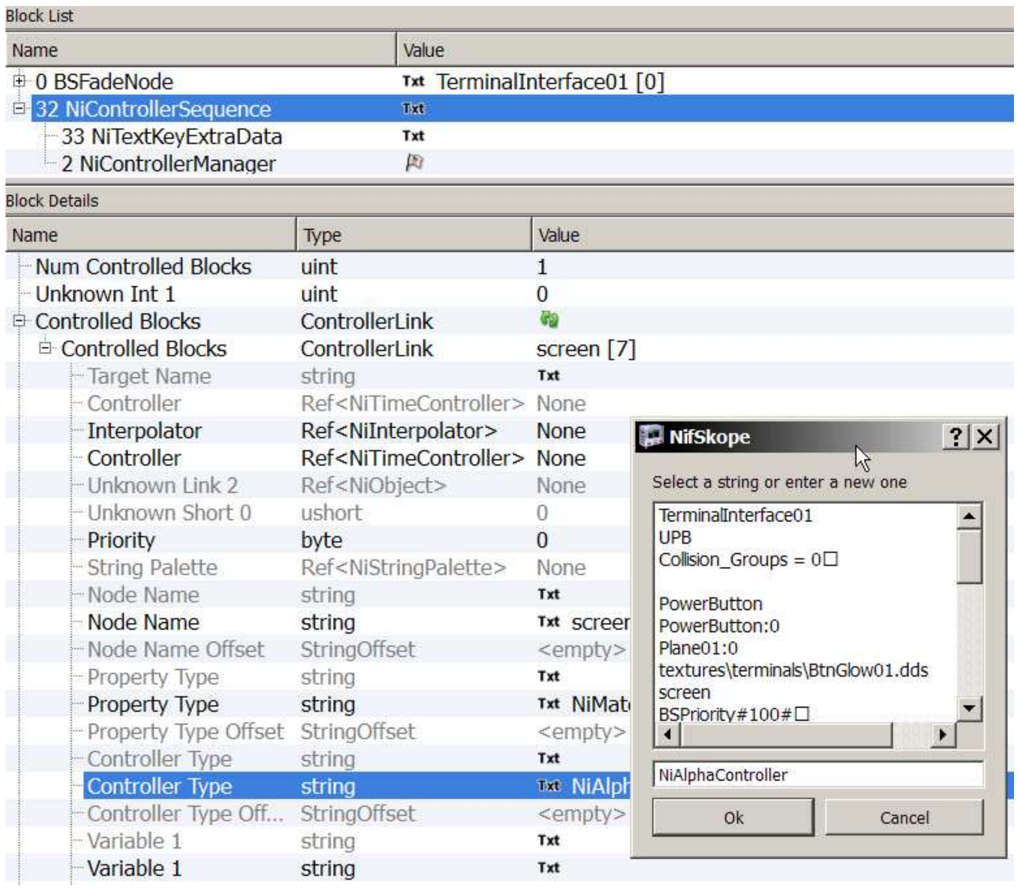
Select a string or enter a new one

```
TerminalInterface01
UPB
Collision_Groups = 0□

PowerButton
PowerButton:0
Plane01:0
textures\terminals\BtnGlow01.dds
screen
BSPriority#100#□
```

NiMaterialProperty

Ok Cancel

1D. Define the Controller type you're going to use.
In **Controller Type** Value insert the following string: NiAlphaController



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
32 NiControllerSequence	Txt
33 NiTextKeyExtraData	Txt
2 NiControllerManager	

Block Details

Name	Type	Value
Num Controlled Blocks	uint	1
Unknown Int 1	uint	0
Controlled Blocks	ControllerLink	
Controlled Blocks	ControllerLink	screen [7]
Target Name	string	Txt
Controller	Ref<NiTimeController>	None
Interpolator	Ref<NiInterpolator>	None
Controller	Ref<NiTimeController>	None
Unknown Link 2	Ref<NiObject>	None
Unknown Short 0	ushort	0
Priority	byte	0
String Palette	Ref<NiStringPalette>	None
Node Name	string	Txt
Node Name	string	Txt screer
Node Name Offset	StringOffset	<empty>
Property Type	string	Txt
Property Type	string	Txt NiMat
Property Type Offset	StringOffset	<empty>
Controller Type	string	Txt
Controller Type	string	Txt NiAlph
Controller Type Off...	StringOffset	<empty>
Variable 1	string	Txt
Variable 1	string	Txt

NifSkope

Select a string or enter a new one

```
TerminalInterface01
UPB
Collision_Groups = 0□

PowerButton
PowerButton:0
Plane01:0
textures\terminals\BtnGlow01.dds
screen
BSPriority#100#□
```

NiAlphaController

Ok Cancel

Let’s start by building an **ALPHA ANIMATION BLOCK**

2. Building an Alpha animation block. Select the NiControllerSequence.

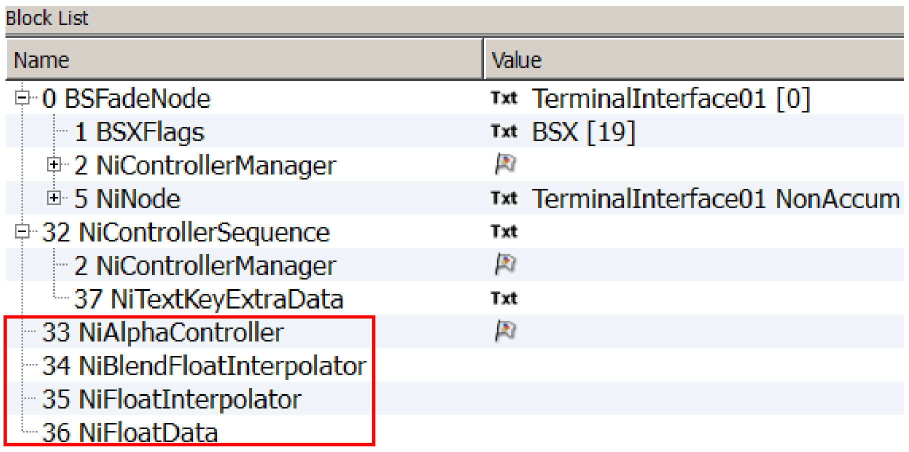
You will need to insert four elements.

- A NiAlphaController:
Block ➪ Insert ➪ NiA ➪ NiAlphaController

- A NiBlendFloatInterpolator:
Block ➪ Insert ➪ NiB ➪ NiBlendFloatInterpolator

- A NiFloatInterpolator:
Block ➪ Insert ➪ NiF ➪ NiFloatInterpolator

- A NiFloatData:
Block ➪ Insert ➪ NiF ➪ NiFloatData

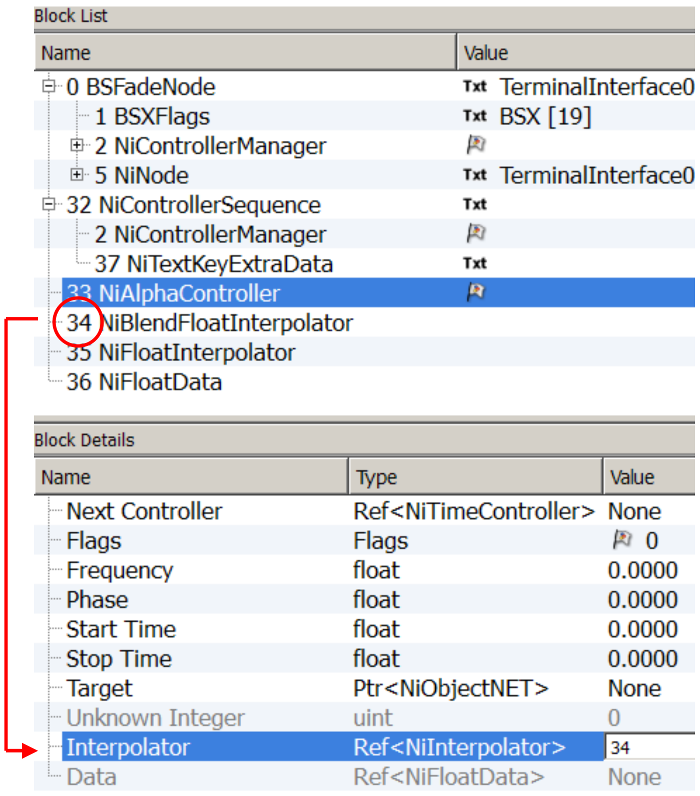

Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
5 NiNode	Txt TerminalInterface01 NonAccum
32 NiControllerSequence	Txt
2 NiControllerManager	
37 NiTextKeyExtraData	Txt
33 NiAlphaController	
34 NiBlendFloatInterpolator	
35 NiFloatInterpolator	
36 NiFloatData	

3. Linking the elements.

The NiBlendFloatInterpolator will be linked to the NiAlphaController.

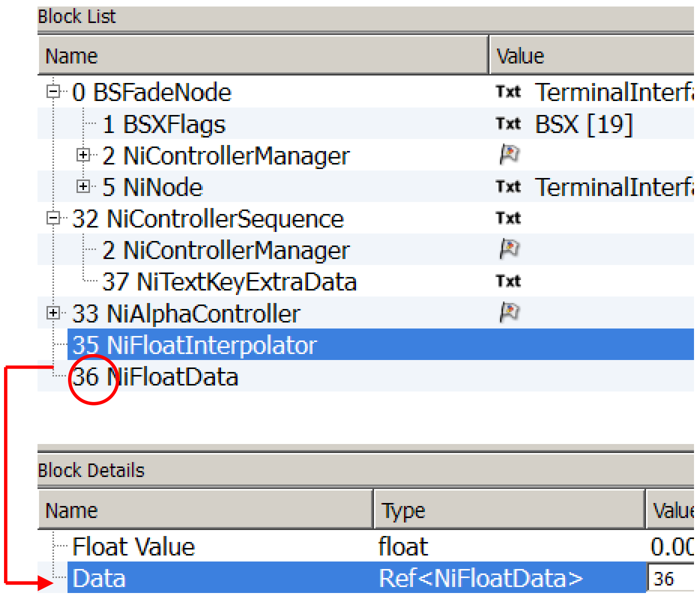
The NiFloatData will be linked to the NiFloatInterpolator.


Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface0
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
5 NiNode	Txt TerminalInterface0
32 NiControllerSequence	Txt
2 NiControllerManager	
37 NiTextKeyExtraData	Txt
33 NiAlphaController	
34 NiBlendFloatInterpolator	
35 NiFloatInterpolator	
36 NiFloatData	

Block Details

Name	Type	Value
Next Controller	Ref<NiTimeController>	None
Flags	Flags	0
Frequency	float	0.0000
Phase	float	0.0000
Start Time	float	0.0000
Stop Time	float	0.0000
Target	Ptr<NiObjectNET>	None
Unknown Integer	uint	0
Interpolator	Ref<NiInterpolator>	34
Data	Ref<NiFloatData>	None


Block List

Name	Value
0 BSFadeNode	Txt TerminalInterf
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
5 NiNode	Txt TerminalInterf
32 NiControllerSequence	Txt
2 NiControllerManager	
37 NiTextKeyExtraData	Txt
33 NiAlphaController	
35 NiFloatInterpolator	
36 NiFloatData	

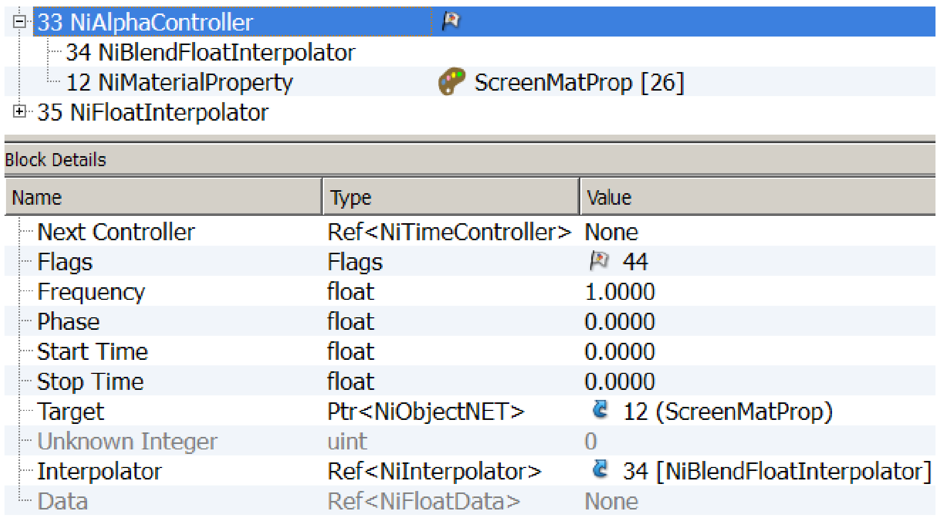
Block Details

Name	Type	Value
Float Value	float	0.00
Data	Ref<NiFloatData>	36

4. Adjusting the settings.

The NiAlphaController:

Flags	40 for loop, 44 for Clamp
Frequency	1.0000
Target	The **NiMaterialProperty** of the NiTriStrips you're animating



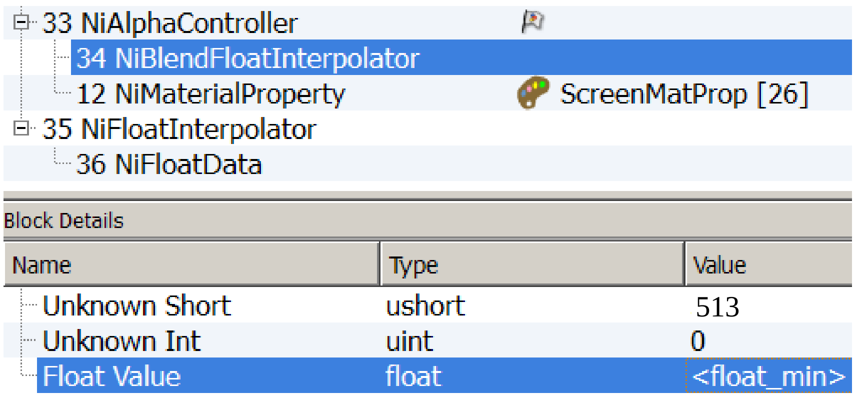
```
33 NiAlphaController
  34 NiBlendFloatInterpolator
  12 NiMaterialProperty          ScreenMatProp [26]
35 NiFloatInterpolator
```

Block Details

Name	Type	Value
Next Controller	Ref<NiTimeController>	None
Flags	Flags	44
Frequency	float	1.0000
Phase	float	0.0000
Start Time	float	0.0000
Stop Time	float	0.0000
Target	Ptr<NiObjectNET>	12 (ScreenMatProp)
Unknown Integer	uint	0
Interpolator	Ref<NiInterpolator>	34 [NiBlendFloatInterpolator]
Data	Ref<NiFloatData>	None

The NiBlendFloatInterpolator:

Unknown Short	513
Float Value	<float_min>



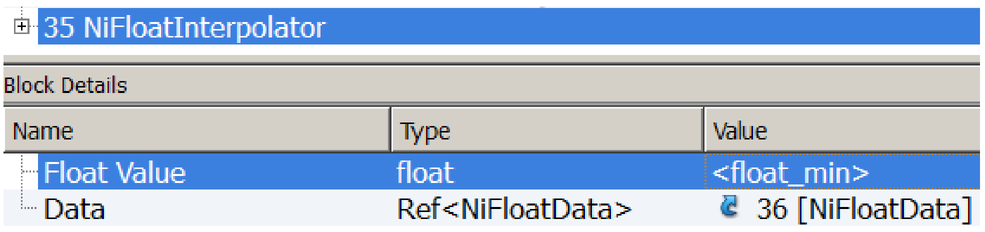
```
33 NiAlphaController
  34 NiBlendFloatInterpolator
  12 NiMaterialProperty          ScreenMatProp [26]
35 NiFloatInterpolator
  36 NiFloatData
```

Block Details

Name	Type	Value
Unknown Short	ushort	513
Unknown Int	uint	0
Float Value	float	<float_min>

The NiFloatInterpolator:

Float Value	<float_min>



```
35 NiFloatInterpolator
```

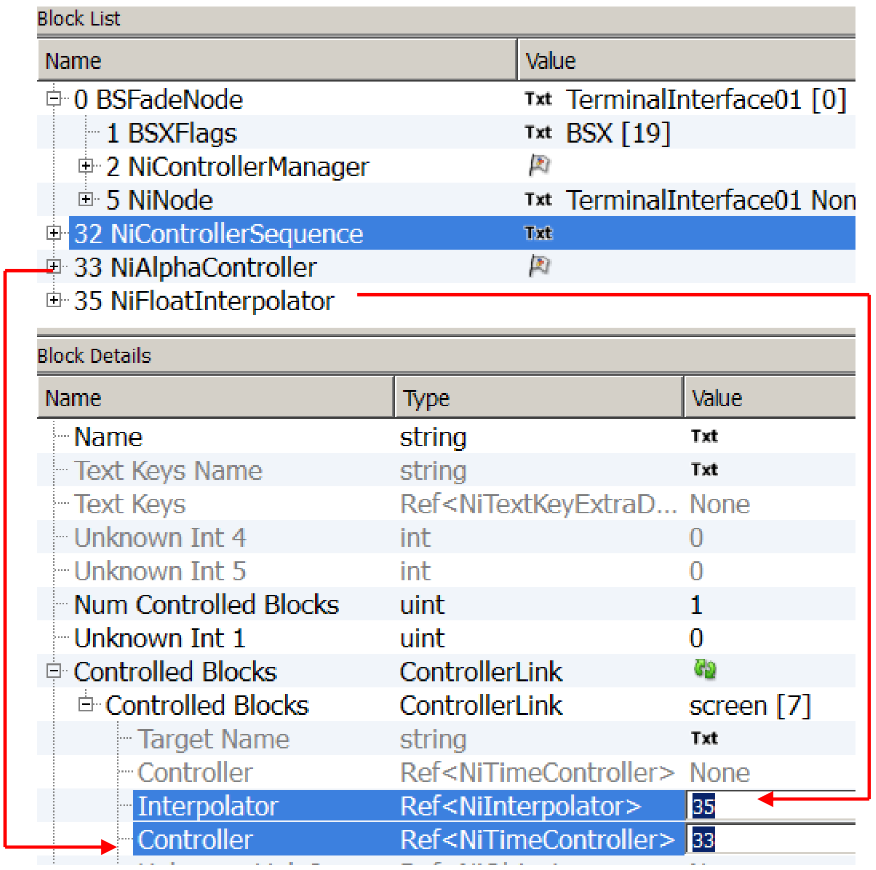
Block Details

Name	Type	Value
Float Value	float	<float_min>
Data	Ref<NiFloatData>	36 [NiFloatData]

The NiFloatData (the keys) will be set later.

These two sub-blocks (Controller and Interpolator) will be referenced inside the NiControllerSequence.

5. Select the NiControllerSequence; go to your Controlled Blocks and insert the values of the NiAlphaController and the NiFloatInterpolator in their respective Value Fields.



Block List

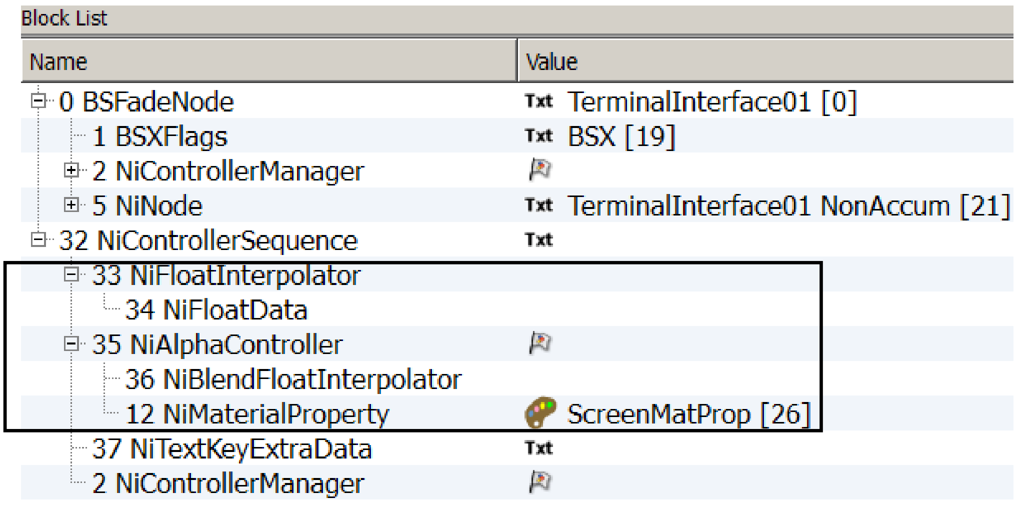
Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
5 NiNode	Txt TerminalInterface01 Non
32 NiControllerSequence	Txt
33 NiAlphaController	
35 NiFloatInterpolator	

Block Details

Name	Type	Value
Name	string	Txt
Text Keys Name	string	Txt
Text Keys	Ref<NiTextKeyExtraD...	None
Unknown Int 4	int	0
Unknown Int 5	int	0
Num Controlled Blocks	uint	1
Unknown Int 1	uint	0
Controlled Blocks	ControllerLink	
Controlled Blocks	ControllerLink	screen [7]
Target Name	string	Txt
Controller	Ref<NiTimeController>	None
Interpolator	Ref<NiInterpolator>	35
Controller	Ref<NiTimeController>	33

After blocks reorganisation, you should have this result:

an Alpha animation block inside a sequence.

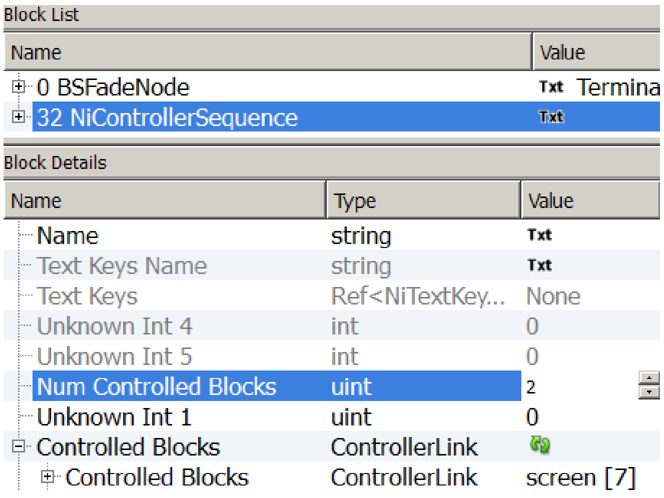


Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
5 NiNode	Txt TerminalInterface01 NonAccum [21]
32 NiControllerSequence	Txt
33 NiFloatInterpolator	
34 NiFloatData	
35 NiAlphaController	
36 NiBlendFloatInterpolator	
12 NiMaterialProperty	ScreenMatProp [26]
37 NiTextKeyExtraData	Txt
2 NiControllerManager	

Let's go further and build an **EMISSIVE ANIMATION BLOCK** for the Power Button.

6. Go to the NiControllerSequence, in block details: increase the **Num Controlled Blocks** from 1 unit. Update.



Block List

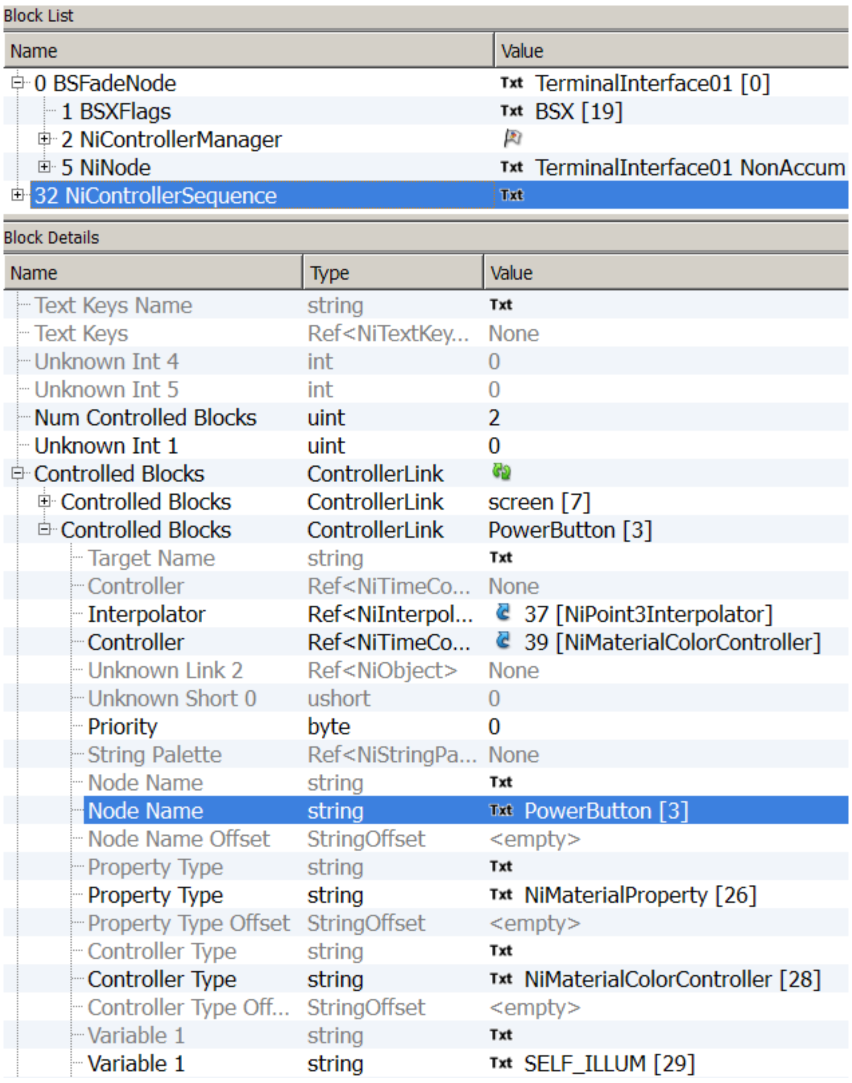
Name	Value
0 BSFadeNode	Txt Termina
32 NiControllerSequence	Txt

Block Details

Name	Type	Value
Name	string	Txt
Text Keys Name	string	Txt
Text Keys	Ref<NiTextKey...	None
Unknown Int 4	int	0
Unknown Int 5	int	0
Num Controlled Blocks	uint	2
Unknown Int 1	uint	0
Controlled Blocks	ControllerLink	
Controlled Blocks	ControllerLink	screen [7]

7. Expand and define the new controlled block.

- In **Node Name**: enter the name of the NiTriStrips you're planning to animate.
- In **Property Type** Value insert the following string: NiMaterialProperty
- In **Controller Type** Value insert the following string: NiMaterialColorController
- In Variable 1 Value Insert SELF_ILLUM string.



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
5 NiNode	Txt TerminalInterface01 NonAccum
32 NiControllerSequence	Txt

Block Details

Name	Type	Value
Text Keys Name	string	Txt
Text Keys	Ref<NiTextKey...	None
Unknown Int 4	int	0
Unknown Int 5	int	0
Num Controlled Blocks	uint	2
Unknown Int 1	uint	0
Controlled Blocks	ControllerLink	
Controlled Blocks	ControllerLink	screen [7]
Controlled Blocks	ControllerLink	PowerButton [3]
Target Name	string	Txt
Controller	Ref<NiTimeCo...	None
Interpolator	Ref<NiInterpol...	37 [NiPoint3Interpolator]
Controller	Ref<NiTimeCo...	39 [NiMaterialColorController]
Unknown Link 2	Ref<NiObject>	None
Unknown Short 0	ushort	0
Priority	byte	0
String Palette	Ref<NiStringPa...	None
Node Name	string	Txt
Node Name	string	Txt PowerButton [3]
Node Name Offset	StringOffset	<empty>
Property Type	string	Txt
Property Type	string	Txt NiMaterialProperty [26]
Property Type Offset	StringOffset	<empty>
Controller Type	string	Txt
Controller Type	string	Txt NiMaterialColorController [28]
Controller Type Off...	StringOffset	<empty>
Variable 1	string	Txt
Variable 1	string	Txt SELF_ILLUM [29]

8. Building an Emissive animation block, four elements needed.

A NiMaterialColorController
A NiBlendPoint3Interpolator
A NiPoint3Interpolator
A NiPosData

For some reason I never manage to set the NiBlendPoint3Interpolator and the NiPoint3Interpolator parameters to <Float_Min> the usual way.

So the workaround would be to copy these, correctly set, from another mesh.

Open DemoCameraMonitor.nif in a second occurrence of Nifskope. Locate a NiBlendPoint3Interpolator and a NiPoint3Interpolator, any one will do.

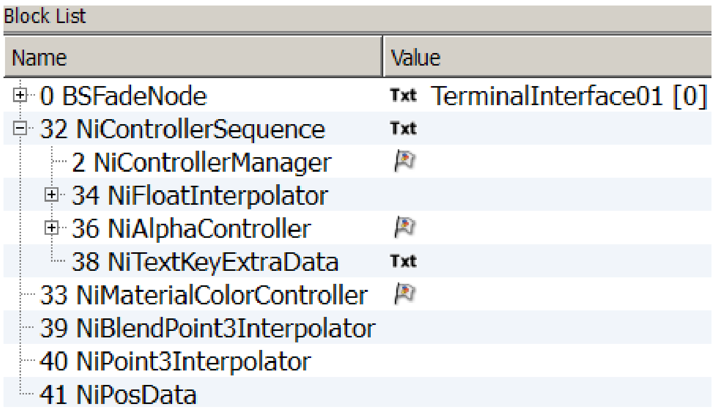
Follow these steps:

A. In your mesh: Block ➪ Insert ➪ NiM ➪ NiMaterialColorController

B. In the 2nd mesh: NiBlendPoint3Interpolator ➪ Block ➪ Copy
In your mesh: ➪ Block ➪ Paste

C. In the 2nd mesh: NiPoint3Interpolator ➪ Block ➪ Copy
In your mesh: ➪ Block ➪ Paste (this one may retain a Data Value from the base mesh, don't bother, you'll change that the next step)

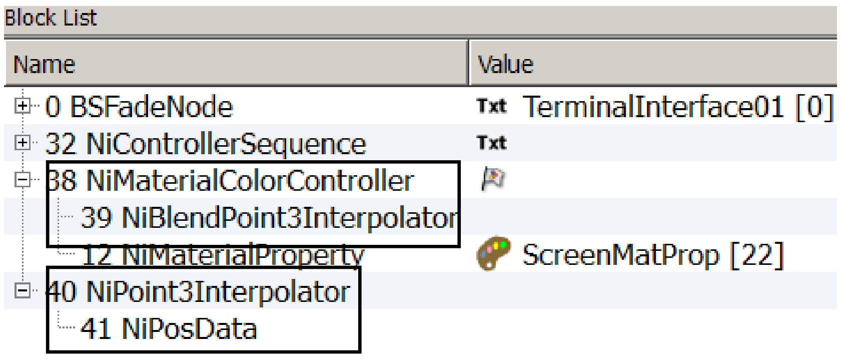
D. In your mesh: Block ➪ Insert ➪ NiP ➪ NiPosData


Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
32 NiControllerSequence	Txt
2 NiControllerManager	
34 NiFloatInterpolator	
36 NiAlphaController	
38 NiTextKeyExtraData	Txt
33 NiMaterialColorController	
39 NiBlendPoint3Interpolator	
40 NiPoint3Interpolator	
41 NiPosData	

9. Linking the elements. *(Just as we did previously, point 3).*

- The NiBlendPoint3Interpolator will be linked to the NiMaterialColorController.
- The new NiPosData will be linked to the NiPoint3Interpolator.


