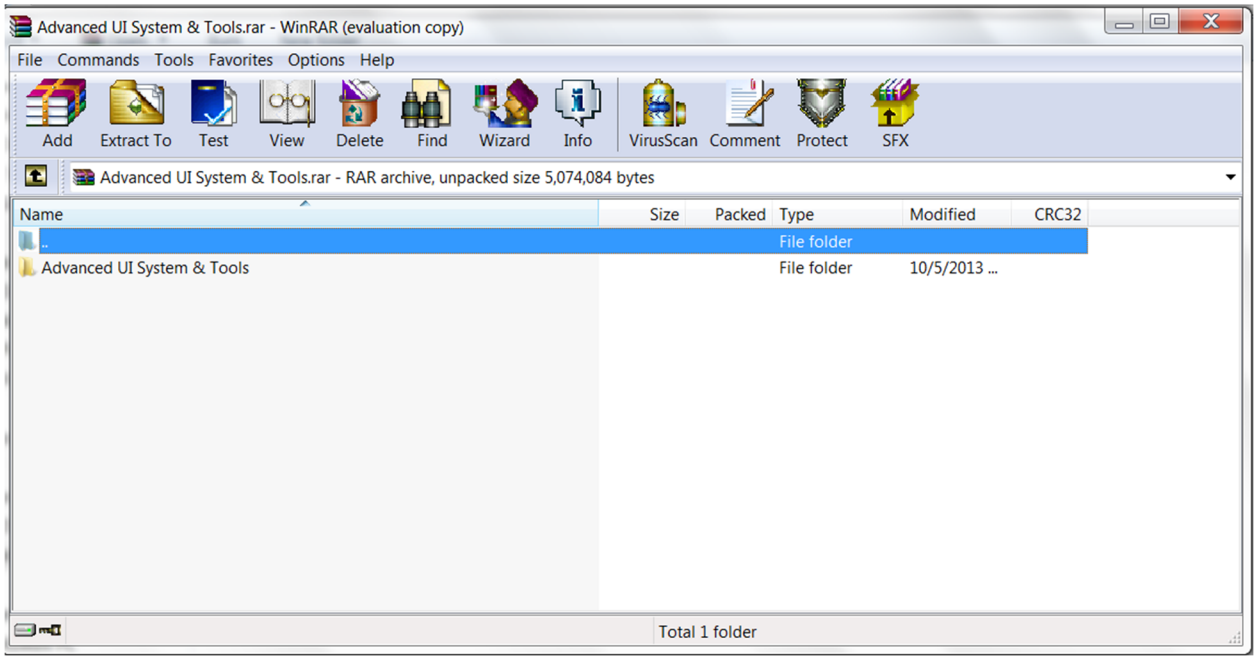
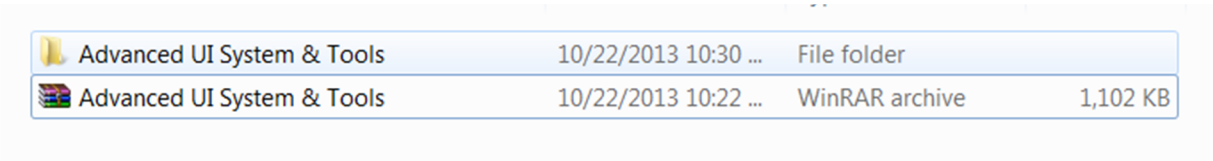
Installation Guide

Follow these steps for installation.

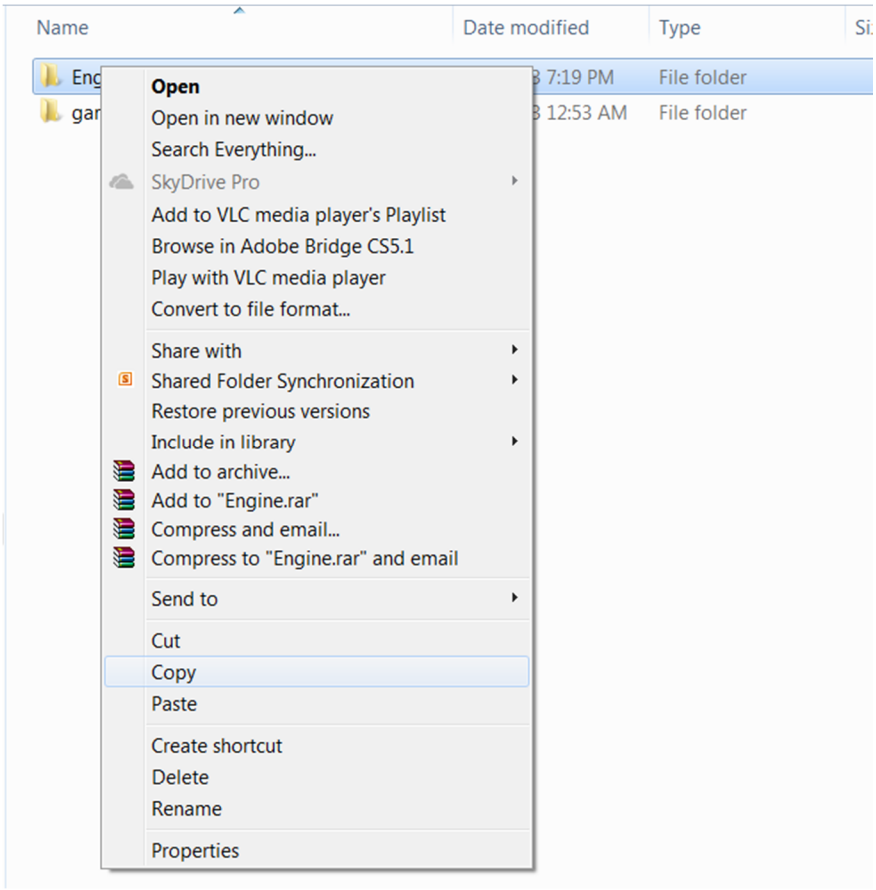
1. Run the installer and extract the contents wherever you want. Then, open up Advanced UI System & Tools.rar



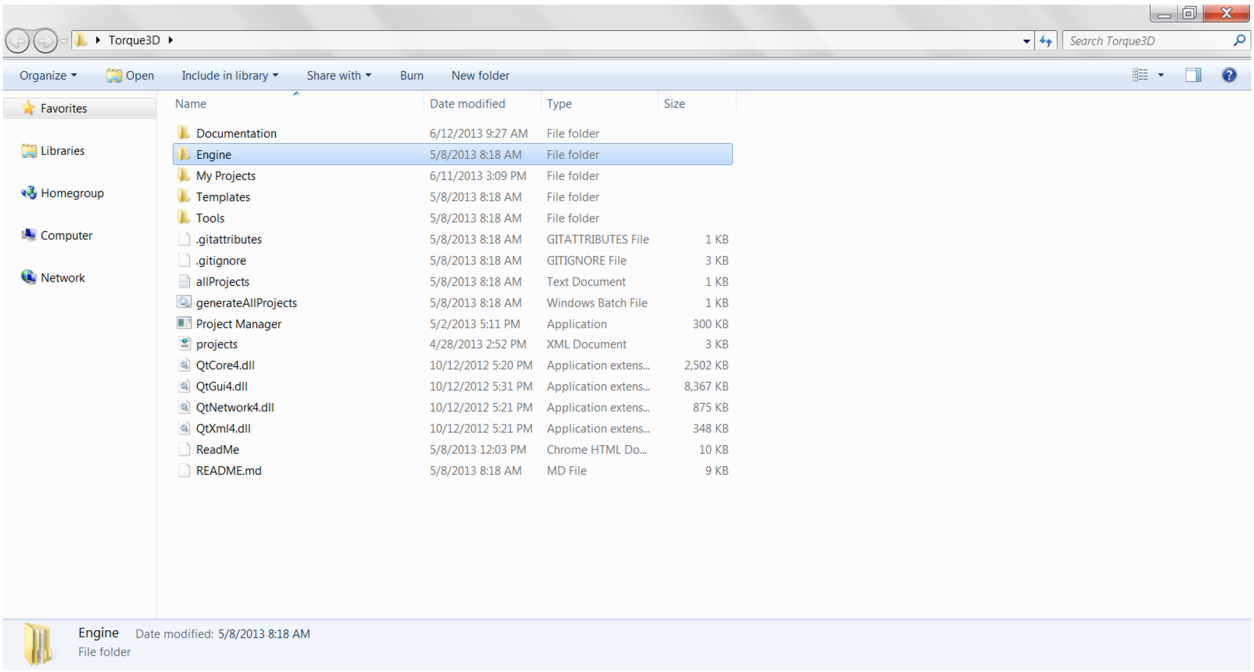
2. After extraction.



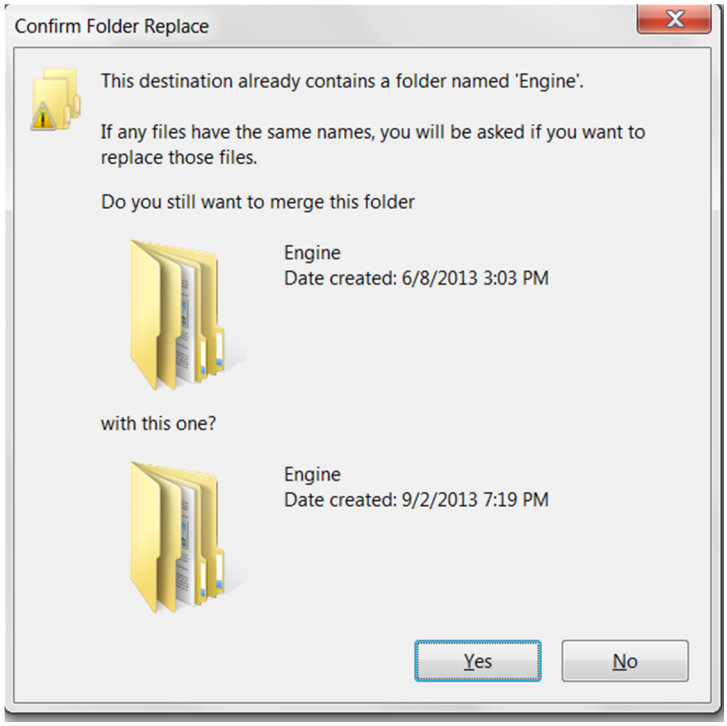
3. Copy the **Engine** folder.