Block List

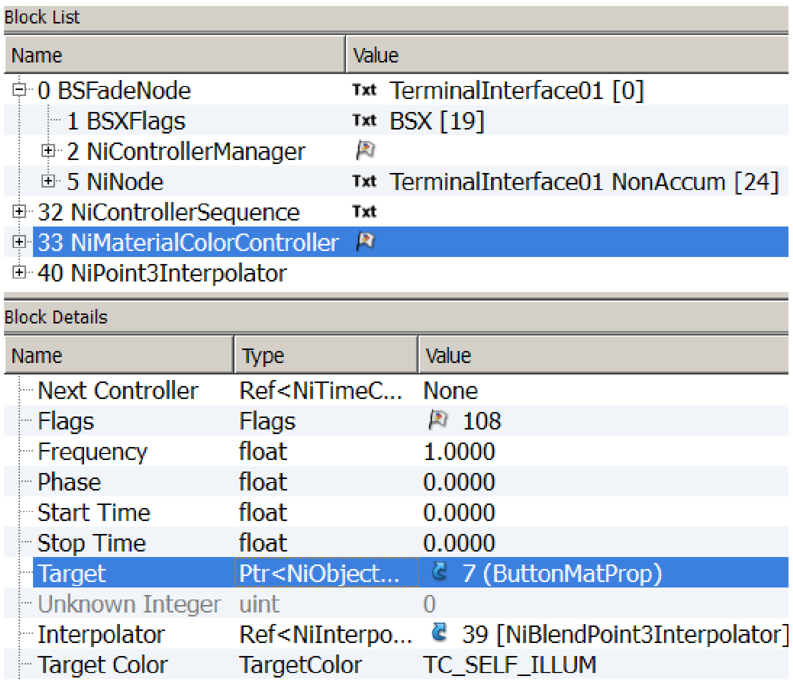
Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
32 NiControllerSequence	Txt
38 NiMaterialColorController	
39 NiBlendPoint3Interpolator	
12 NiMaterialProperty	ScreenMatProp [22]
40 NiPoint3Interpolator	
41 NiPosData	

Reorder blocks with Spells ➪ Sanitize ➪ Reorder Blocks.

10. Adjusting the settings.

The NiMaterialColorController.

Flags	108
Frequency	1.0000
Target	The NiMaterialProperty of the NiTriStrips you're animating
Target Color	TC_SELF_ILLUM



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
5 NiNode	Txt TerminalInterface01 NonAccum [24]
32 NiControllerSequence	Txt
33 NiMaterialColorController	
40 NiPoint3Interpolator	

Block Details

Name	Type	Value
Next Controller	Ref<NiTimeC...	None
Flags	Flags	108
Frequency	float	1.0000
Phase	float	0.0000
Start Time	float	0.0000
Stop Time	float	0.0000
Target	Ptr<NiObject...	7 (ButtonMatProp)
Unknown Integer	uint	0
Interpolator	Ref<NiInterpo...	39 [NiBlendPoint3Interpolator]
Target Color	TargetColor	TC_SELF_ILLUM

The NiBlendPoint3Interpolator.

Unknown Short	513
Point Value	X <float_min> Y <float_min> Z <float_min>

The NiPoint3Interpolator

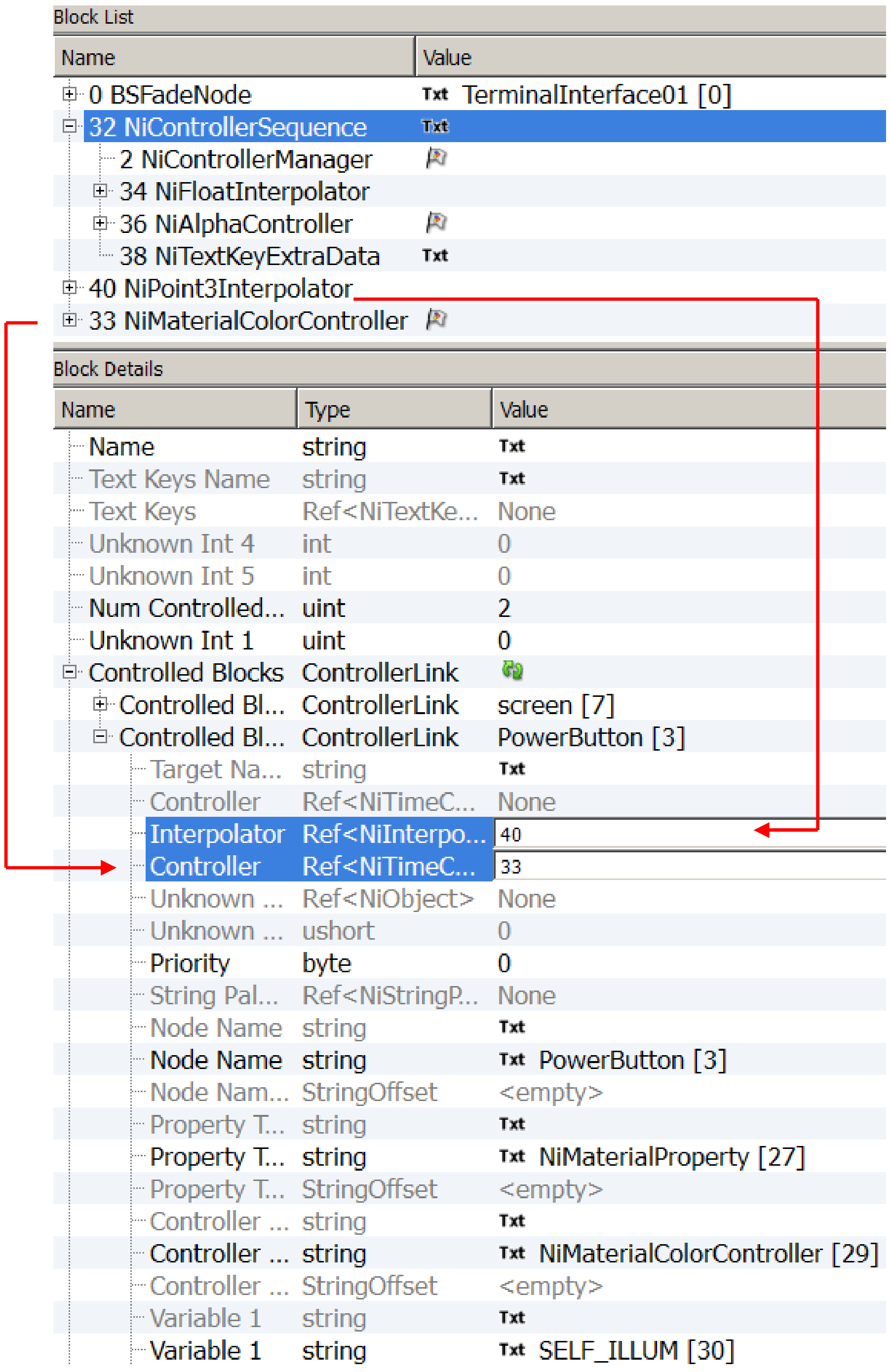
Point 3 Value	X <float_min> Y <float_min> Z <float_min>

Both should be correctly set from copy.

The NiFloatData (the keys) will be set later.

Again these two sub-blocks (Controller and Interpolator) need to be referenced inside the NiControllerSequence. *(Like in step 5)*

11. In the NiControllerSequence; go to your new Controlled Blocks and insert the values of the NiMaterialColorController and the NiPoint3Interpolator in their respective Value Fields.



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
32 NiControllerSequence	Txt
2 NiControllerManager	
34 NiFloatInterpolator	
36 NiAlphaController	
38 NiTextKeyExtraData	Txt
40 NiPoint3Interpolator	
33 NiMaterialColorController	

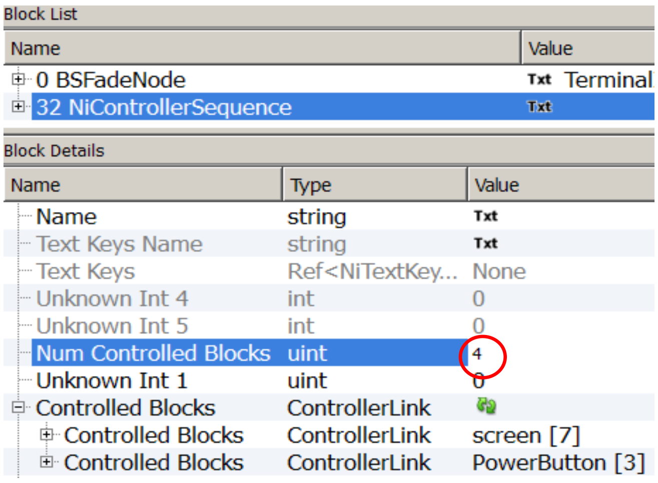
Block Details

Name	Type	Value
Name	string	Txt
Text Keys Name	string	Txt
Text Keys	Ref<NiTextKe...	None
Unknown Int 4	int	0
Unknown Int 5	int	0
Num Controlled...	uint	2
Unknown Int 1	uint	0
Controlled Blocks	ControllerLink	
Controlled Bl...	ControllerLink	screen [7]
Controlled Bl...	ControllerLink	PowerButton [3]
Target Na...	string	Txt
Controller	Ref<NiTimeC...	None
Interpolator	Ref<NiInterpo...	40
Controller	Ref<NiTimeC...	33
Unknown ...	Ref<NiObject>	None
Unknown ...	ushort	0
Priority	byte	0
String Pal...	Ref<NiStringP...	None
Node Name	string	Txt
Node Name	string	Txt PowerButton [3]
Node Nam...	StringOffset	<empty>
Property T...	string	Txt
Property T...	string	Txt NiMaterialProperty [27]
Property T...	StringOffset	<empty>
Controller ...	string	Txt
Controller ...	string	Txt NiMaterialColorController [29]
Controller ...	StringOffset	<empty>
Variable 1	string	Txt
Variable 1	string	Txt SELF_ILLUM [30]

And finally let's build an **UV TRANSLATE ANIMATION BLOCK,** for the screen.

Two blocks are needed (one for U, one for V), since their structure is the same, apart from one setting, we can build one, duplicate it and set the respective settings after.

12. Go to the NiControllerSequence, in block details: increase the **Num Controlled Blocks** from two units. Update.



Block List

Name	Value
0 BSFadeNode	Txt Terminal
32 NiControllerSequence	Txt

Block Details

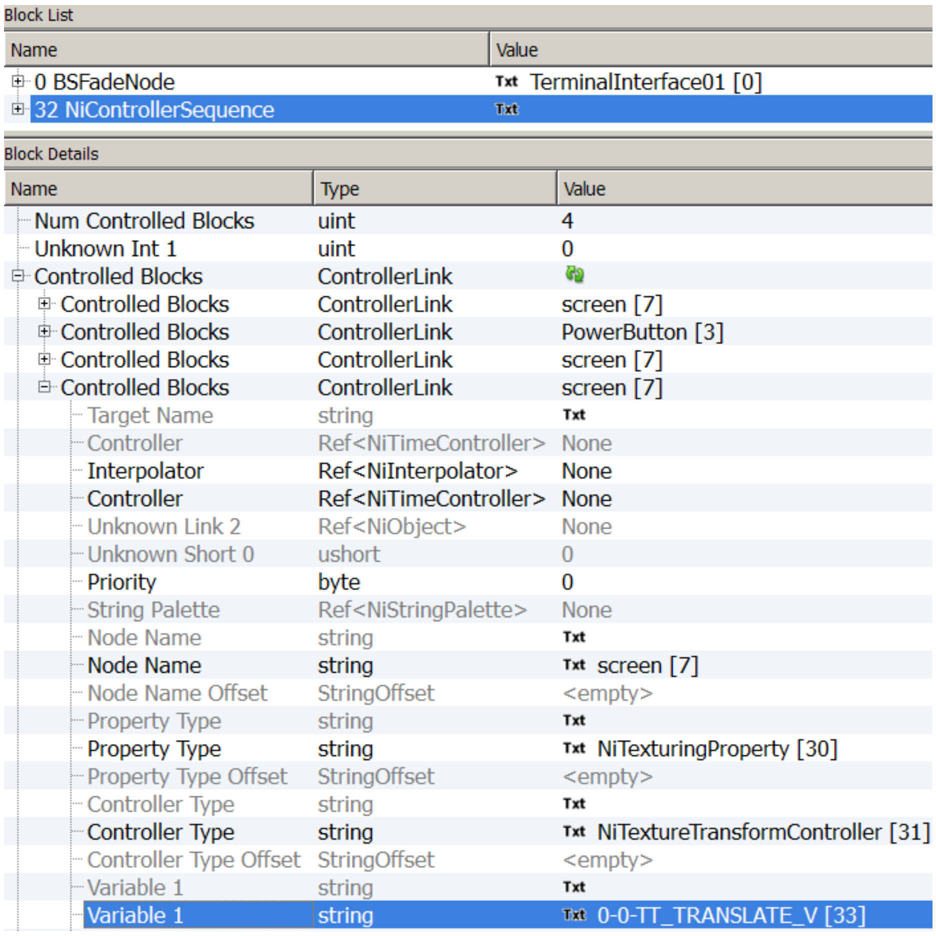
Name	Type	Value
Name	string	Txt
Text Keys Name	string	Txt
Text Keys	Ref<NiTextKey...	None
Unknown Int 4	int	0
Unknown Int 5	int	0
Num Controlled Blocks	uint	4
Unknown Int 1	uint	0
Controlled Blocks	ControllerLink	
Controlled Blocks	ControllerLink	screen [7]
Controlled Blocks	ControllerLink	PowerButton [3]

13. Define the two new controlled blocks.

- In **Node Name**: enter the name of the node you're planning to animate.
- In **Property Type** Value insert the following string: NiTexturingProperty
- In **Controller Type** Value insert the following string: NiTextureTransformController

In the first new Controlled Bock enter 0-0-TT_TRANSLATE_U in the **Variable 1**

In the second Controlled Block enter 0-0-TT_TRANSLATE_V in the **Variable 1** (as pictured).



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
32 NiControllerSequence	Txt

Block Details

Name	Type	Value
Num Controlled Blocks	uint	4
Unknown Int 1	uint	0
Controlled Blocks	ControllerLink	
Controlled Blocks	ControllerLink	screen [7]
Controlled Blocks	ControllerLink	PowerButton [3]
Controlled Blocks	ControllerLink	screen [7]
Controlled Blocks	ControllerLink	screen [7]
Target Name	string	Txt
Controller	Ref<NiTimeController>	None
Interpolator	Ref<NiInterpolator>	None
Controller	Ref<NiTimeController>	None
Unknown Link 2	Ref<NiObject>	None
Unknown Short 0	ushort	0
Priority	byte	0
String Palette	Ref<NiStringPalette>	None
Node Name	string	Txt
Node Name	string	Txt screen [7]
Node Name Offset	StringOffset	<empty>
Property Type	string	Txt
Property Type	string	Txt NiTexturingProperty [30]
Property Type Offset	StringOffset	<empty>
Controller Type	string	Txt
Controller Type	string	Txt NiTextureTransformController [31]
Controller Type Offset	StringOffset	<empty>
Variable 1	string	Txt
Variable 1	string	Txt 0-0-TT_TRANSLATE_V [33]

14. Building an UV animation block. Again, you will need to insert four elements:

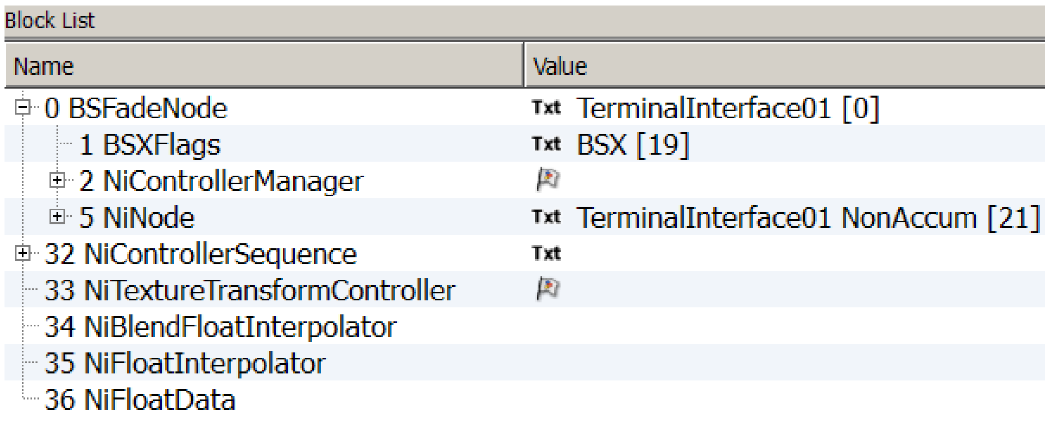
A NiTextureTransformController:

Block ➪ Insert ➪ NiT ➪ NiTextureTransformController

A NiBlendFloatInterpolator:
Block ➪ Insert ➪ NiB ➪ NiBlendFloatInterpolator

A NiFloatInterpolator:
Block ➪ Insert ➪ NiF ➪ NiFloatInterpolator

A NiFloatData:
Block ➪ Insert ➪ NiF ➪ NiFloatData



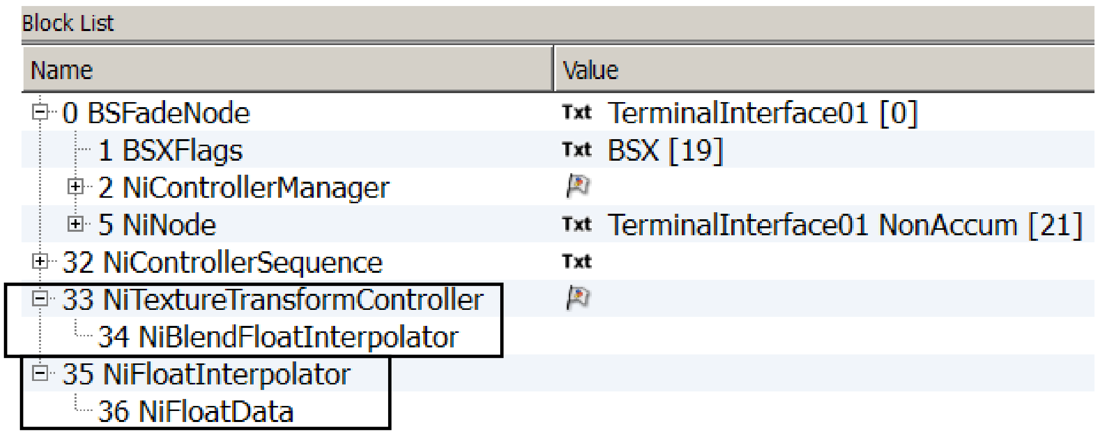
Block List

Name	Value
⊟ 0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
⊞ 2 NiControllerManager	
⊞ 5 NiNode	Txt TerminalInterface01 NonAccum [21]
⊞ 32 NiControllerSequence	Txt
33 NiTextureTransformController	
34 NiBlendFloatInterpolator	
35 NiFloatInterpolator	
36 NiFloatData	

15. Linking the elements.

The NiBlendFloatInterpolator will be linked to the NiTextureTransformController.

The NiFloatData will be linked to the NiFloatInterpolator.



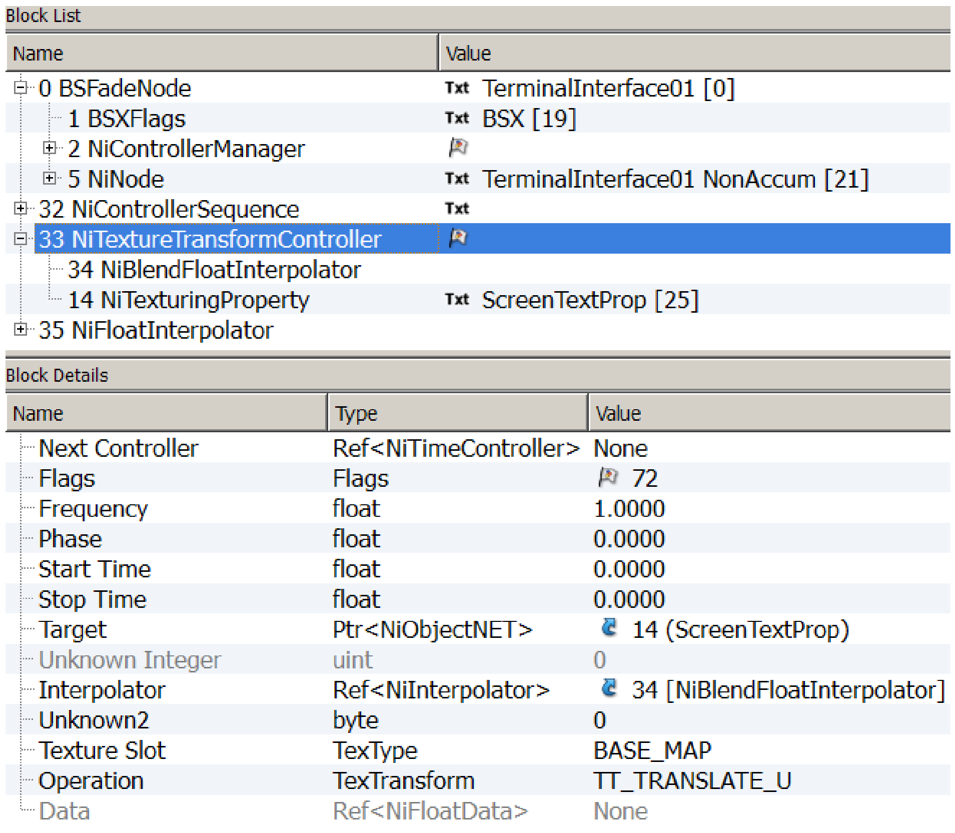
Block List

Name	Value
⊟ 0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
⊞ 2 NiControllerManager	
⊞ 5 NiNode	Txt TerminalInterface01 NonAccum [21]
⊞ 32 NiControllerSequence	Txt
⊟ 33 NiTextureTransformController	
34 NiBlendFloatInterpolator	
⊟ 35 NiFloatInterpolator	
36 NiFloatData	

16. Adjusting the settings.

NiTextureTransformController:

Flags	72
Frequency	1.0000
Target	The NiTexturingProperty of the NiTriStrips you're animating



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
5 NiNode	Txt TerminalInterface01 NonAccum [21]
32 NiControllerSequence	Txt
33 NiTextureTransformController	
34 NiBlendFloatInterpolator	
14 NiTexturingProperty	Txt ScreenTextProp [25]
35 NiFloatInterpolator	

Block Details

Name	Type	Value
Next Controller	Ref<NiTimeController>	None
Flags	Flags	72
Frequency	float	1.0000
Phase	float	0.0000
Start Time	float	0.0000
Stop Time	float	0.0000
Target	Ptr<NiObjectNET>	14 (ScreenTextProp)
Unknown Integer	uint	0
Interpolator	Ref<NiInterpolator>	34 [NiBlendFloatInterpolator]
Unknown2	byte	0
Texture Slot	TexType	BASE_MAP
Operation	TexTransform	TT_TRANSLATE_U
Data	Ref<NiFloatData>	None

The NiBlendFloatInterpolator:

Unknown Short	513
Float Value	<float_min>

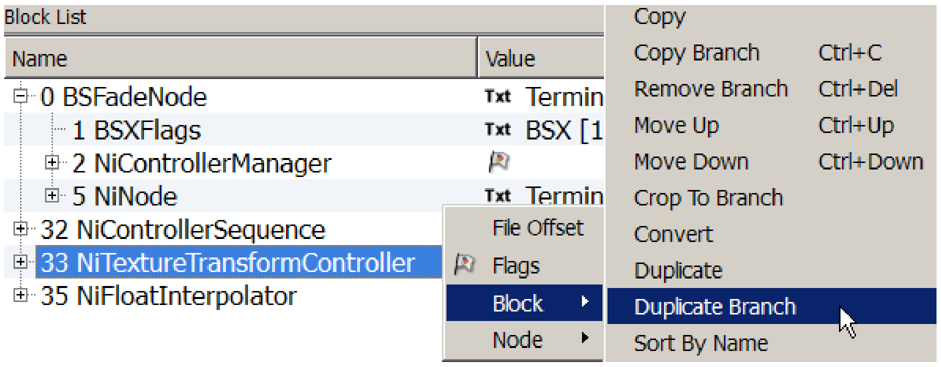
The NiFloatInterpolator:

Float Value	<float_min>

Data keys and Operation Type will be set later.

Since we want Two Texture Transform blocks (U & V), we can use the duplicate function and make a second set of block out of the one we have just built.

17. Select the NiTextureTransformController then right click Block ➪ Duplicate Branch.



Block List

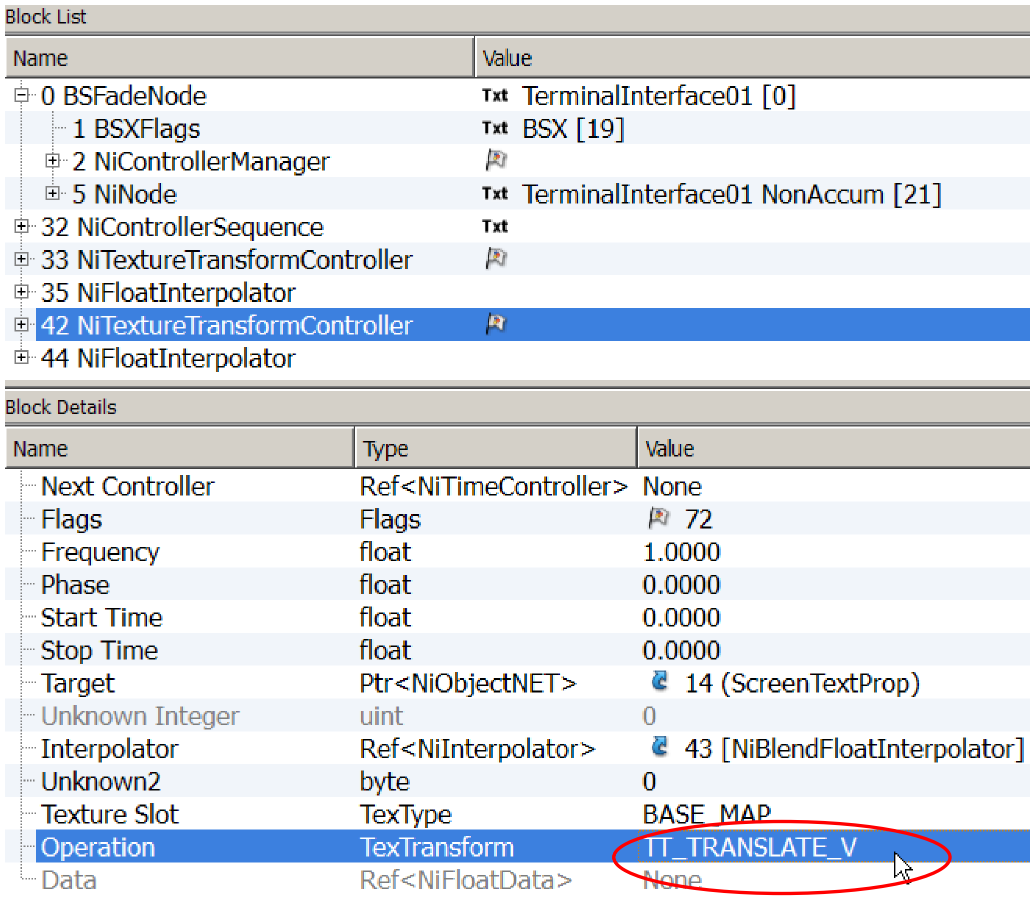
Name	Value
0 BSFadeNode	Txt Termin
1 BSXFlags	Txt BSX [1
2 NiControllerManager	
5 NiNode	Txt Termin
32 NiControllerSequence	
33 NiTextureTransformController	
35 NiFloatInterpolator	

18. Same with the NiFloatInterpolator.

Select the NiFloatInterpolator then right click Block ➪ Duplicate Branch.

19. Let's adjust the specific settings of the **second** NiTextureTransformController.

Set the Operation Value to TT_TRANSLATE_V



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
5 NiNode	Txt TerminalInterface01 NonAccum [21]
32 NiControllerSequence	Txt
33 NiTextureTransformController	
35 NiFloatInterpolator	
42 NiTextureTransformController	
44 NiFloatInterpolator	

Block Details

Name	Type	Value
Next Controller	Ref<NiTimeController>	None
Flags	Flags	72
Frequency	float	1.0000
Phase	float	0.0000
Start Time	float	0.0000
Stop Time	float	0.0000
Target	Ptr<NiObjectNET>	14 (ScreenTextProp)
Unknown Integer	uint	0
Interpolator	Ref<NiInterpolator>	43 [NiBlendFloatInterpolator]
Unknown2	byte	0
Texture Slot	TexType	BASE_MAP
Operation	TexTransform	TT_TRANSLATE_V
Data	Ref<NiFloatData>	None

The **first** NiTextureTransformController should be on TT_TRANSLATE_U by default.

We will now reference these sub blocks in the sequence. *(Like we did in step 5 and 11.)*

20. Select the NiControllerSequence. Go to the third Controlled Block.

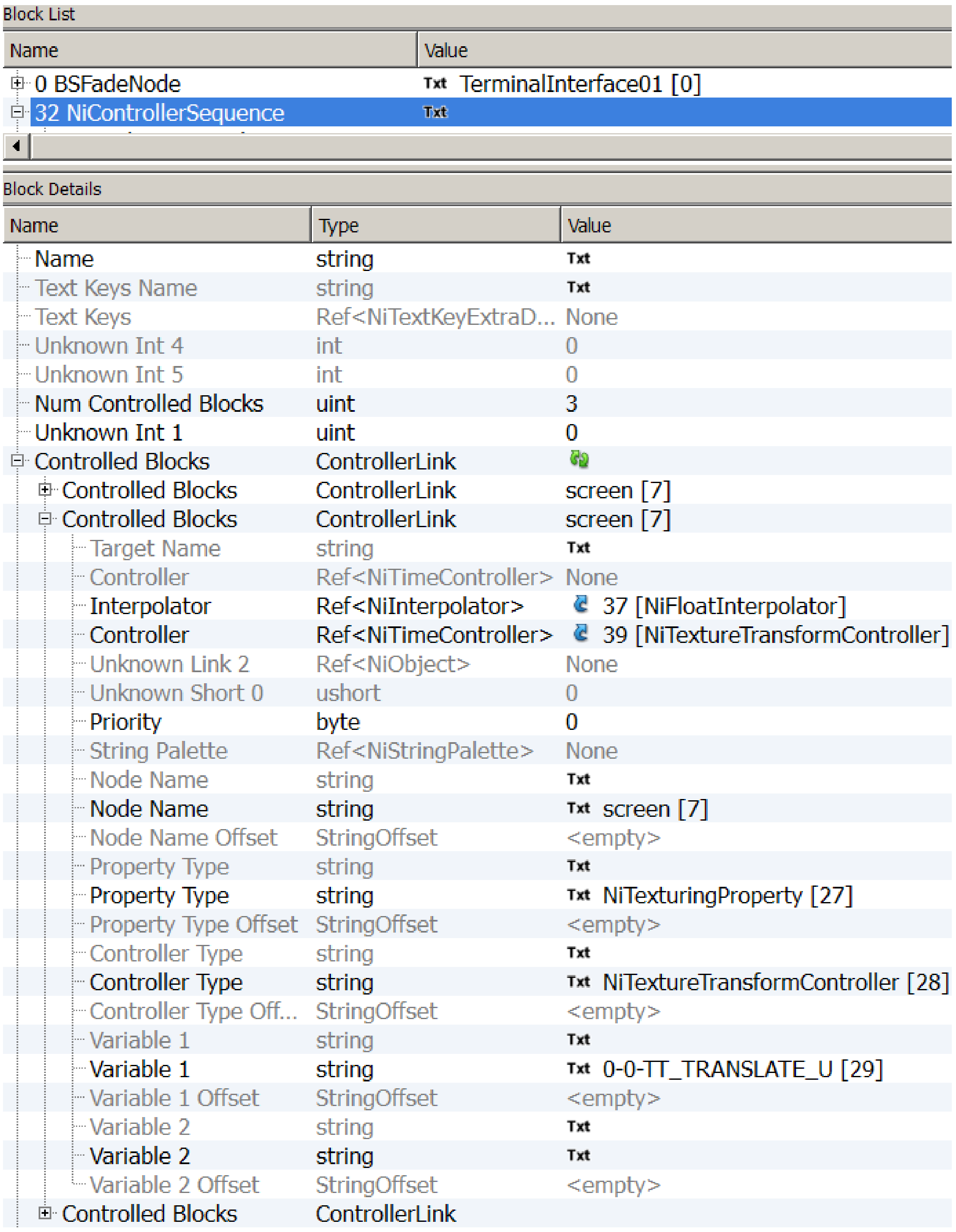
The **U Controlled Block**

- As Interpolator set the value of the **first of the two** NiFloatInterpolator you've created.
- As Controller set the value of the **first of the two** NiTextureTransformController you've created.