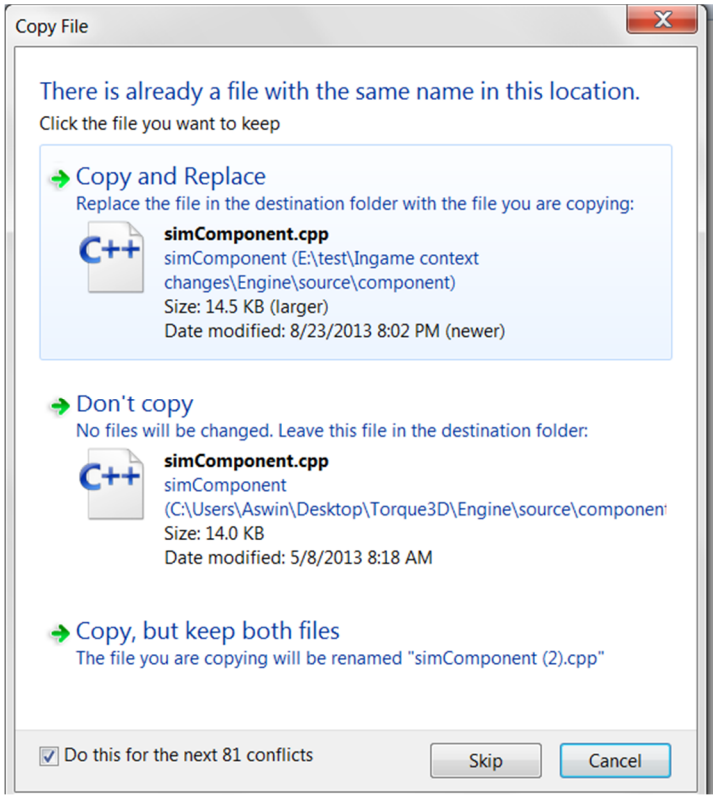


4. Go to your Torque3D folder where the engine is located.

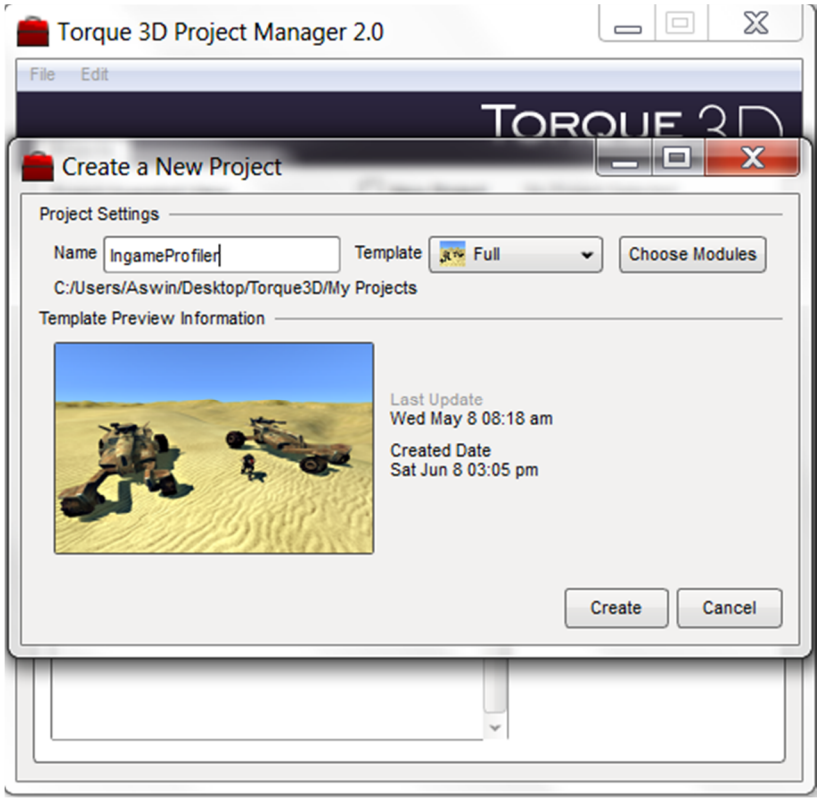


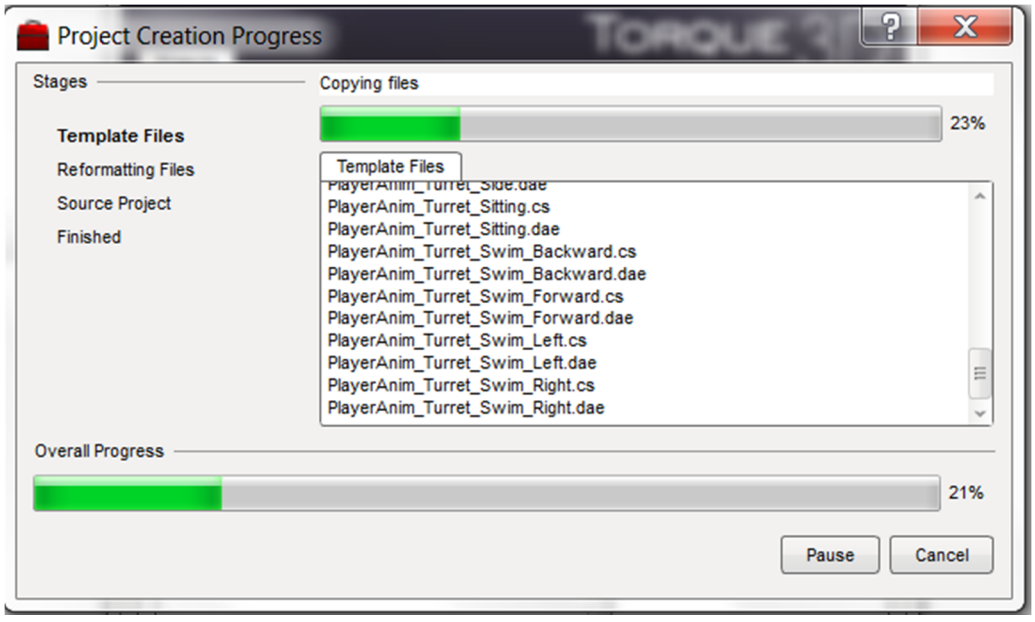
5. Copy the Engine folder here and copy/replace the existing ones.

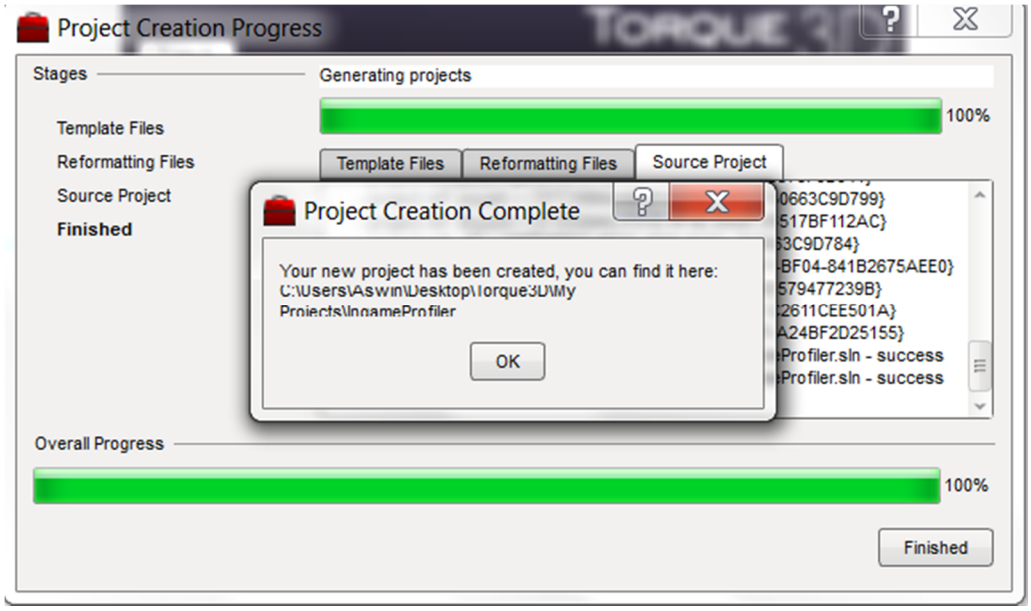




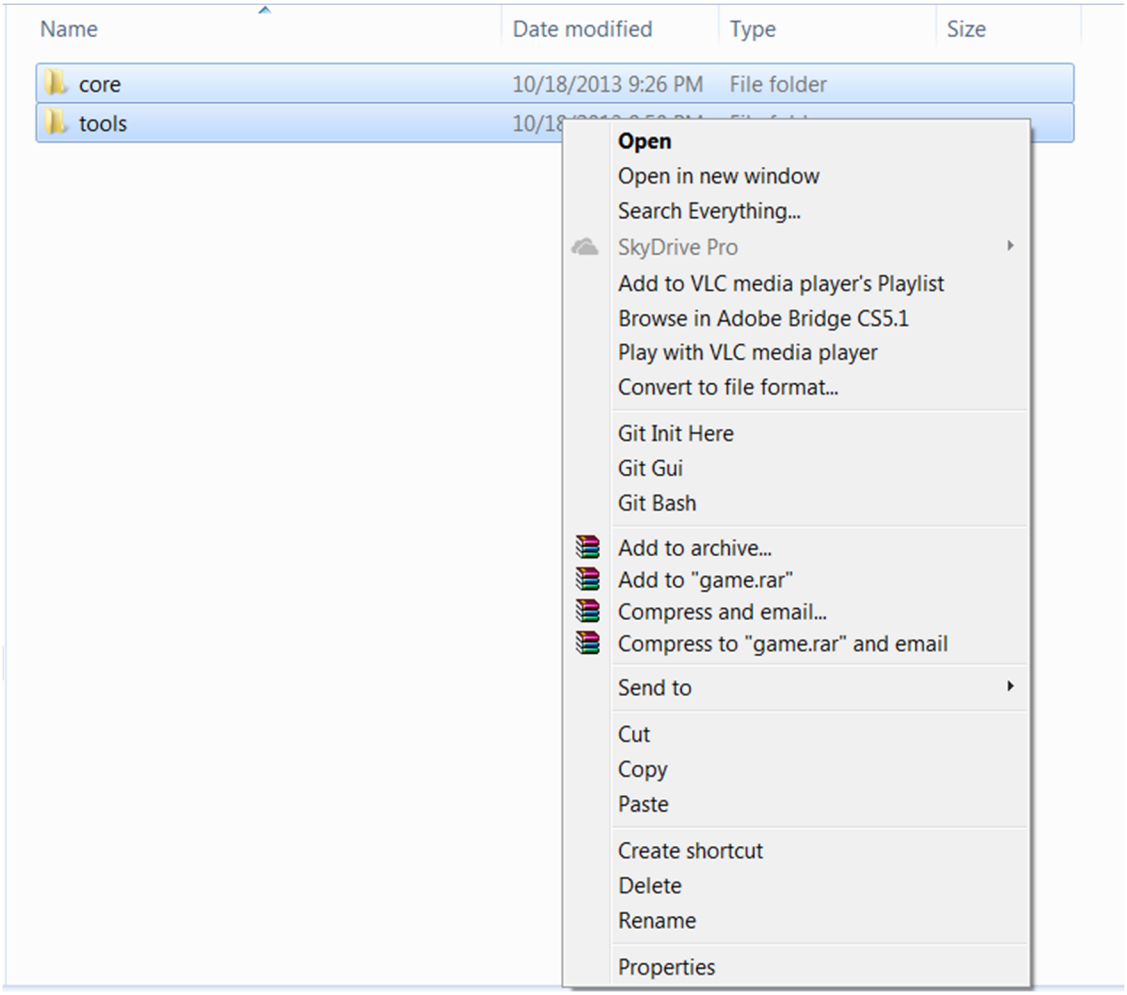
6. Go to **Project Manager** and create a new project or if you want to use you existing project, make sure you use **generateAllProjects** first.