The **V Controlled Block**

- As Interpolator set the value of the **second** NiFloatInterpolator you've created.
- As Controller set the value of the **second** NiTextureTransformController you've created.

Illustration for the U Controlled Block.



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
32 NiControllerSequence	Txt

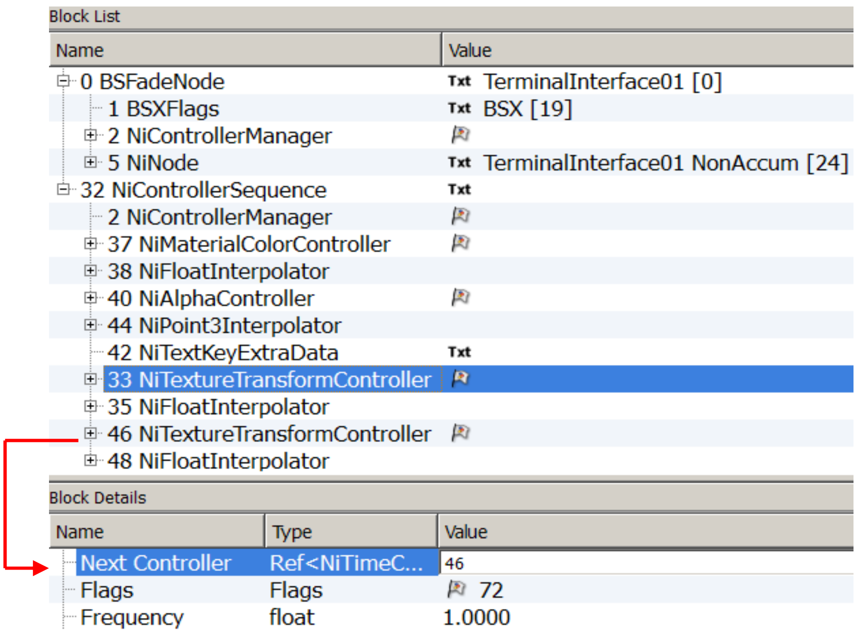
Block Details

Name	Type	Value
Name	string	Txt
Text Keys Name	string	Txt
Text Keys	Ref<NiTextKeyExtraD...	None
Unknown Int 4	int	0
Unknown Int 5	int	0
Num Controlled Blocks	uint	3
Unknown Int 1	uint	0
Controlled Blocks	ControllerLink	
Controlled Blocks	ControllerLink	screen [7]
Controlled Blocks	ControllerLink	screen [7]
Target Name	string	Txt
Controller	Ref<NiTimeController>	None
Interpolator	Ref<NiInterpolator>	37 [NiFloatInterpolator]
Controller	Ref<NiTimeController>	39 [NiTextureTransformController]
Unknown Link 2	Ref<NiObject>	None
Unknown Short 0	ushort	0
Priority	byte	0
String Palette	Ref<NiStringPalette>	None
Node Name	string	Txt
Node Name	string	Txt screen [7]
Node Name Offset	StringOffset	<empty>
Property Type	string	Txt
Property Type	string	Txt NiTexturingProperty [27]
Property Type Offset	StringOffset	<empty>
Controller Type	string	Txt
Controller Type	string	Txt NiTextureTransformController [28]
Controller Type Off...	StringOffset	<empty>
Variable 1	string	Txt
Variable 1	string	Txt 0-0-TT_TRANSLATE_U [29]
Variable 1 Offset	StringOffset	<empty>
Variable 2	string	Txt
Variable 2	string	Txt
Variable 2 Offset	StringOffset	<empty>
Controlled Blocks	ControllerLink	

21. Link the two Translate NiTextureTransformController together as they act as a pair.

In block List, expand the NiControllerSequence and

- Select the first of your NiTextureTransformController.
- Set the value of the second NiTextureTransformController as Next Controller.



Block List

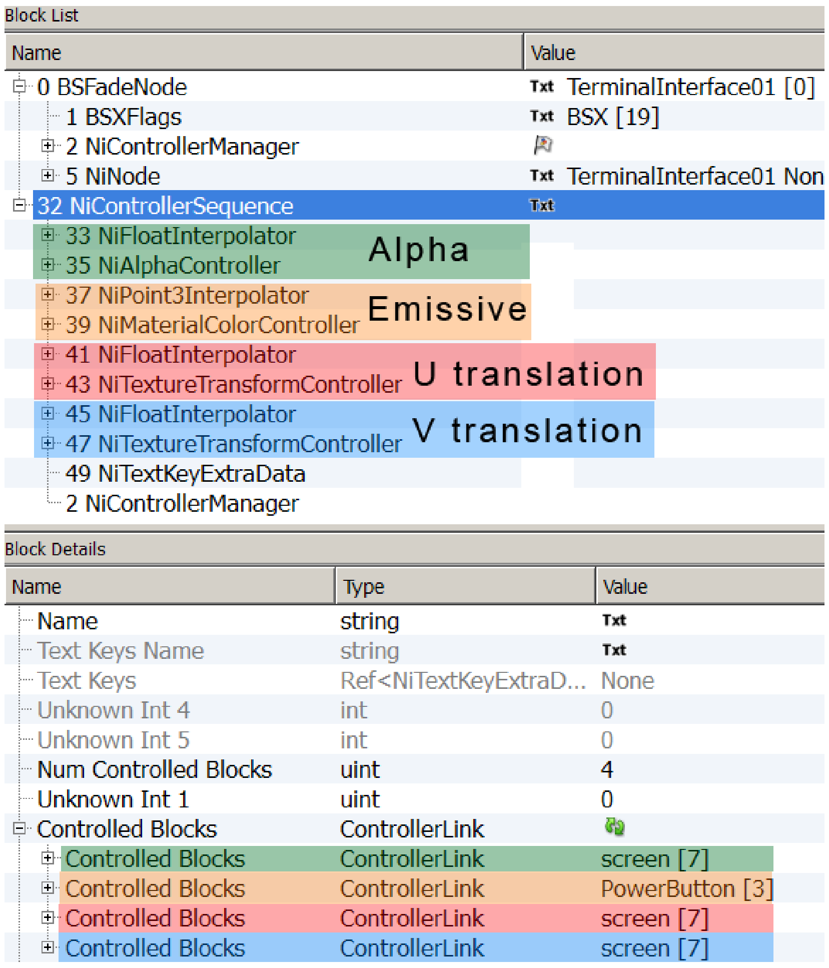
Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
5 NiNode	Txt TerminalInterface01 NonAccum [24]
32 NiControllerSequence	Txt
2 NiControllerManager	
37 NiMaterialColorController	
38 NiFloatInterpolator	
40 NiAlphaController	
44 NiPoint3Interpolator	
42 NiTextKeyExtraData	Txt
33 NiTextureTransformController	
35 NiFloatInterpolator	
46 NiTextureTransformController	
48 NiFloatInterpolator	

Block Details

Name	Type	Value
Next Controller	Ref<NiTimeC...	46
Flags	Flags	72
Frequency	float	1.0000

Reorder Blocks. (Spells -> Sanitize -> Reorder blocks)

You should now have one unnamed NiControllerSequence, containing four Controlled Blocks: One Alpha animation block, one Emissive animation block, one Translate U animation block and one Translate V.



Block List

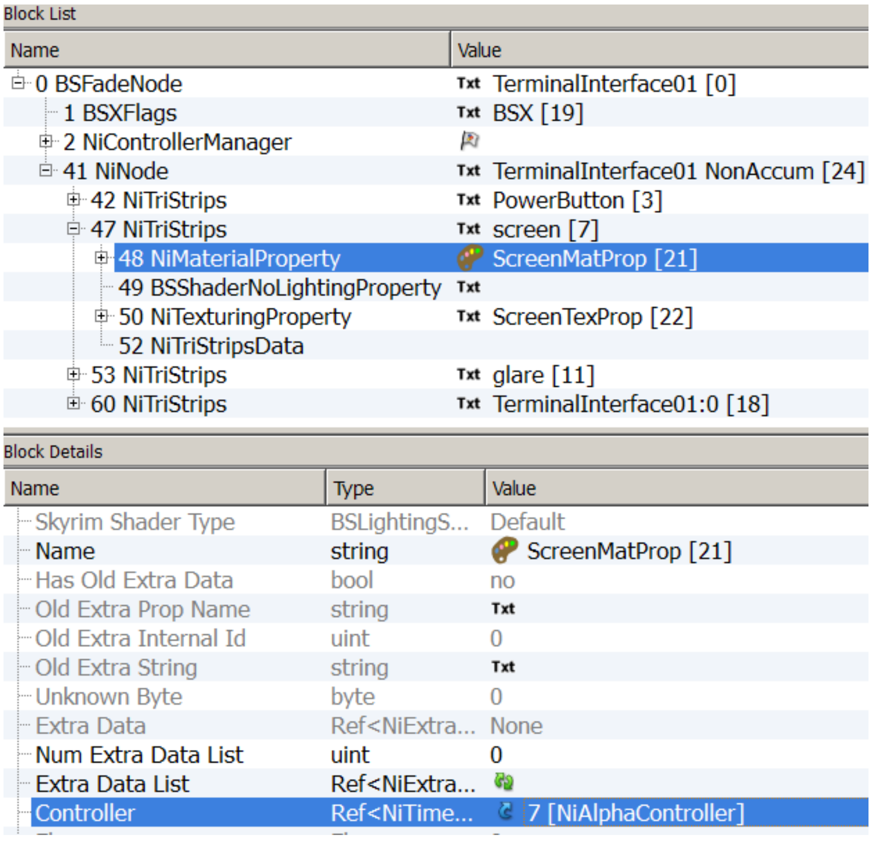
Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
5 NiNode	Txt TerminalInterface01 Non
32 NiControllerSequence	Txt
33 NiFloatInterpolator	
35 NiAlphaController	
37 NiPoint3Interpolator	
39 NiMaterialColorController	
41 NiFloatInterpolator	
43 NiTextureTransformController	
45 NiFloatInterpolator	
47 NiTextureTransformController	
49 NiTextKeyExtraData	
2 NiControllerManager	

Block Details

Name	Type	Value
Name	string	Txt
Text Keys Name	string	Txt
Text Keys	Ref<NiTextKeyExtraD...	None
Unknown Int 4	int	0
Unknown Int 5	int	0
Num Controlled Blocks	uint	4
Unknown Int 1	uint	0
Controlled Blocks	ControllerLink	
Controlled Blocks	ControllerLink	screen [7]
Controlled Blocks	ControllerLink	PowerButton [3]
Controlled Blocks	ControllerLink	screen [7]
Controlled Blocks	ControllerLink	screen [7]

You will need to link the different Material and Texturing Properties to their respective Controllers.

- Go to the NiMaterialProperty of the Screen NiTriStrips, set the NiAlphaController as Controller.



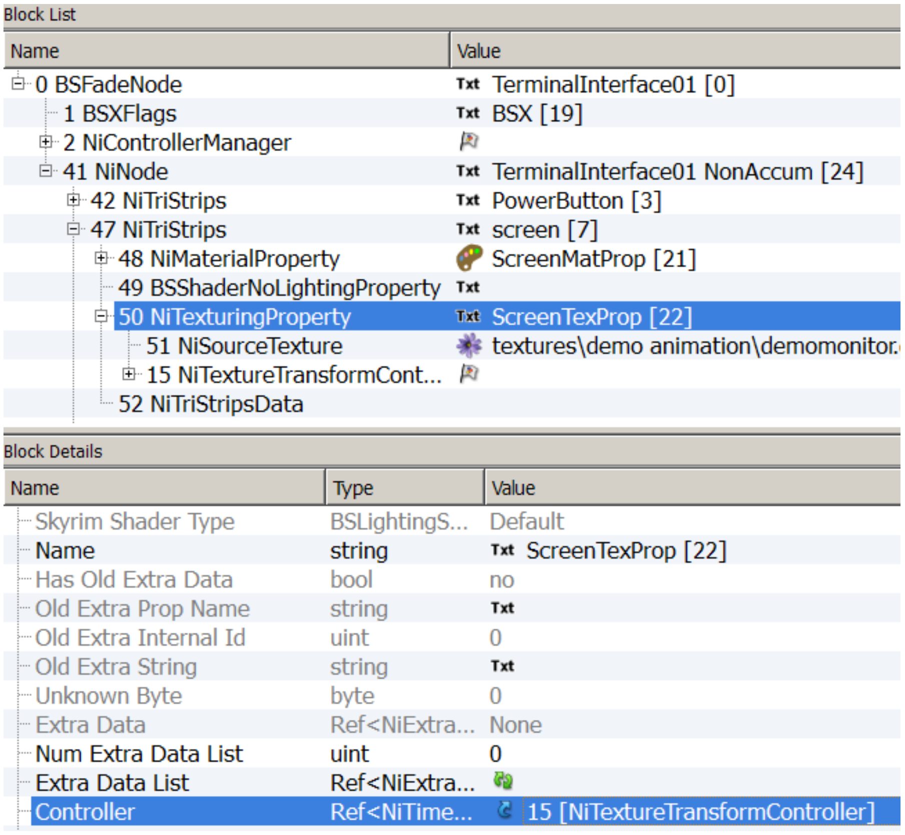
Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
41 NiNode	Txt TerminalInterface01 NonAccum [24]
42 NiTriStrips	Txt PowerButton [3]
47 NiTriStrips	Txt screen [7]
48 NiMaterialProperty	ScreenMatProp [21]
49 BSShaderNoLightingProperty	Txt
50 NiTexturingProperty	Txt ScreenTexProp [22]
52 NiTriStripsData	
53 NiTriStrips	Txt glare [11]
60 NiTriStrips	Txt TerminalInterface01:0 [18]

Block Details

Name	Type	Value
Skyrim Shader Type	BSLightingS...	Default
Name	string	ScreenMatProp [21]
Has Old Extra Data	bool	no
Old Extra Prop Name	string	Txt
Old Extra Internal Id	uint	0
Old Extra String	string	Txt
Unknown Byte	byte	0
Extra Data	Ref<NiExtra...	None
Num Extra Data List	uint	0
Extra Data List	Ref<NiExtra...	
Controller	Ref<NiTime...	7 [NiAlphaController]

- Go to the NiTexturingProperty of the Screen, set the first NiTextureTransformController as Controller.



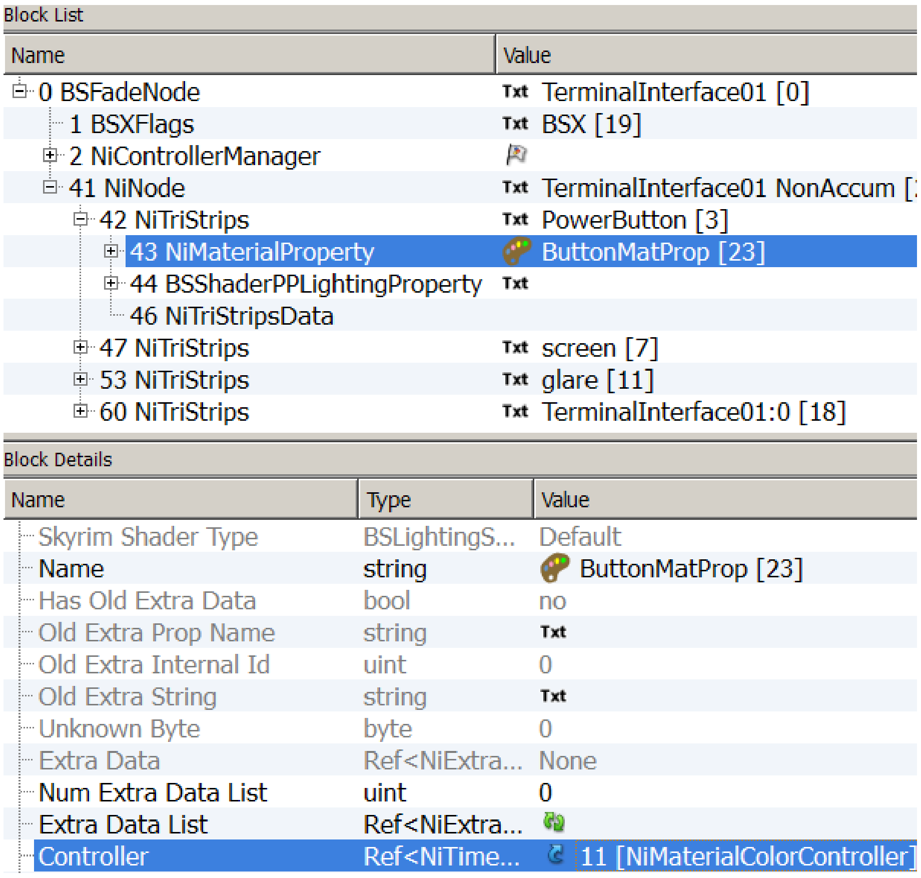
Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
41 NiNode	Txt TerminalInterface01 NonAccum [24]
42 NiTriStrips	Txt PowerButton [3]
47 NiTriStrips	Txt screen [7]
48 NiMaterialProperty	ScreenMatProp [21]
49 BSShaderNoLightingProperty	Txt
50 NiTexturingProperty	Txt ScreenTexProp [22]
51 NiSourceTexture	textures\demo animation\demomonitor.
15 NiTextureTransformCont...	
52 NiTriStripsData	

Block Details

Name	Type	Value
Skyrim Shader Type	BSLightingS...	Default
Name	string	Txt ScreenTexProp [22]
Has Old Extra Data	bool	no
Old Extra Prop Name	string	Txt
Old Extra Internal Id	uint	0
Old Extra String	string	Txt
Unknown Byte	byte	0
Extra Data	Ref<NiExtra...	None
Num Extra Data List	uint	0
Extra Data List	Ref<NiExtra...	
Controller	Ref<NiTime...	15 [NiTextureTransformController]

- Go to the NiMaterialProperty of the PowerButton, set the NiMaterialColorController as Controller.


Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
41 NiNode	Txt TerminalInterface01 NonAccum [
42 NiTriStrips	Txt PowerButton [3]
43 NiMaterialProperty	ButtonMatProp [23]
44 BSShaderPPLightingProperty	Txt
46 NiTriStripsData	
47 NiTriStrips	Txt screen [7]
53 NiTriStrips	Txt glare [11]
60 NiTriStrips	Txt TerminalInterface01:0 [18]

Block Details

Name	Type	Value
Skyrim Shader Type	BSLightingS...	Default
Name	string	ButtonMatProp [23]
Has Old Extra Data	bool	no
Old Extra Prop Name	string	Txt
Old Extra Internal Id	uint	0
Old Extra String	string	Txt
Unknown Byte	byte	0
Extra Data	Ref<NiExtra...	None
Num Extra Data List	uint	0
Extra Data List	Ref<NiExtra...	
Controller	Ref<NiTime...	11 [NiMaterialColorController]

If you plan to use more animation blocks that will take place in both sequences, you can add it now, following the same steps.

If you are confident enough to have all the animations needed and all the settings correctly set you can duplicate the sequence.

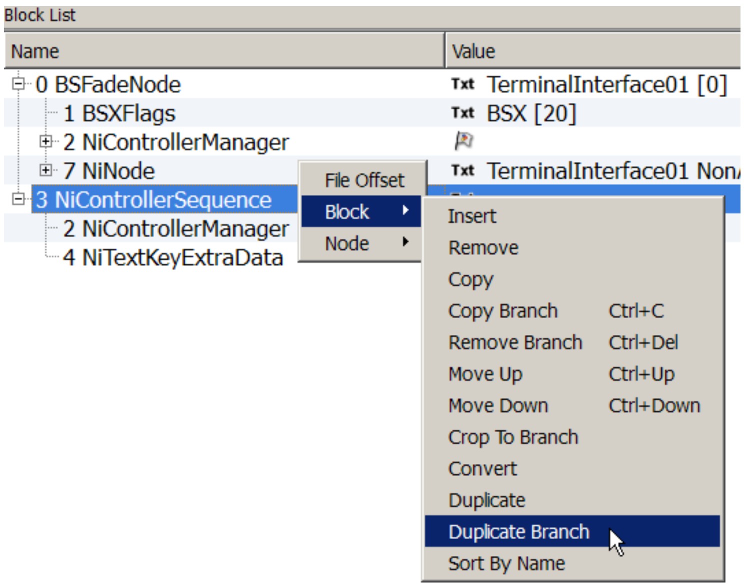
As said before, the names given to the sequences do matter; they are related to script triggering.

Those of interest for us now are: **Forward** and **Backward.**

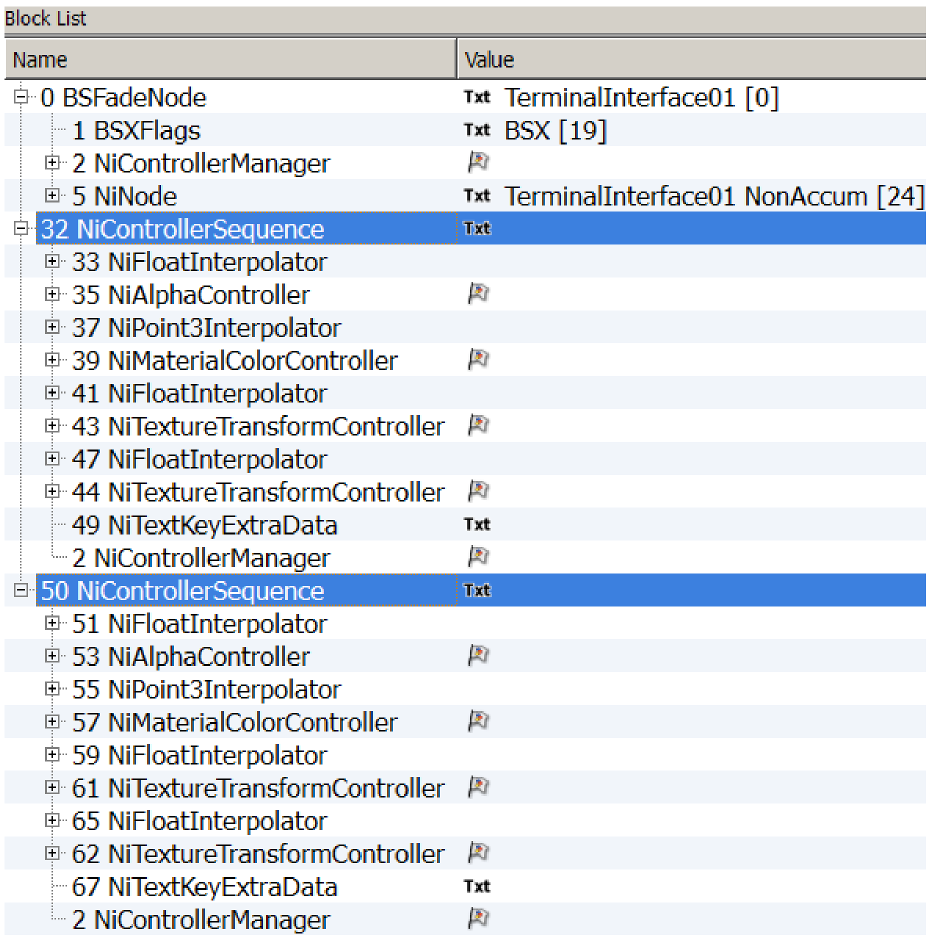
DUPLICATING SEQUENCE

Since there is more than one element, we will use Duplicate Branch.

1. Select the NiControllerSequence then, Block ➪ Duplicate Branch.



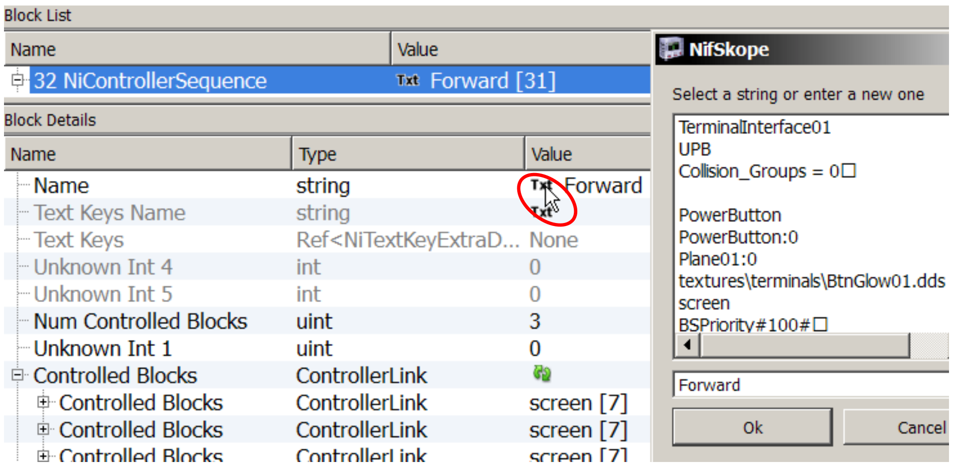
This should be the result at this stage.



Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
5 NiNode	Txt TerminalInterface01 NonAccum [24]
32 NiControllerSequence	Txt
33 NiFloatInterpolator	
35 NiAlphaController	
37 NiPoint3Interpolator	
39 NiMaterialColorController	
41 NiFloatInterpolator	
43 NiTextureTransformController	
47 NiFloatInterpolator	
44 NiTextureTransformController	
49 NiTextKeyExtraData	Txt
2 NiControllerManager	
50 NiControllerSequence	Txt
51 NiFloatInterpolator	
53 NiAlphaController	
55 NiPoint3Interpolator	
57 NiMaterialColorController	
59 NiFloatInterpolator	
61 NiTextureTransformController	
65 NiFloatInterpolator	
62 NiTextureTransformController	
67 NiTextKeyExtraData	Txt
2 NiControllerManager	

2. Defining the two different NiControllerSequence.

- Select the first NiControllerSequence.
- Enter the String-Name: Forward (Forward will be played by default, in games).



Block List

Name	Value
32 NiControllerSequence	Txt Forward [31]

Block Details

Name	Type	Value
Name	string	Txt Forward
Text Keys Name	string	Txt
Text Keys	Ref<NiTextKeyExtraD...	None
Unknown Int 4	int	0
Unknown Int 5	int	0
Num Controlled Blocks	uint	3
Unknown Int 1	uint	0
Controlled Blocks	ControllerLink	
Controlled Blocks	ControllerLink	screen [7]
Controlled Blocks	ControllerLink	screen [7]
Controlled Blocks	ControllerLink	screen [7]

NifSkope

Select a string or enter a new one

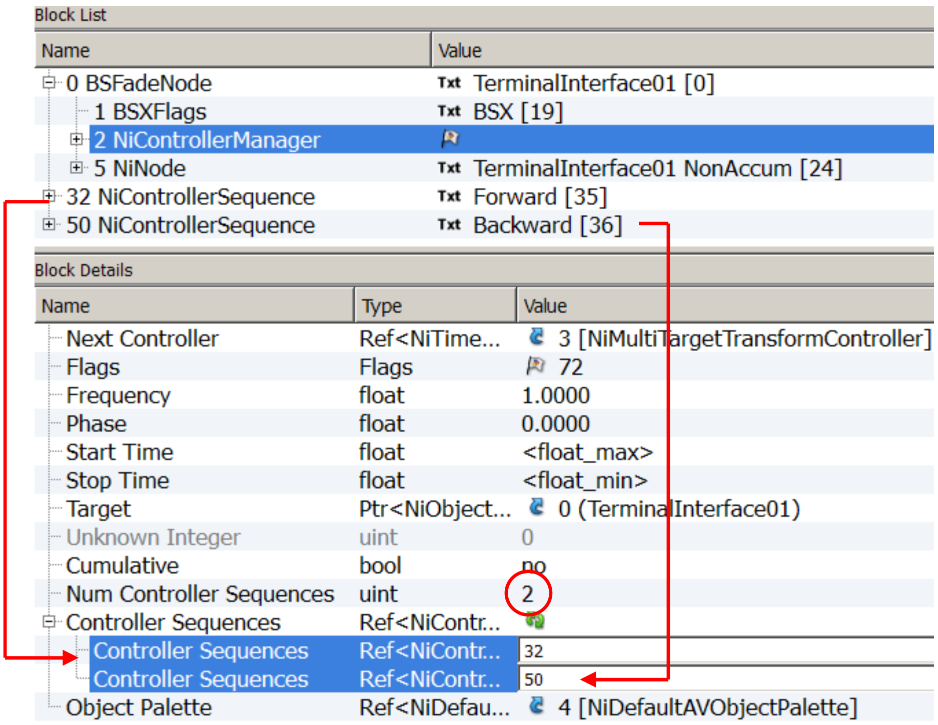
```
TerminalInterface01
UPB
Collision_Groups = 0□

PowerButton
PowerButton:0
Plane01:0
textures\terminals\BtnGlow01.dds
screen
BSPriority#100#□
```

Forward

- Select the second NiControllerSequence.
- Name it Backward

3. Attaching the two sequences to the NiControllerManager.

- Select the NiControllerManager.
- Set the number of sequences in the Num Controller Sequences (2). Update.
- Link the two sequences.



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
5 NiNode	Txt TerminalInterface01 NonAccum [24]
32 NiControllerSequence	Txt Forward [35]
50 NiControllerSequence	Txt Backward [36]

Block Details

Name	Type	Value
Next Controller	Ref<NiTime...	3 [NiMultiTargetTransformController]
Flags	Flags	72
Frequency	float	1.0000
Phase	float	0.0000
Start Time	float	<float_max>
Stop Time	float	<float_min>
Target	Ptr<NiObject...	0 (TerminalInterface01)
Unknown Integer	uint	0
Cumulative	bool	no
Num Controller Sequences	uint	2
Controller Sequences	Ref<NiContr...	
Controller Sequences	Ref<NiContr...	32
Controller Sequences	Ref<NiContr...	50
Object Palette	Ref<NiDefau...	4 [NiDefaultAVObjectPalette]

Save.

At this point you can check in Geck preview if your structure is stable.

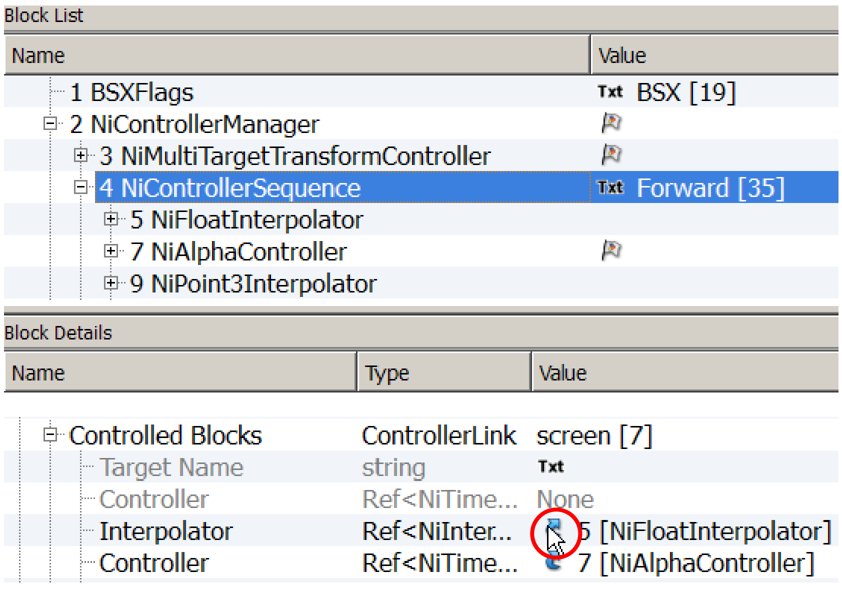
It might be a good starting point for variations, as now, the remaining work will only be setting times and events.

TIME AND KEYS SETTINGS

Let's start with the Alpha animation of the first sequence, the Forward one.

1. Finding of the NiFloatData corresponding to the NiAlphaController.

- Select the first of the Controlled blocks (the Alpha one) and expand it.
- Click on the blue arrow of the Interpolator.