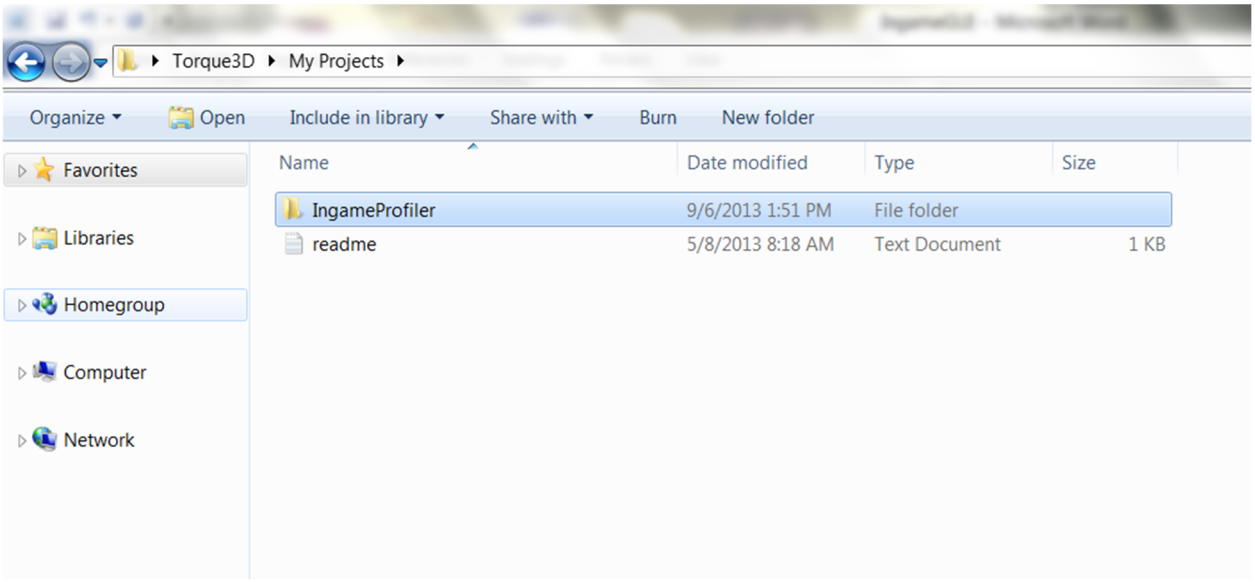




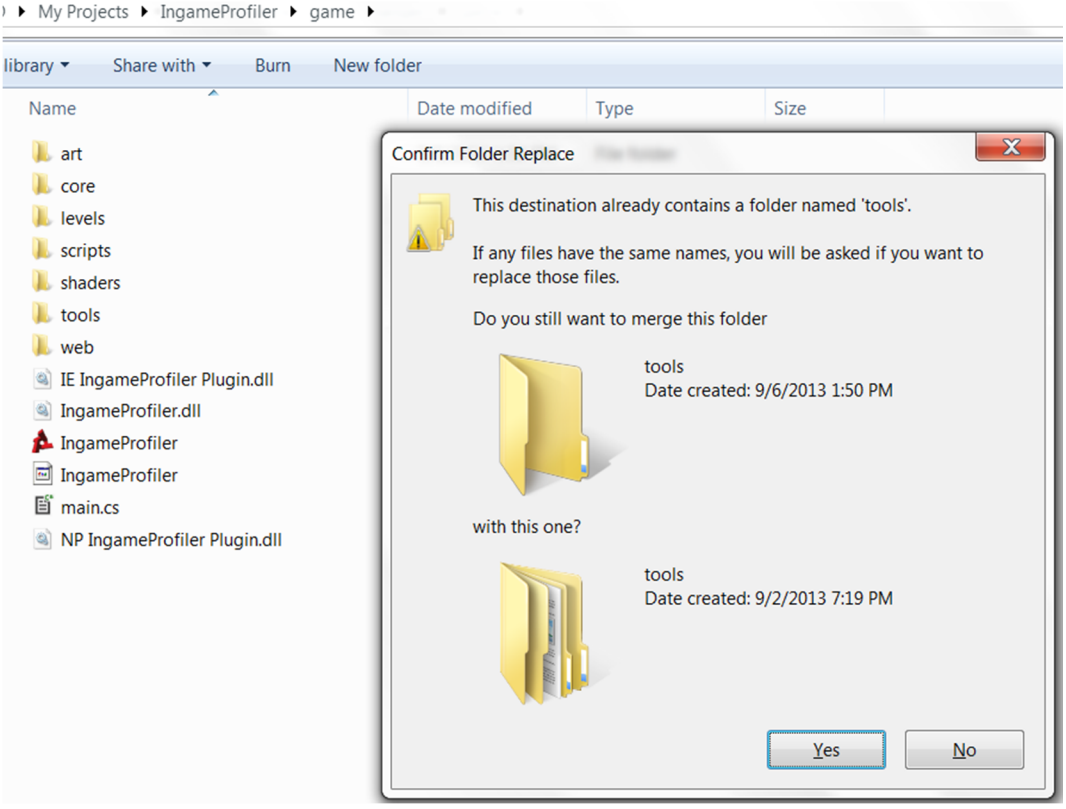
7. Now go back to the extracted directory and copy **tools** and **core** files inside the **game** folder.