Block List

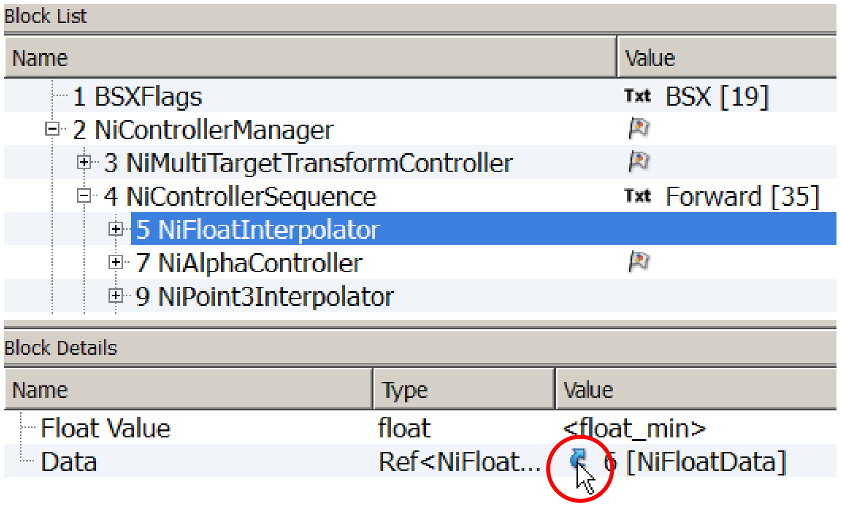
Name	Value
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
3 NiMultiTargetTransformController	
4 NiControllerSequence	Txt Forward [35]
5 NiFloatInterpolator	
7 NiAlphaController	
9 NiPoint3Interpolator	

Block Details

Name	Type	Value
Controlled Blocks	ControllerLink	screen [7]
Target Name	string	Txt
Controller	Ref<NiTime...	None
Interpolator	Ref<NiInter...	5 [NiFloatInterpolator]
Controller	Ref<NiTime...	7 [NiAlphaController]

You will be transported to it.

- Same there, click on the blue arrow to go to the Data.



Block List

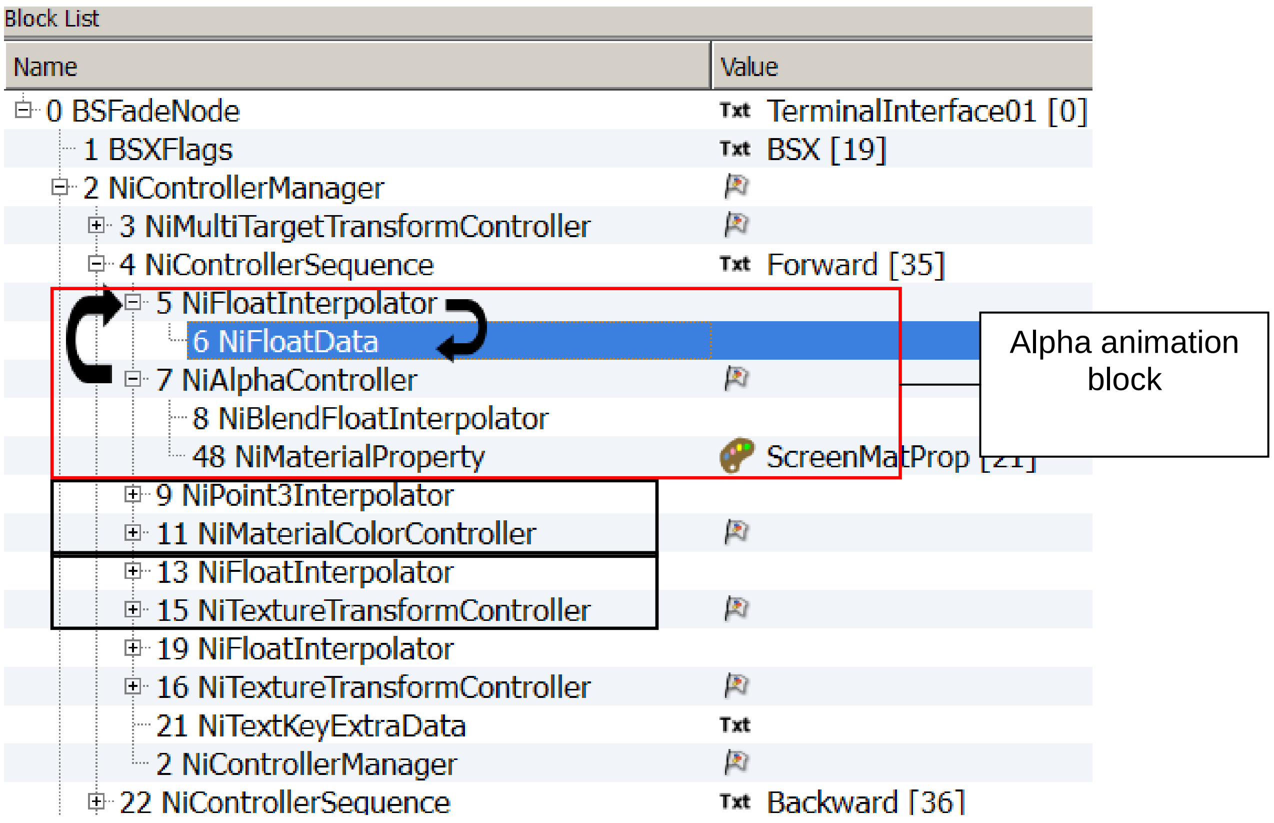
Name	Value
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
3 NiMultiTargetTransformController	
4 NiControllerSequence	Txt Forward [35]
5 NiFloatInterpolator	
7 NiAlphaController	
9 NiPoint3Interpolator	

Block Details

Name	Type	Value
Float Value	float	<float_min>
Data	Ref<NiFloat...	6 [NiFloatData]

Again, this is the safest way to navigate in your structure.

You can also find the Data by looking at the structure and reading its pattern.
Above each Controller is the Interpolator related to it in the Sequence.
Linked to it is the Data with the keys.



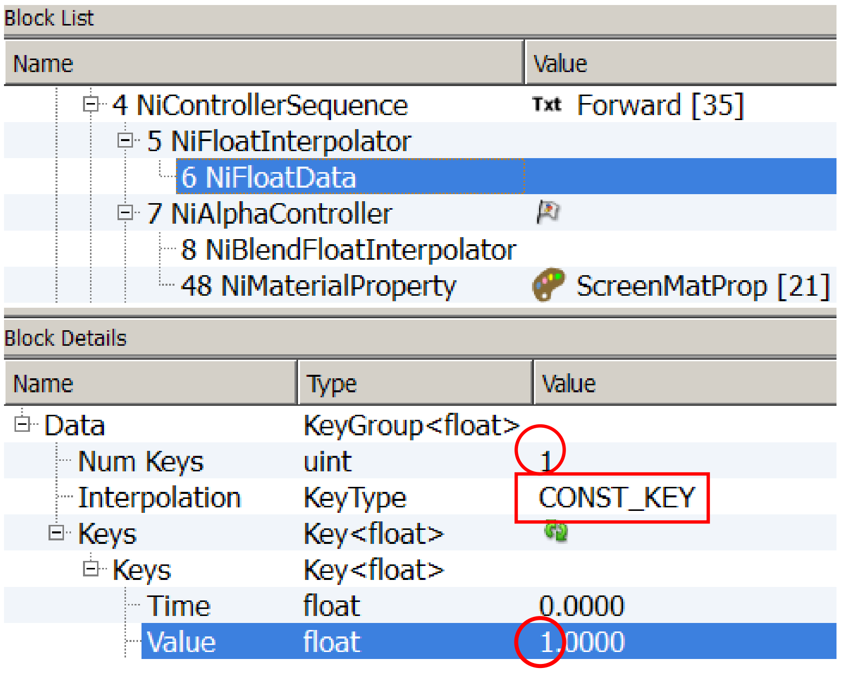
Block List

```
Name                                        Value
0 BSFadeNode                                Txt TerminalInterface01 [0]
  1 BSXFlags                                Txt BSX [19]
  2 NiControllerManager
    3 NiMultiTargetTransformController
    4 NiControllerSequence                  Txt Forward [35]
      5 NiFloatInterpolator
        6 NiFloatData
      7 NiAlphaController
        8 NiBlendFloatInterpolator
        48 NiMaterialProperty                   ScreenMatProp [21]
      9 NiPoint3Interpolator
      11 NiMaterialColorController
      13 NiFloatInterpolator
      15 NiTextureTransformController
      19 NiFloatInterpolator
      16 NiTextureTransformController
      21 NiTextKeyExtraData                 Txt
      2 NiControllerManager
    22 NiControllerSequence                 Txt Backward [36]
```

Any ways, you've ended up in the NiFloatData of the Alpha Controller.

2. Setting the keys of the Alpha Controller - Sequence Forward.

- Choose the number of keys = 1
- Set the interpolation to CONST_KEY
- Set the Key Value to 1



Block List

Name	Value
4 NiControllerSequence	Txt Forward [35]
5 NiFloatInterpolator	
6 NiFloatData	
7 NiAlphaController	
8 NiBlendFloatInterpolator	
48 NiMaterialProperty	ScreenMatProp [21]

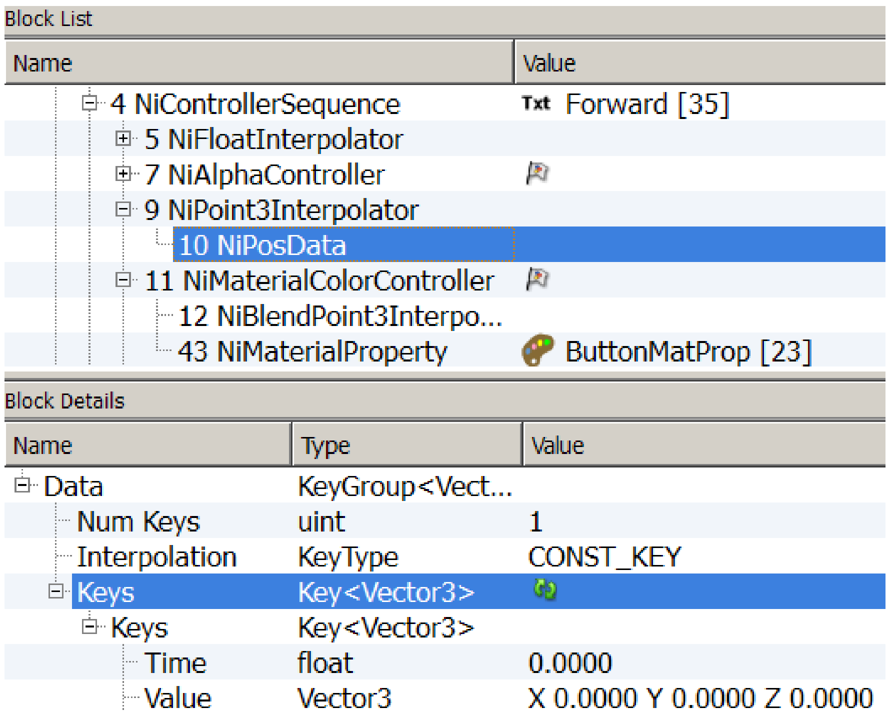
Block Details

Name	Type	Value
Data	KeyGroup<float>	
Num Keys	uint	1
Interpolation	KeyType	CONST_KEY
Keys	Key<float>	
Keys	Key<float>	
Time	float	0.0000
Value	float	1.0000

3. Setting the keys of the Material Controller - Sequence Forward.

Locate the NiPosData related to the Material Controller.

- Number of key = 1
- Set the interpolation to CONST_KEY
- Leave the Time and Value to 0 as it will be the unlit state of the button.



Block List

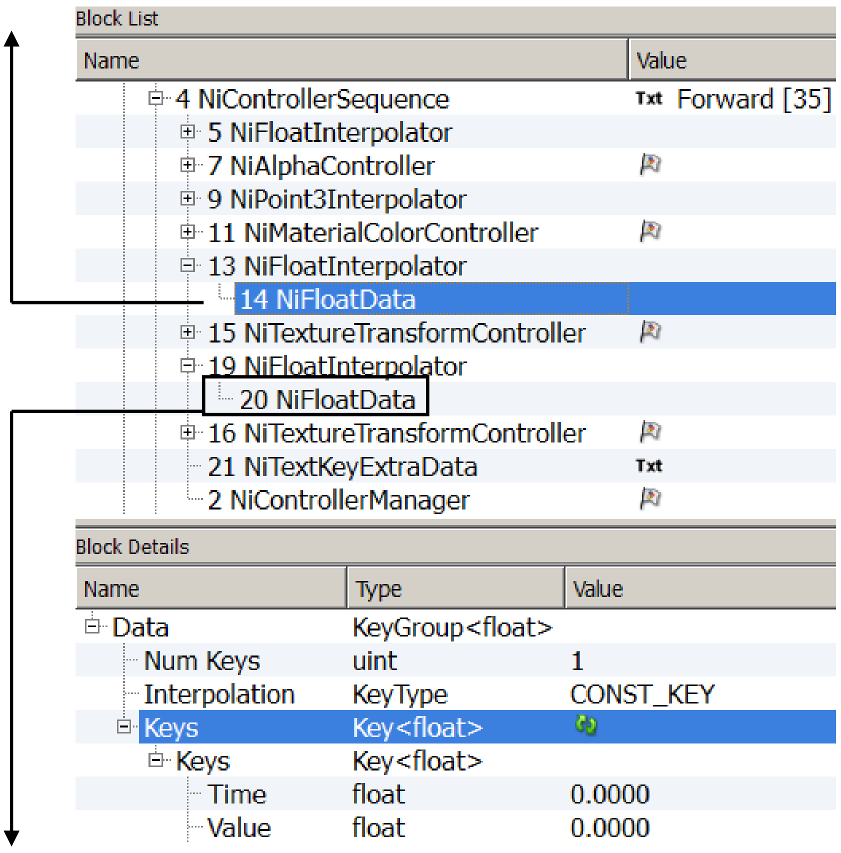
Name	Value
4 NiControllerSequence	Txt Forward [35]
5 NiFloatInterpolator	
7 NiAlphaController	
9 NiPoint3Interpolator	
10 NiPosData	
11 NiMaterialColorController	
12 NiBlendPoint3Interpo...	
43 NiMaterialProperty	ButtonMatProp [23]

Block Details

Name	Type	Value
Data	KeyGroup<Vect...	
Num Keys	uint	1
Interpolation	KeyType	CONST_KEY
Keys	Key<Vector3>	
Keys	Key<Vector3>	
Time	float	0.0000
Value	Vector3	X 0.0000 Y 0.0000 Z 0.0000

4. Setting the keys of the NiTextureTransformController U - Sequence Forward.

- Locate the NiFloatData related to the NiTextureTransformController U.
- Choose the number of keys = 1
- Set the interpolation to CONST_KEY
- The Time and Value will be left at 0



Block List

Name	Value
4 NiControllerSequence	Txt Forward [35]
5 NiFloatInterpolator	
7 NiAlphaController	
9 NiPoint3Interpolator	
11 NiMaterialColorController	
13 NiFloatInterpolator	
14 NiFloatData	
15 NiTextureTransformController	
19 NiFloatInterpolator	
20 NiFloatData	
16 NiTextureTransformController	
21 NiTextKeyExtraData	Txt
2 NiControllerManager	

Block Details

Name	Type	Value
Data	KeyGroup<float>	
Num Keys	uint	1
Interpolation	KeyType	CONST_KEY
Keys	Key<float>	
Keys	Key<float>	
Time	float	0.0000
Value	float	0.0000

5. Setting the keys of the NiTextureTransformController V - Sequence Forward.

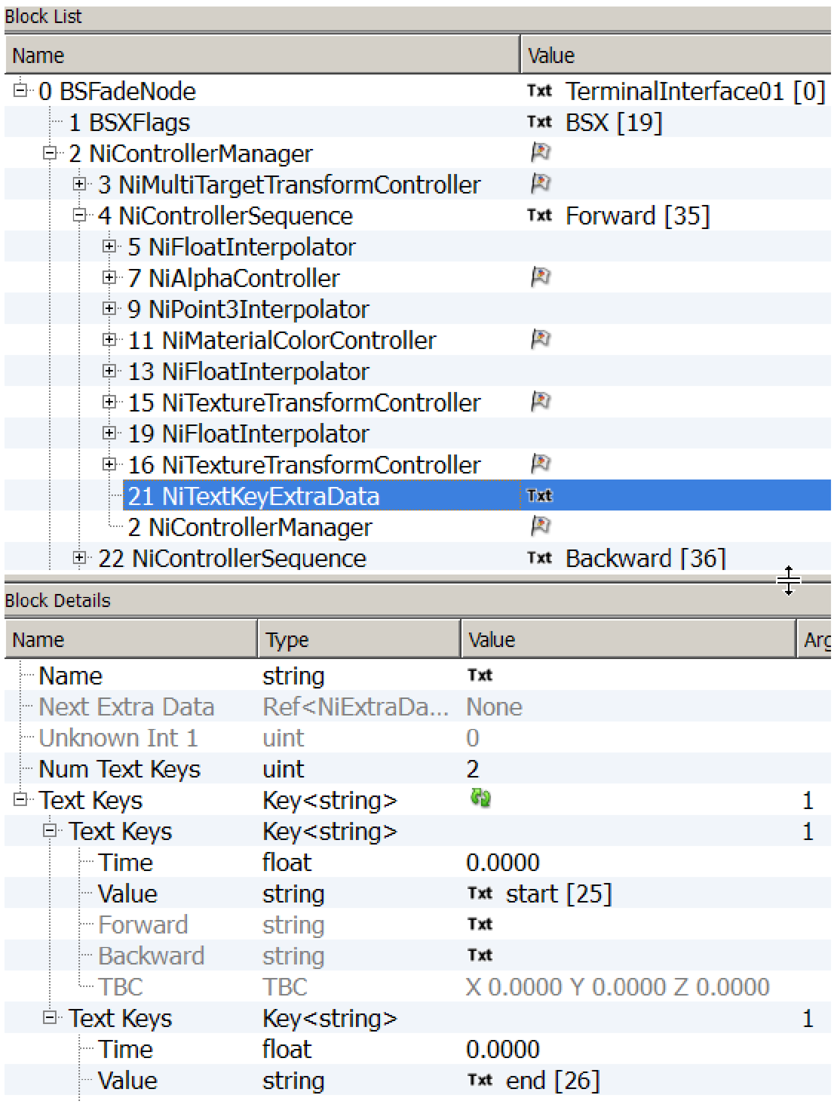
- Locate the NiFloatData related to the NiTextureTransformController V.
- Choose the number of keys = 1
- Set the interpolation to CONST_KEY
- The Time and Value will be left at 0

Last element to be adjusted for the forward sequence: the NiTextKeyExtraData.

Actually, in this case the NiTextKeyExtraData is all set up from our previous steps *(point 4 of Sequence Building – page 17).*

6. Setting of the NiTextKeyExtraData - Sequence Forward.

- Select the NiTextKeyExtraData. (Should be found at the end of the sequence.)
- Check the settings.



Block List

Name	Value
0 BSFadeNode	Txt TerminalInterface01 [0]
1 BSXFlags	Txt BSX [19]
2 NiControllerManager	
3 NiMultiTargetTransformController	
4 NiControllerSequence	Txt Forward [35]
5 NiFloatInterpolator	
7 NiAlphaController	
9 NiPoint3Interpolator	
11 NiMaterialColorController	
13 NiFloatInterpolator	
15 NiTextureTransformController	
19 NiFloatInterpolator	
16 NiTextureTransformController	
21 NiTextKeyExtraData	Txt
2 NiControllerManager	
22 NiControllerSequence	Txt Backward [36]

Block Details

Name	Type	Value	Arg
Name	string	Txt	
Next Extra Data	Ref<NiExtraDa...	None	
Unknown Int 1	uint	0	
Num Text Keys	uint	2	
Text Keys	Key<string>		1
Text Keys	Key<string>		1
Time	float	0.0000	
Value	string	Txt start [25]	
Forward	string	Txt	
Backward	string	Txt	
TBC	TBC	X 0.0000 Y 0.0000 Z 0.0000	
Text Keys	Key<string>		1
Time	float	0.0000	
Value	string	Txt end [26]	

You have now one functional Forward-OFF sequence with four animation blocks.

The Alpha block will maintain visibility of the Screen.

The Material Color block will set the Button unlit.

The Translate U block
The Translate V block } will place and keep the Screen texture up left .

The steps to set the Backward sequence will be the same, only the parameters values will differ.

1. Select the NiControllerSequence Backward.
   - Set the stop time to 24 (24 seconds is the time a camera take to make its panning animation).

2. Setting the keys of the Alpha Controller - Sequence Backward.
   - Locate the NiFloatData related to the NiAlphaController.
   - Choose the number of key (1)
   - Set the interpolation to CONST_KEY
   - Set the Key Value to 1

3. Setting the keys of the NiMaterialColorController.
   - Locate the NiPosData related to the NiMaterialColorController.
   - Choose the number of key (1)
   - Set the interpolation to CONST_KEY
   - Set the key Values to X 0.9000 Y 0.9000 Z 0.9000

4. Setting the keys of the NiTextureTransformController U - Sequence Backward.
   - Locate the NiFloatData related to the NiTextureTransformController U.
   - Choose the number of key (10)
   - Set the interpolation to LINEAR_KEY
   - Set the Time and Value as following:

Key	**Time**	**0.0000**
	Value	0.5000
Key	**Time**	**1.0000**
	Value	0.5000
Key	**Time**	**5.0000**
	Value	0.0000
Key	**Time**	**7.2000**
	Value	0.0000
Key	**Time**	**11.0000**
	Value	0.5000
Key	**Time**	**12.2000**
	Value	0.5000
Key	**Time**	**16.0000**
	Value	0.0000
Key	**Time**	**18.2000**
	Value	0.0000
Key	**Time**	**22.0000**
	Value	0.5000
Key	**Time**	**23.0000**
	Value	0.5000

5. Setting the keys of the NiTextureTransformController V - Sequence Backward.

- Locate the NiFloatData related to the NiTextureTransformController V.
- Choose the number of key (1)
- Set the interpolation to LINEAR_KEY
- Set the Value to 0.5000

6. Setting of the NiTextKeyExtraData - Sequence Backward.

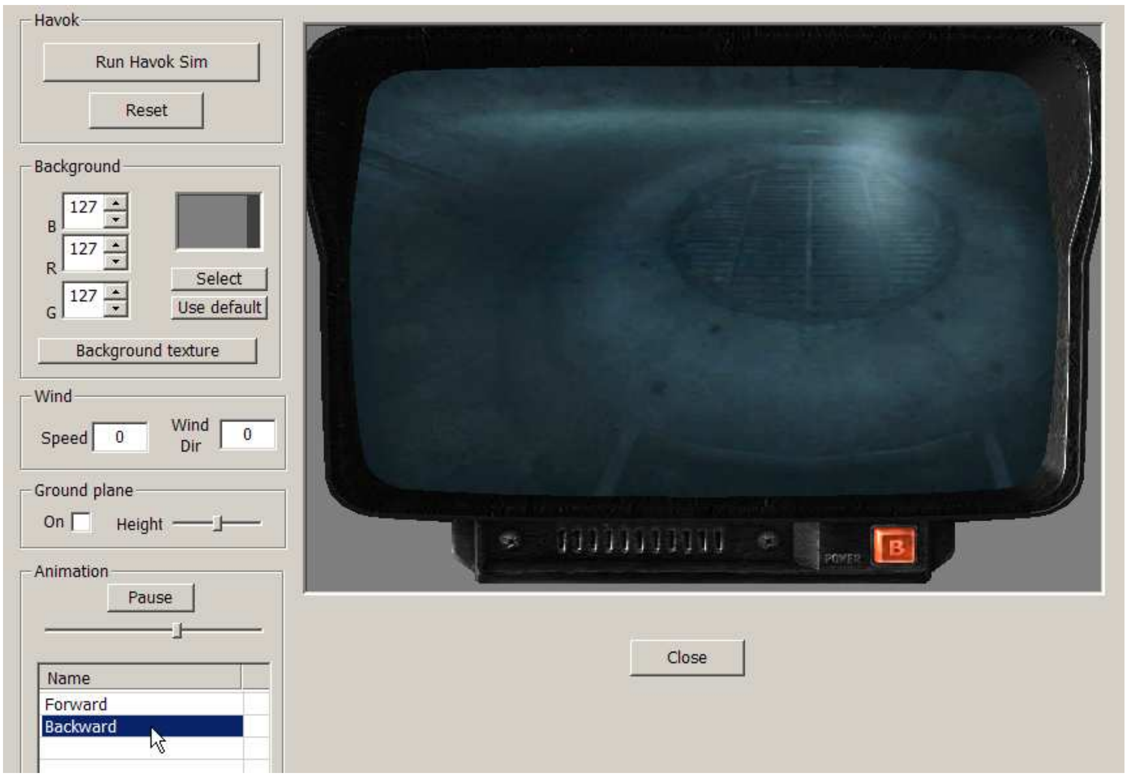
- Select the NiTextKeyExtraData. (should be found at the end of the sequence)
- Enter 24.0000 in the "end" Time Value.

Save.

Congratulations!

You have now a fully functional nif with two sequences triggering four animation blocks each. The NifSkope preview doesn't give justice to your hard work, open it in Geck preview.

Select your sequence



It’s a nice base, but it could certainly be improved!

For example, the Glare is a nice touch when the monitor is OFF, but might be distracting when ON.

Since the present Alpha Block Animation is not really active, you could make use of it to set the Glare invisible at ON state and visible at OFF state.

The procedure would be for both Sequences:

- Select the Alpha Controlled Block, change the Node Name to Glare
- Adjust the Keys in the NiFloatData (1 in Forward Sequence, 0 in Backward)

Then, in the Material Property of the Glare, Select the first Alpha Controller as Controller and delete it in the Material Property of the screen.

Or you may want to add some images jumps with the UV Translate.

You may want to animate the Glare with scan lines.

These examples can be seen in the DemoCameraMonitor.nif.

It’s time for you to test and play!

SCRIPT

If you make an activator based on the monitor nif you've just built and attach a script to it, you would be making the Player able to trigger the ON/OFF states in game.

The PlayGroup command plays an animation group on the calling actor.
The AnimGrpoups have specific names.

See AnimGroups for a list of possible animations.
http://geck.bethsoft.com/index.php?title=AnimGroups

A flag must be used to control initialization of the group.

0 = Normal - The current animation will finish its full cycle, and the new animation will start from its beginning.

1 = Immediate Start - The current animation will stop regardless of the frame it is on, and the new animation will start from its beginning.

2 = Immediate Loop -The current animation will stop regardless of the frame it is on, and the new animation will start at the beginning of its loop cycle.

Here is a short script to switch ON & OFF the monitor (or any other nif with two sequences).

```
Scn MyMonitorPlayScript


Short MonitorOn        ; 0 = off, 1= on

Begin OnActivate
        Activate
        if MonitorOn == 0
            Playgroup Backward 0
            Set MonitorOn to 1
        else
            Playgroup Forward 0
            Set MonitorOn to 0
        endif
End
```

This is of course the basics; it can be extended and adapted to your need.

TROUBLESHOOTING CHECK LIST

Error message when trying to copy branch.	Are the **NiMaterialProperty** and **NiTexturingProperty** properly named?
Rotating texture is not centered.	In the NiTexturingProperty, adjust **Center Offset** setting of the Base Texture section. *(Part 3, page 11)*
Animated texture is acting weird. (reversed, random, ...)	Check **Transform Type?** in the Base Texture section of the NiTexturingProperty. *(Part 3, page 7)*
The Time setting doesn't work.	With a Controller Manager, Time is set in three places: in the **NiControllerSequence,** the **NiTextKeyExtraData,** (these need to match), and in the **Texture or Material Controllers** itself (those can be shorter or equal to the general length set above).
The object can't be activated in game.	Is there a **Collision Object** in the mesh? **BSXFlag** Enable Collision checked?
Texture is not visible.	With a Controller Manager, texture need to be referenced twice: in the **NiSourceTexture** and in the **BSShaderNoLightingProperty**. *(Part 3 Page 5)*
Everything is set, but animation won't start.	Is the **Has Texture Transform** set to Yes? (In NiTexturingProperty's Base Texture settings) *(Part 3 Page 6)*
Everything is set, but animation won't start.	Does the NiMaterialProperty or the NiTexturingProperty have their respective NiMaterialColorController or NiTextureTransformController referenced? (*Part 2 Page 7 Point 8, Page 12 Point 9*. *- Part 3 Page 9 Point 8, Page 13 Point 8. - Part 4, Page 34)*

FINAL NOTE

I hope that this tutorial has given you a rough idea of what can be done with Material and Texture Animation and the possibility to do it yourself.

The potential is endless and the applicable fields are barely touched here.

Think of Special Fx, weapons, armors, etc.

I have tried my best to bring to light the structures and patterns as I have understood them.

My biggest reward would be that the tutorial becomes rapidly useless to you. Because you don’t need to follow a step by step instruction any more, you actually know what you are doing and why you are doing it.

And in case you are lost, go open some Vanilla meshes and inspects them in detail, the answer is there somewhere…

Some meshes of interest:

Base Game

Computerrack01.nif – in Clutter\consoles

Console02.nif – in Clutter\consoles

projectorlightfx01.nif – in Clutter\projector

radiowave.nif – in interface\radiowave

shishkebab.nif – in weapons\1handmeele

fxtesla.nif – in armor\powerarmor

DLC – Point Lookout

Dlc04tvmansionsecurity01.nif – in dlc04\Clutter

And of course, a masterpiece: Television Comes To Fallout by RazoWire.

http://www.nexusmods.com/fallout3/mods/14675/?

You have tools, now your only limitation is your imagination … and your time. Good luck!

If this has brought you some satisfaction, don’t hesitate to express it; a comment, an endorsement, a credit or Kudos gives a warm fuzzy feeling and encourages me to continue … ☺
Thank you.
Pixelhate. June 2014.

CREDITS (ALPHABETIC)

I would like to thank deeply and sincerely.

- **Anoxeron** for his dedication, enthusiasm and benevolence. Thank you for testing each exercise, each step without respite and for giving me countless opportunities to improve this tutorial. Thank you for proof reading and corrections.
  http://www.nexusmods.com/fallout3/users/3694499/?
- **BrettM** for his tutorial about Textures Animations for Skyrim, despite a big difference in the engine, his tutorial was an inspiration for layout and presentation.
  http://www.nexusmods.com/skyrim/mods/47104/?
- **Ghogiel** for his guide about NiMaterialControlers, his resources and mods which are real learning material and for his kind private answers about Specularity.
  http://wiki.tesnexus.com/index.php/Adding_material_controllers_to_objects_in_Nifskope
  http://www.nexusmods.com/fallout3/users/209926/?tb=mods&pUp=1
- Nexus for hosting this.
  http://www.nexusmods.com/fallout3/
- **Prensa** for her kindness, constant support and patient explanations. For sending me back faulty experiments, miraculously repaired and working. Her posts, on Nexus, can be compiled and will answer most of the troubles you'll encounter in FO3. Thank you for proof reading and corrections.
  http://www.nexusmods.com/fallout3/users/751212/?tb=mods&pUp=1
  http://forums.nexusmods.com/index.php?/user/751212-prensa/
- **Sullyvanj93** for partial proof reading and writing advices.
- **TrickyVein** for his very clear tutorial on NiControllerManager on Nexus which has helped me to tame "this intimidating beast" and understand the necessity to develop a workflow.
  http://forums.nexusmods.com/index.php?/topic/984792-tutorial-working-with-the-nicontrollermanager/
- **Turboscalpeur** for his friendship, support and beautiful images.
  http://forums.nexusmods.com/index.php?/user/4708873-turboscalpeur/
  https://www.flickr.com/photos/turboscalpeur/
- **Weijiesen** for overall theory consistency scrutinizations and awesomeness!
  http://www.nexusmods.com/newvegas/users/1026866/?tb=images&pUp=1
- The good people who takes time to write articles, tutorials, guides and make them available to others.
  https://www.google.com
- The nice people helping on forums like Nexus, NifTools Forum, etc.
  http://forums.nexusmods.com/
  http://niftools.sourceforge.net/forum/
- The makers of The Elder Scrolls NifSkope guide for their very detailed guide offering the bases on what this tutorial is build
  http://cs.elderscrolls.com/index.php?title=Working_With_Nifs_101_:_An_Introduction
  http://cs.elderscrolls.com/index.php?title=Working_With_Nifs_101_:_Basic_Use
- The makers of the tools that allow us to express our creativity and share it.
  NifSkope http://sourceforge.net/projects/niftools/files/nifskope/1.1.3/
  Geck http://geck.bethsoft.com/index.php?title=Main_Page
  Geck PowerUp http://www.nexusmods.com/fallout3/mods/15067/?