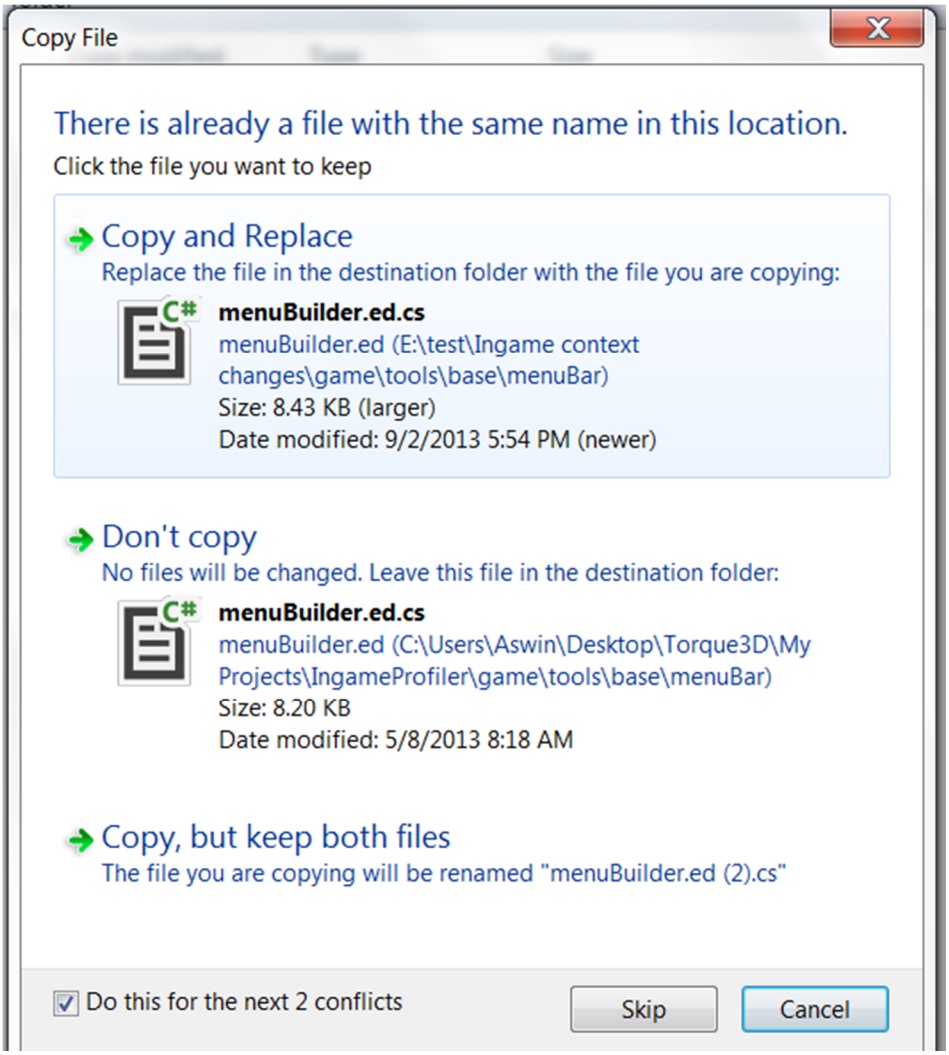


8. Now go to the newly created project **IngameProfiler** (in my case) under **My Projects**.

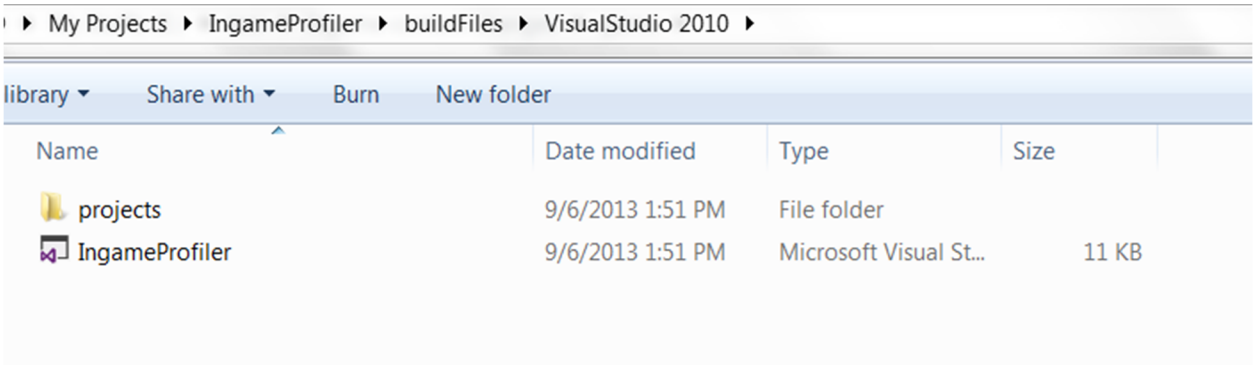


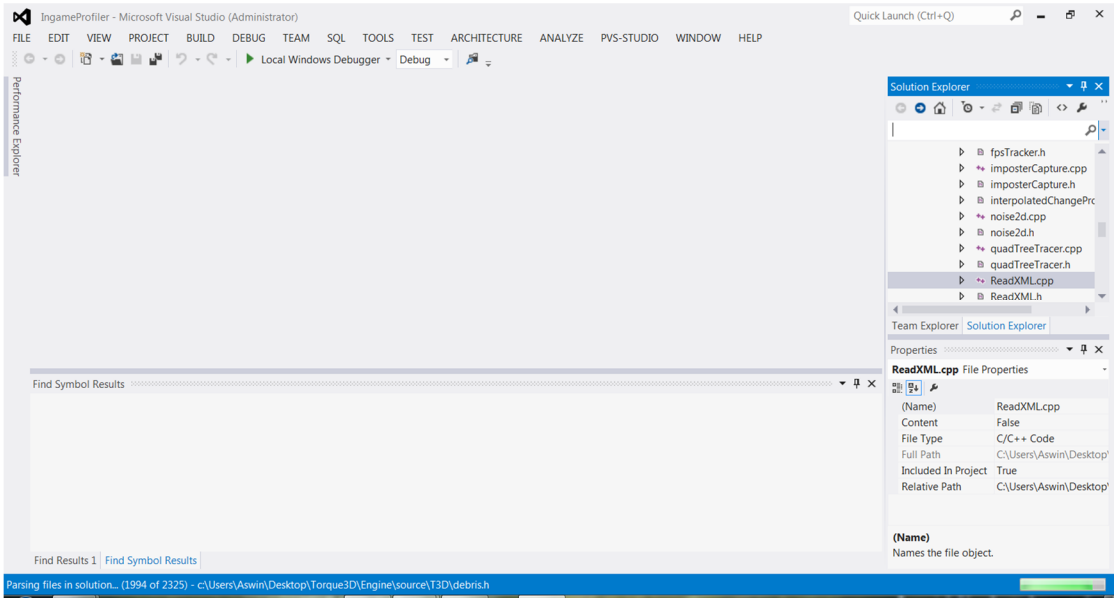
9. Go inside **game** folder and copy the contents.



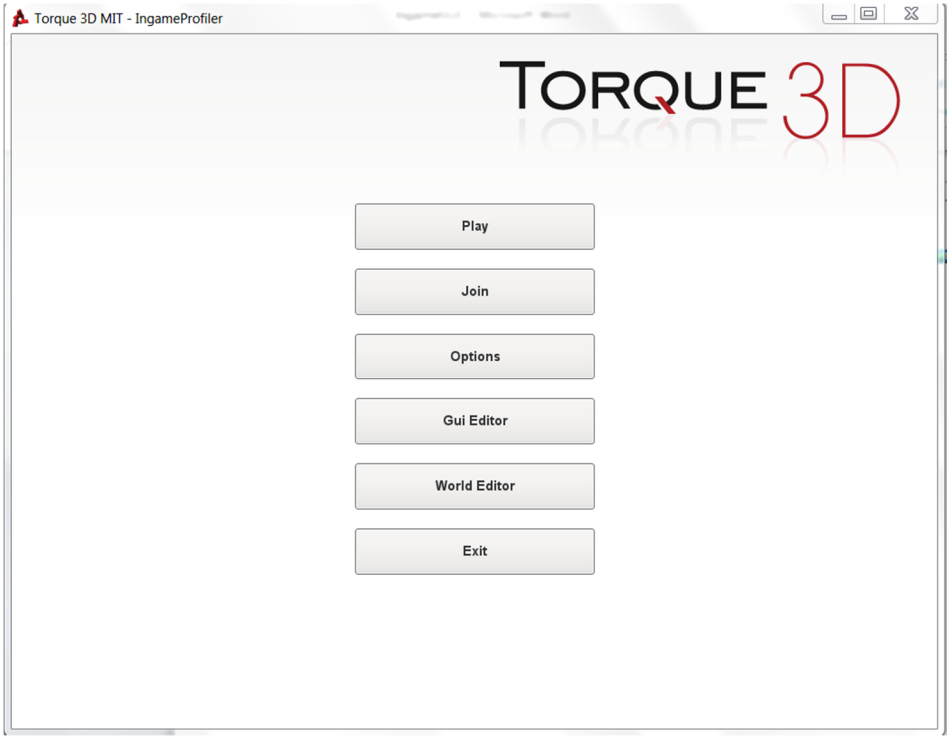


10. After the copy process is done, go to the **buildFiles/VisualStudio2010** folder or **buildFiles/VisualStudio2008** folder (in case of VS 2008). Remember the ***stock*** executable wouldn't work, you have to rebuild/recompile again.



11. Open up the **IngameProfiler** solution file. Will take some time to parse the files.
12. 

13. After all is done, debug the program. This will take a couple of minutes. Take a pee break. Make sure you have the **Microsoft SDK (latest if possible)**.

14. There you have it, the compiled version.


In-game GUI

Context Menu Options

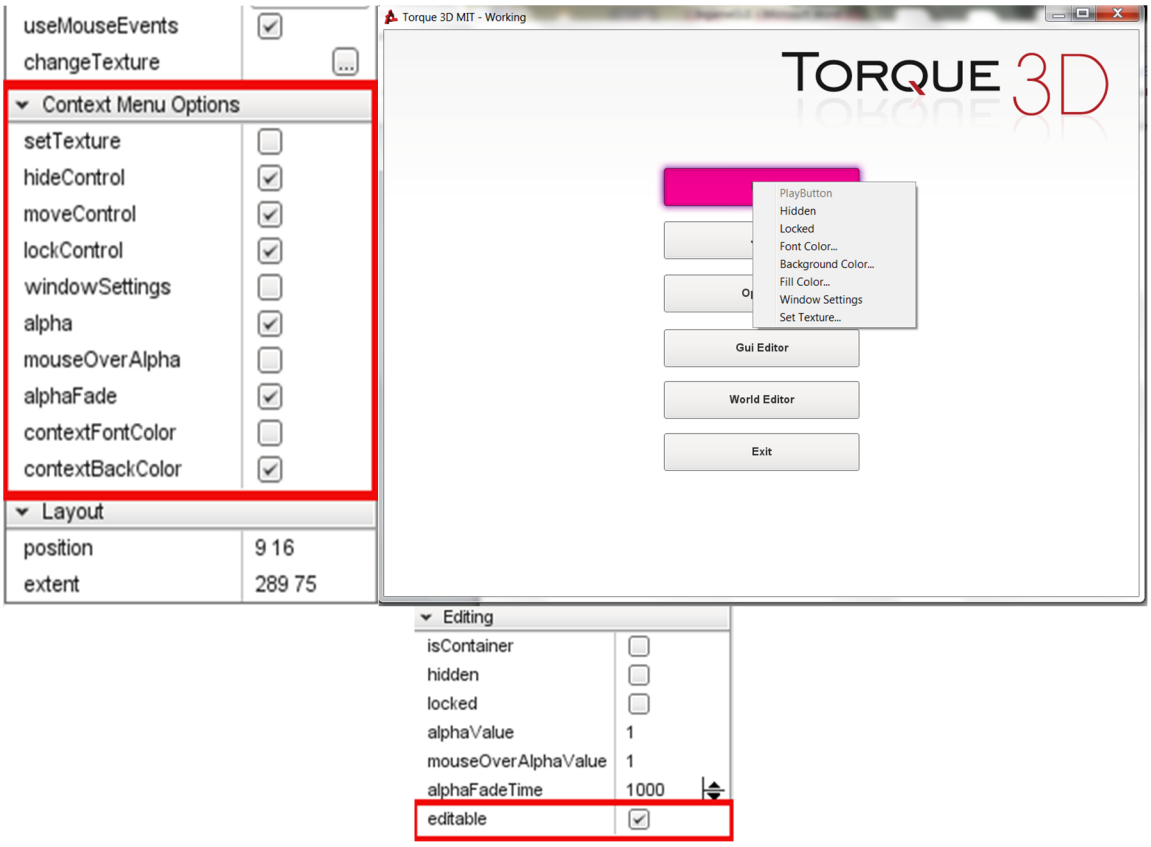
These are the options for the in-game context menu. The enabled options are displayed when right-clicked on the object. The “editable” attribute should be enabled.

CLASS USED:

ENGINE: Defined in the initPersistFields() of GuiControl Class (GuiControl.cpp)

addGroup(“Context Menu Options”);

SCRIPT: Defined in the ingameContext.cs under “/tools/ingameEditor/scripts”



Name of the control

The right click context menu, visible after enabling the “editable” attribute in the editor, shows up the **NAME** of the control. If the name exists, the name is shown, otherwise the class name is shown. This attribute in the context menu is not clickable.

CLASS USED:

SCRIPT: Defined in the ingameContext.cs under “/tools/ingameEditor/scripts”

```
if( %ctrl.getName() !$= "" )
      %popup.appendItem( %ctrl.getName() TAB "" TAB "");
else
      %popup.appendItem ( %ctrl.getClassName() TAB "" TAB "");

%popup.enableItem( 0, false);
```

Lock/Unlock control

This option is also a part of the right-click context menu, which is only shown if both "*editable*" and "*lock control*" attribute in the "Context menu option". It's not available for all the controls.

CLASS USED:

ENGINE: Defined in initPersistFields() of GuiControl class (GuiControl.cpp)

addProtectedField("lockControl")

FUNCTIONS USED:

ENGINE:

CONTEXT MENU

1. void setContextLockControl(bool lock)

   When enabled, displays the option for locking a control. The option for moving a control is also connected to it.

2. bool isContextLockable()

   Returns whether the control is lockable or not.

CONTROL LOCK (CONSOLE FUNCTIONS)

1. void setControlLock(bool lock)

   When enabled, locks the control.

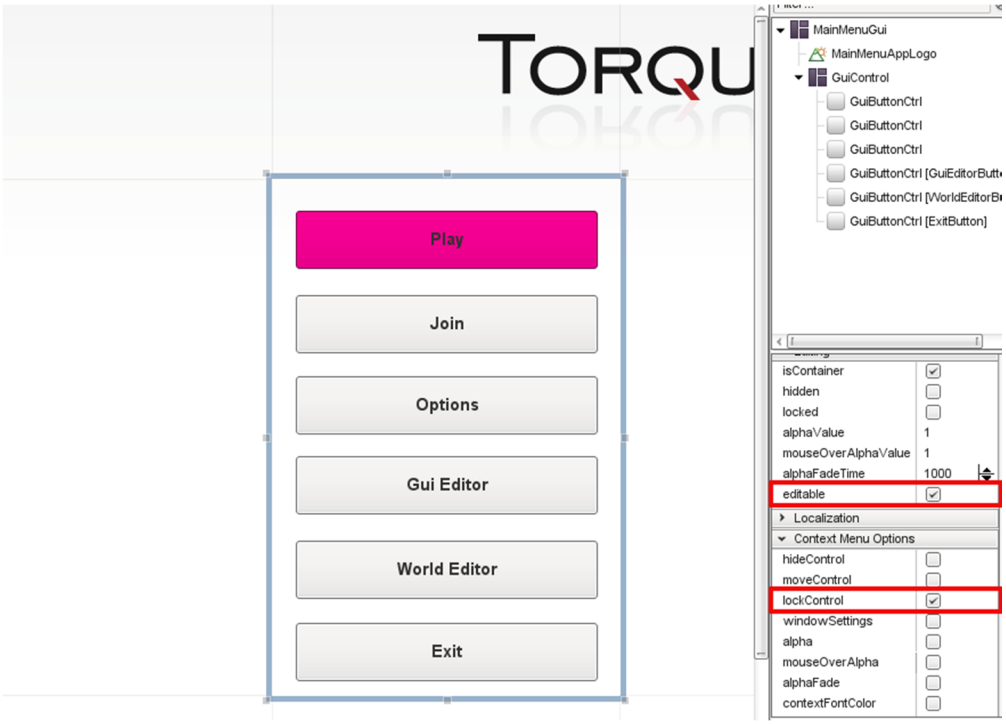
2. bool getControlLock()

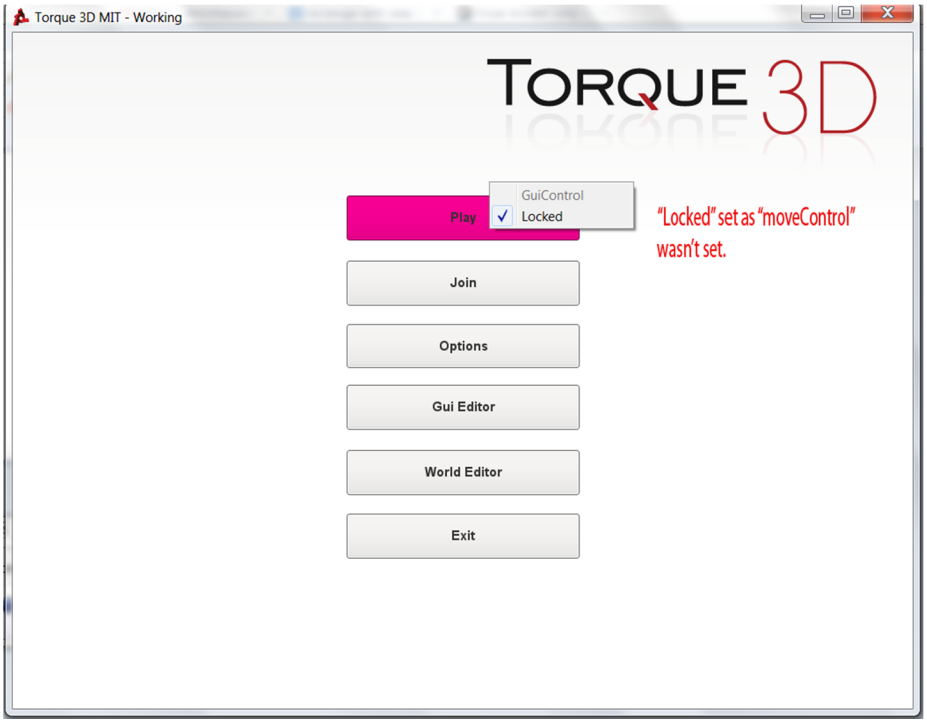
   Returns if the control is locked or not.

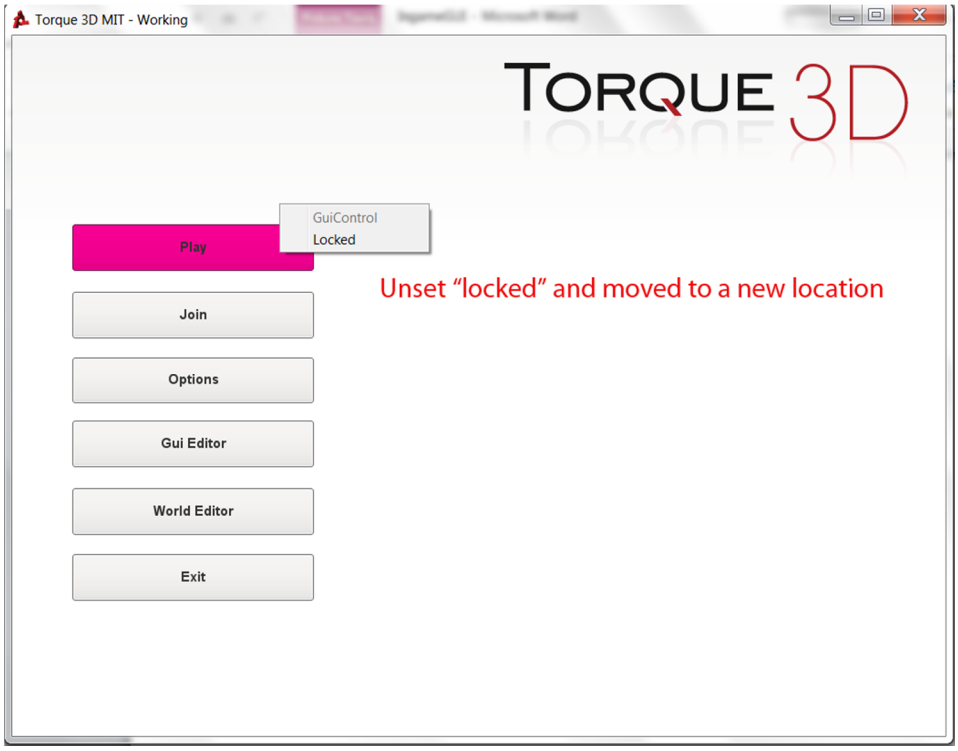
SCRIPT:

```
%lockPos = %popup.appendItem("Locked" TAB "" TAB "%this.object.setControlLock(!%this.object.getControlLock());");

%popup.checkItem(%lockPos, %ctrl.getControlLock());
```


TORQU
Play
Join
Options
Gui Editor
World Editor
Exit
MainMenuGui
MainMenuAppLogo
GuiControl
GuiButtonCtrl
GuiButtonCtrl
GuiButtonCtrl
GuiButtonCtrl [GuiEditorButt
GuiButtonCtrl [WorldEditorB
GuiButtonCtrl [ExitButton]
isContainer
hidden
locked
alphaValue 1
mouseOverAlphaValue 1
alphaFadeTime 1000
editable
Localization
Context Menu Options
hideControl
moveControl
lockControl
windowSettings
alpha
mouseOverAlpha
alphaFade
contextFontColor


Torque 3D MIT - Working
TORQUE 3D
GuiControl
Play
Locked
"Locked" set as "moveControl" wasn't set.
Join
Options
Gui Editor
World Editor
Exit



Alpha level of the control

The alpha level of the control can be changed through the "Window settings" when right-clicked. For this option to be enabled, the "editable" and "alpha" field should be enabled. You need to check the "windowSettings" option as well.

In the editor, this can be changed through the field "alphaValue".

This will change the alpha value of the control and also of its immediate children.

CLASS USED:

ENGINE:

CONTEXT MENU

- Defined in initPersistFields() of GuiControl (GuiControl.cpp)

  addProtectedField("alpha");

CONTROL ALPHA

- Defined in initPersistFields() of GuiControl (GuiControl.cpp)

addProtectedField(“alphaValue”) under group (“Editing”)

FUNCTIONS USED:

ENGINE:

CONTEXT MENU

1. void setContextAlpha(bool alpha)

   When enabled, displays the option for setting the alpha value.

2. bool isContextAlphaEnabled()

   Returns whether the alpha option is enabled or not.

CONTROL ALPHA VALUE (CONSOLE FUNCTIONS)

1. void setAlphaValue(F32 alpha)

   Sets the alpha value of the control.

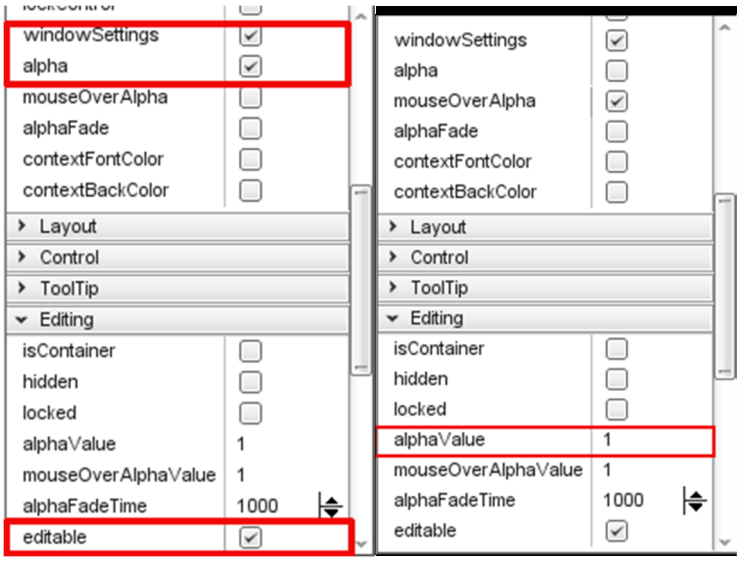
2. F32 getAlphaValue()

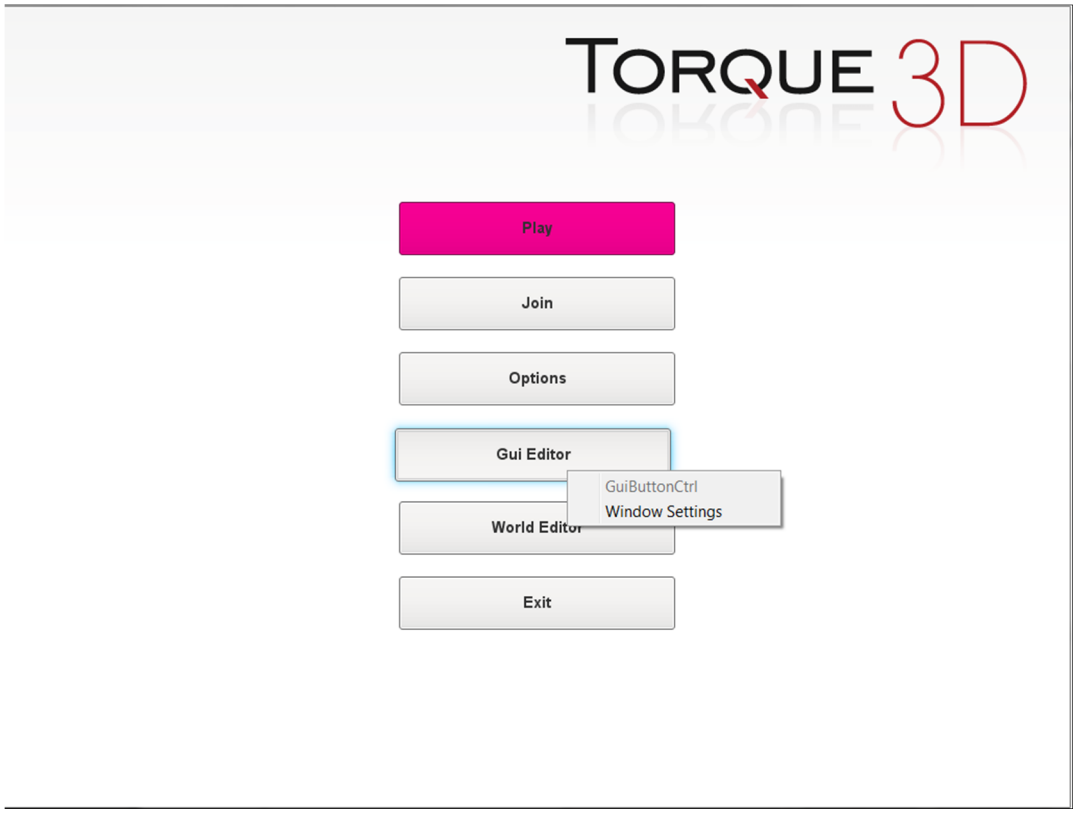
   Returns the alpha value of the control.

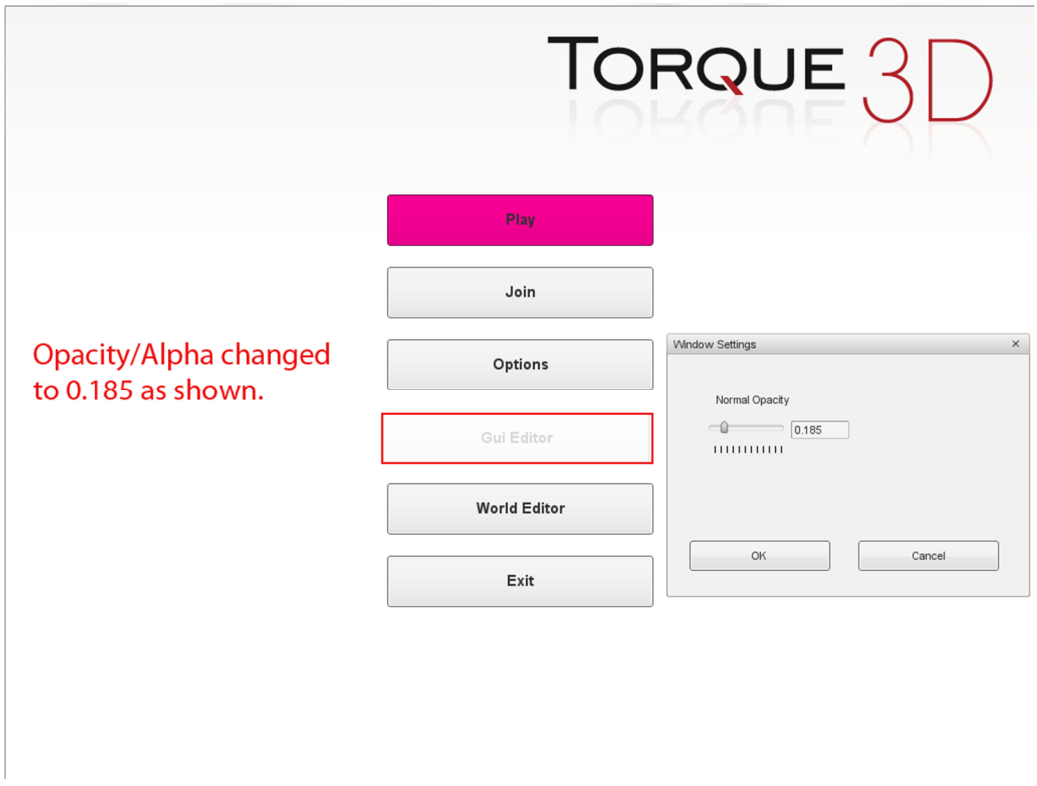
SCRIPT:

```
ctrl.setAlphaValue( %value );
```

```
%this.oldAlphaValue = %ctrl.getAlphaValue();
```







Opacity/Alpha changed to 0.185 as shown.

Alpha fade of the control

The alpha fade of the control can be changed through the “Window settings” when right-clicked. For this option to be enabled, the “editable” and “alphaFade” field should be enabled. You need to check the “windowSettings” option as well.

In the editor, this can be changed through “alphaFadeTime”.

CLASS USED:

ENGINE:

CONTEXT MENU

- Defined in initPersistFields() of GuiControl (GuiControl.cpp)

  addProtectedField(“alphaFade”);

CONTROL ALPHA FADE

- Defined in initPersistFields() of GuiControl (GuiControl.cpp)

  addProtectedField(“alphaFadeTime”) under group (“Editing”)

FUNCTIONS USED:

ENGINE:

CONTEXT MENU

1. void setContextAlphaFade(bool alphaFade)

   When enabled, displays the option for setting the alpha fade time.

2. bool isContextAlphaFadeEnabled()

   Returns whether the alpha fade time option is enabled or not.

CONTROL ALPHA FADE TIME (CONSOLE FUNCTIONS)

1. void setAlphaFadeTime(S32 alphaFade)

   Sets the alpha fade time of the control.

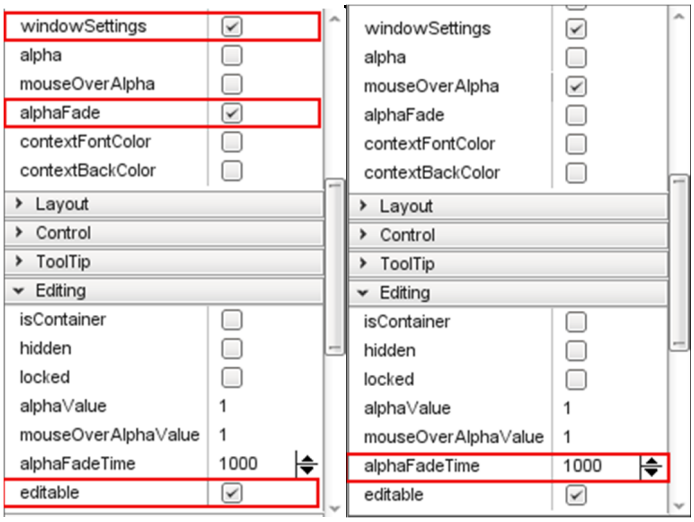
2. S32 getAlphaFadeTime()

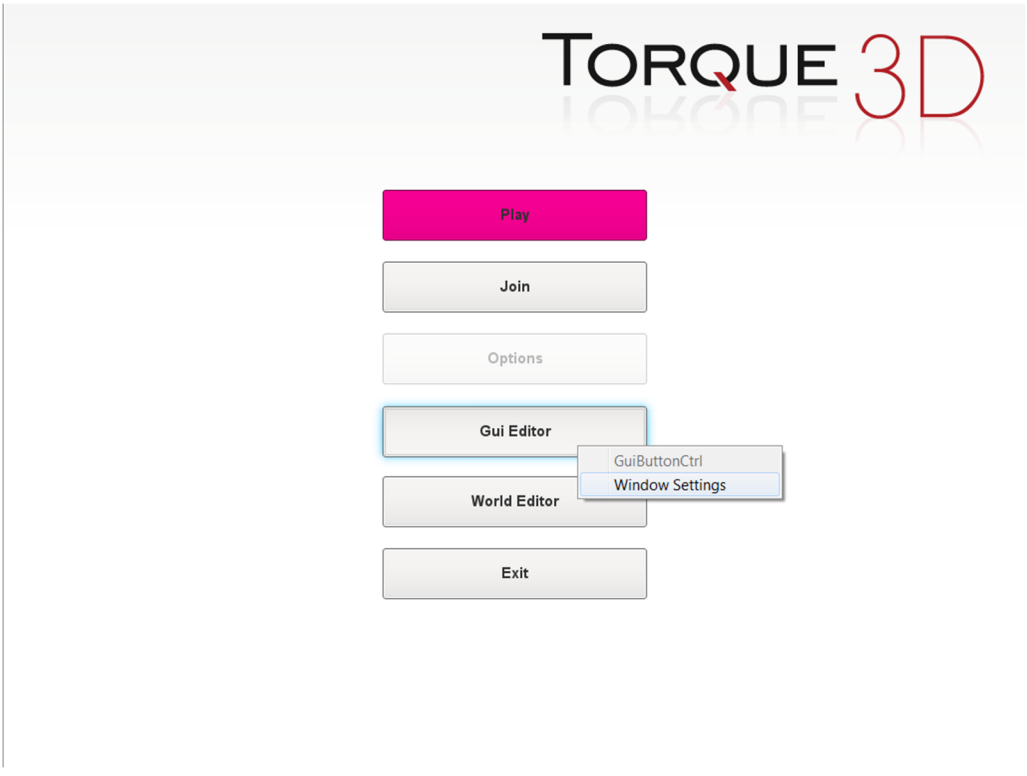
   Returns the alpha fade time of the control.

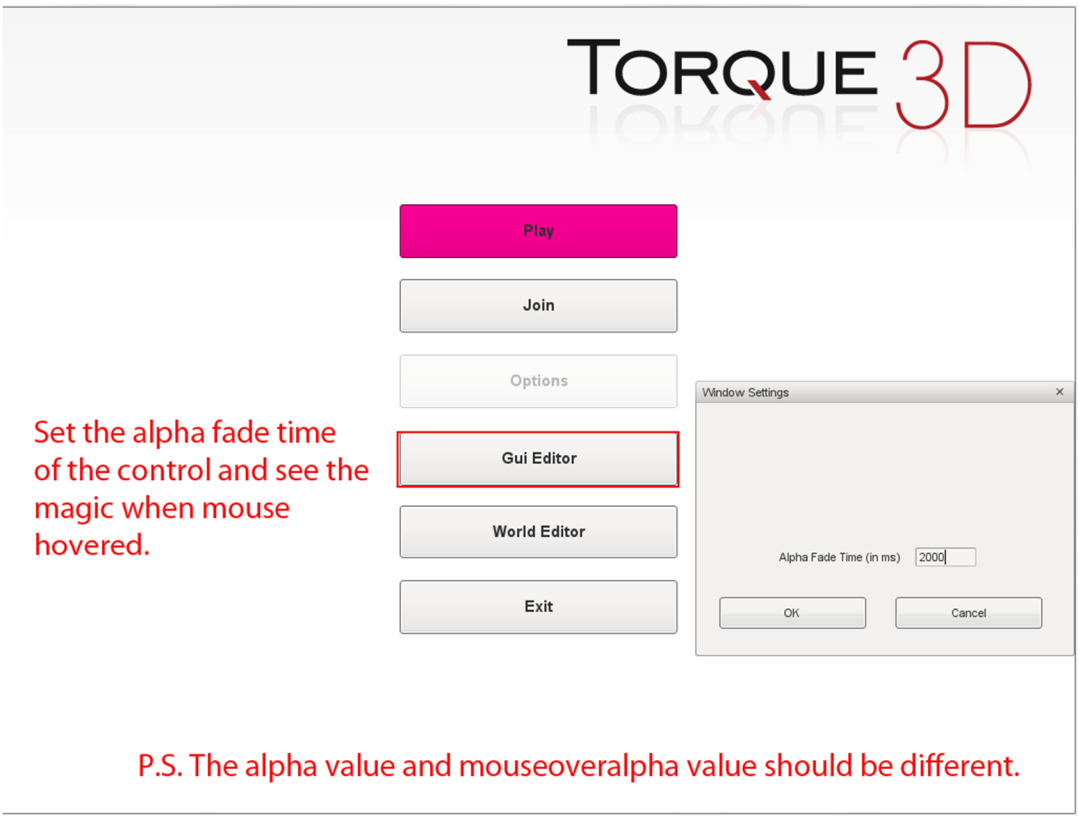
SCRIPT:

```
%this.ctrl.setAlphaFadeTime( %value );

%this.oldAlphaFadeTime = %ctrl.getAlphaFadeTime();
```







Mouse-over alpha of the control

The mouse-over alpha of the control can be changed through the “Window settings” when right-clicked. For this option to be enabled, the “editable” and “mouseOverAlpha” field should be enabled. You need to check the “windowSettings” option as well.

In the editor, this can be changed through “mouseOverAlphaValue”.

This will change the mouse-over alpha value of the control and also of its immediate children.

CLASS USED:

ENGINE:

CONTEXT MENU

- Defined in initPersistFields() of GuiControl (GuiControl.cpp)

  addProtectedField(“mouseOverAlpha”);

CONTROL MOUSE-OVER ALPHA

- Defined in initPersistFields() of GuiControl (GuiControl.cpp)

  addProtectedField(“mouseOverAlphaValue”) under group (“Editing”)

FUNCTIONS USED:

ENGINE:

CONTEXT MENU

1. void setMouseOverAlpha(bool mouseOverAlpha)

   When enabled, displays the option for changing the mouse-over alpha value.

2. bool isContextMouseOverAlphaEnabled()

   Returns whether the mouse-over alpha option is enabled or not.

CONTROL MOUSE-OVER ALPHA (CONSOLE FUNCTIONS)

1. void setMouseOverAlphaValue(F32 alpha)

   Sets the mouse-over alpha value of the control.

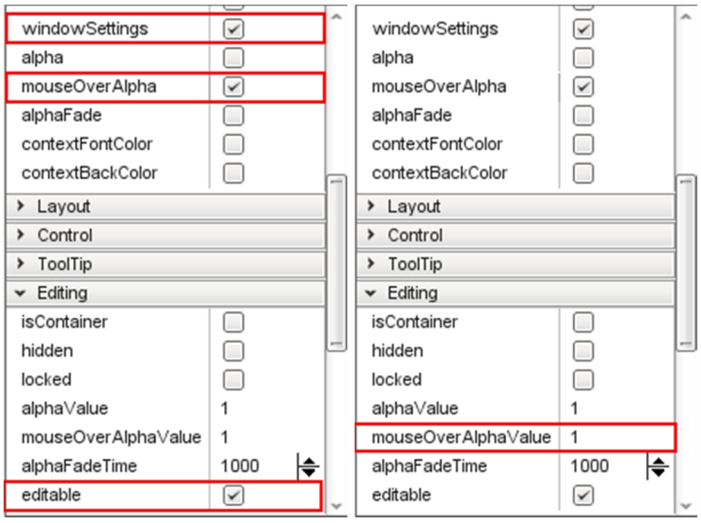
2. F32 getMouseOverAlphaValue()

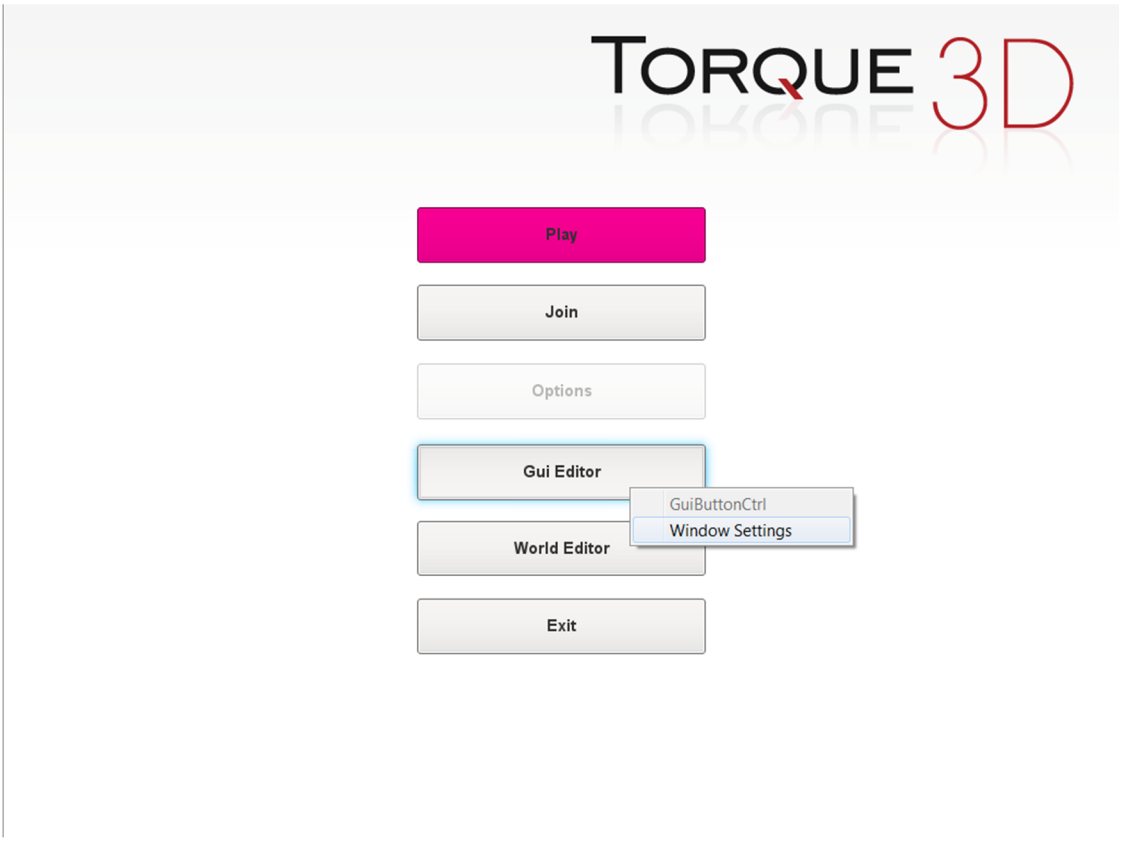
   Returns the mouse-over alpha value of the control.

SCRIPT:

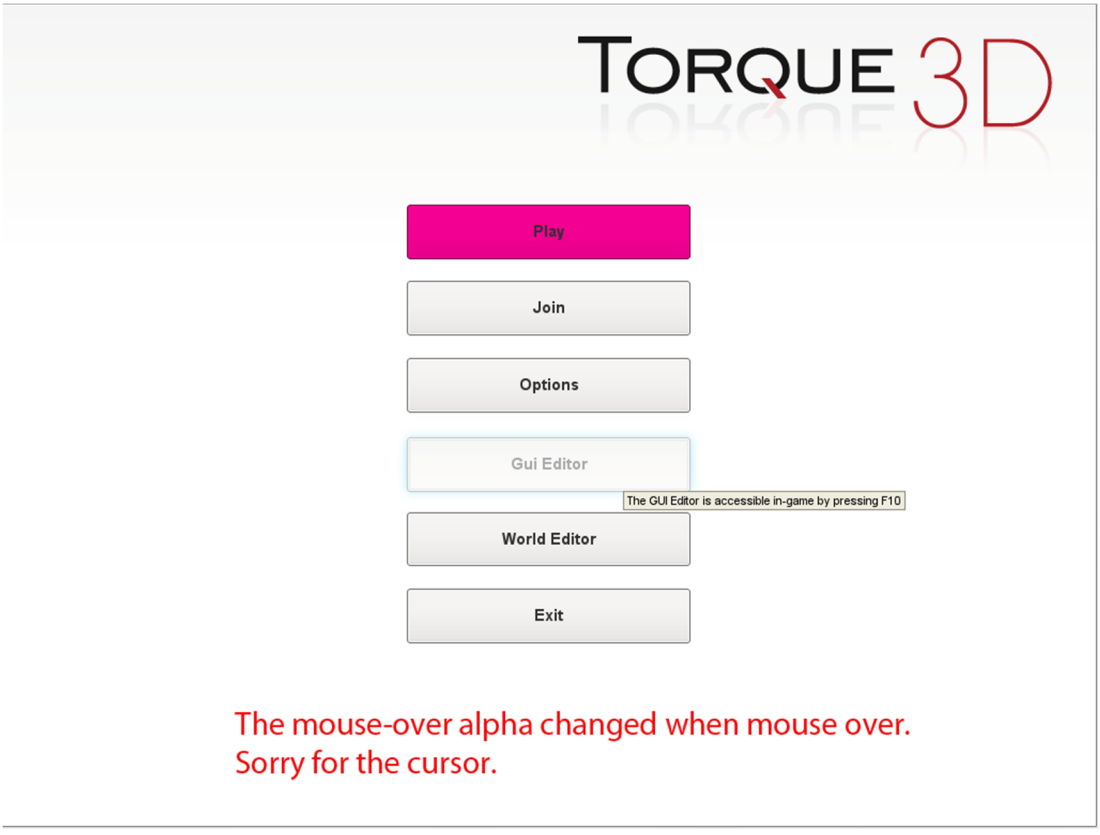
```
%this.ctrl.setMouseOverAlphaValue( %value );

 %this.oldMouseOverAlphaValue = %ctrl.getMouseOverAlphaValue();
```


windowSettings
alpha
mouseOverAlpha
alphaFade
contextFontColor
contextBackColor
Layout
Control
ToolTip
Editing
isContainer
hidden
locked
alphaValue 1
mouseOverAlphaValue 1
alphaFadeTime 1000
editable
windowSettings
alpha
mouseOverAlpha
alphaFade
contextFontColor
contextBackColor
Layout
Control
ToolTip
Editing
isContainer
hidden
locked
alphaValue 1
mouseOverAlphaValue 1
alphaFadeTime 1000
editable


TORQUE 3D
Play
Join
Options
Gui Editor
GuiButtonCtrl
Window Settings
World Editor
Exit

Mouse-over alpha/opacity changed to 0.4 as shown.



The mouse-over alpha changed when mouse over.
Sorry for the cursor.

Set title of the window control

The title of the window control can be changed through “Set title…” when right-clicked. For this option to be enabled, the “editable” and “setTitle” field should be enabled.

CLASS USED:

ENGINE:

CONTEXT MENU

- Defined in initPersistFields() of GuiWindowCtrl (GuiWindowCtrl.cpp)

  addProtectedField(“setTitle”);

FUNCTIONS USED:

ENGINE:

CONTEXT MENU

1. void setContextTitle(bool title)

   When enabled, displays the option for changing the title.

2. bool isTitleSet()

   Returns whether the “set title” option is enabled or not.

CONTROL SET TITLE (CONSOLE FUNCTIONS)

1. void setWIndowTitle(const char *title)

   Sets the title of the window control.

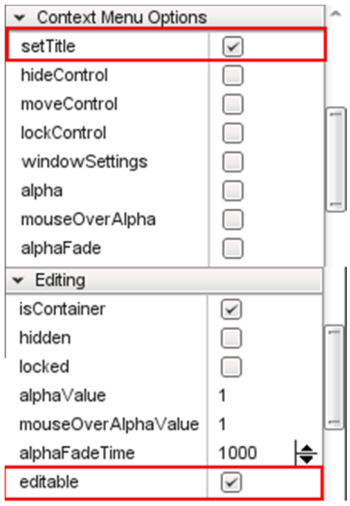
2. const char *getWindowTitle()

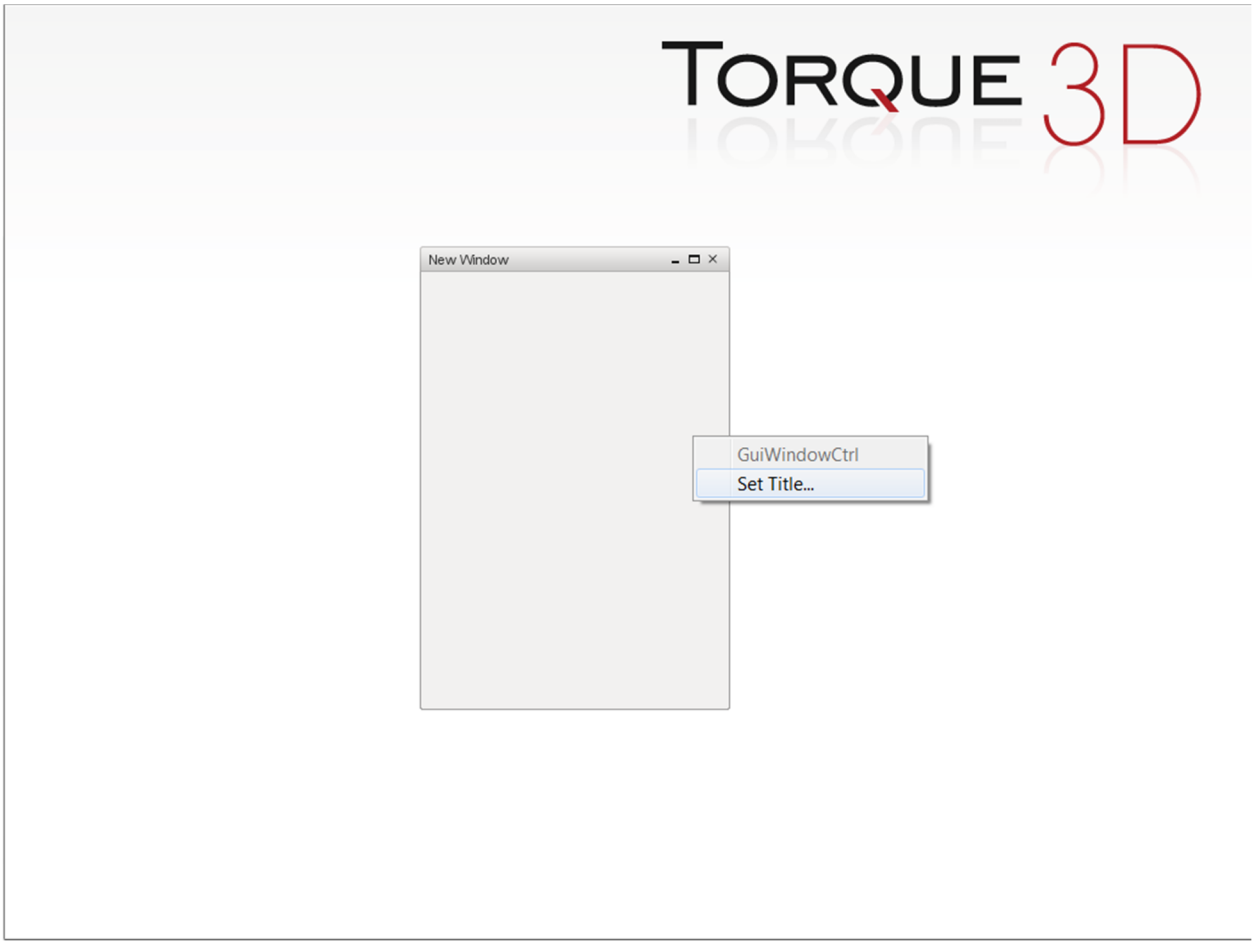
   Returns the title of the window control.

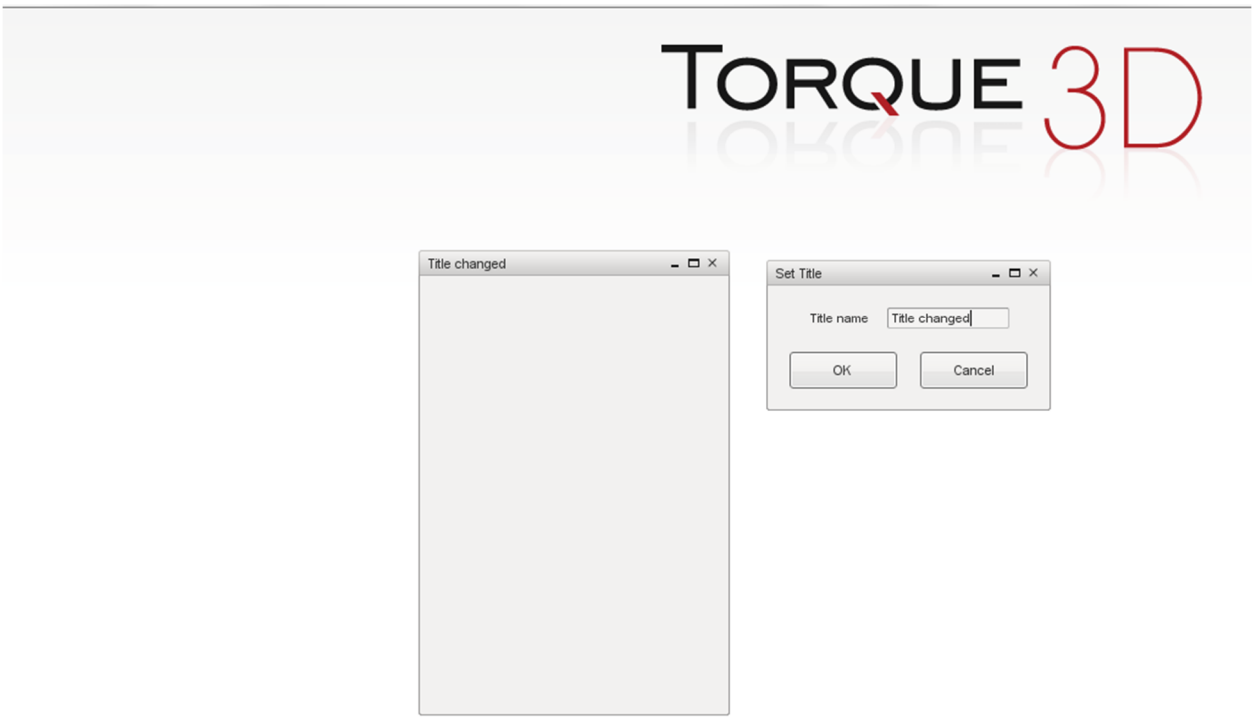
SCRIPT:

```
%this.ctrl.setWindowTitle(%value);

%this.oldTitleName = %ctrl.getWindowTitle();
```


Context Menu Options
setTitle
hideControl
moveControl
lockControl
windowSettings
alpha
mouseOverAlpha
alphaFade
Editing
isContainer
hidden
locked
alphaValue 1
mouseOverAlphaValue 1
alphaFadeTime 1000
editable


TORQUE 3D
New Window
GuiWindowCtrl
Set Title...



The title of the window changed to "Title changed" as shown.

Font color of the control

The font color of the control can be changed through "Font color…" when right-clicked. For this option to be enabled, the "editable" and "contextFontColor" field should be enabled.

In the editor, this can be changed through "controlFontColor".

This will change the font color of the control and also of its immediate children.

CLASS USED:

ENGINE:

CONTEXT MENU

- Defined in initPersistFields() of GuiControl (GuiControl.cpp)

  addProtectedField("contextFontColor");

CONTROL FONT COLOR

- Defined in initPersistFields() of GuiControl (GuiControl.cpp)

  addField("controlFontColor"); under group ("Control")

FUNCTIONS USED:

ENGINE:

CONTEXT MENU

1. void setContextFontColor(bool fontColor)

   When enabled, displays the option for changing the font color.

2. bool canChangeContextFontColor()

   Returns whether the "context font color" option is enabled or not.

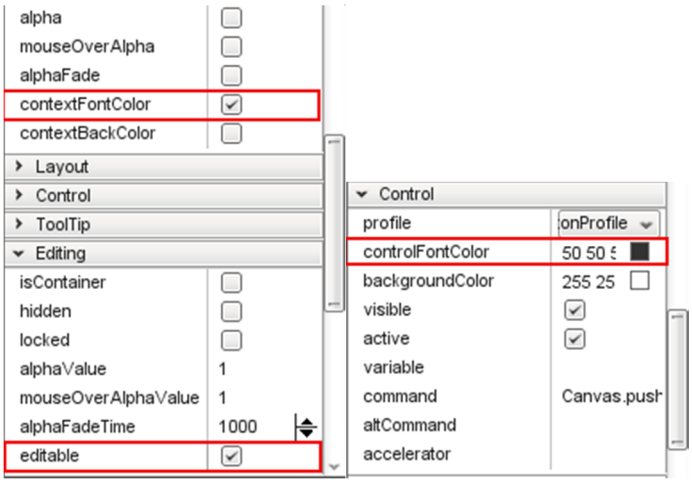
CONTROL FONT COLOR (CONSOLE FUNCTIONS)

1. void setControlFontColor(ColorI color)

   Sets the font color of the control.

SCRIPT:

```
%fontColorPos = %popup.appendItem("Font Color..." TAB "" TAB "getColorI(  %this.object.controlFontColor , %this.object
 @ \".setControlFontColor\", %this.object.getRoot(), %this.object @ \".setControlFontColor\", %this.object
@ \".setControlFontColor\" );");
```



The font color of the control is changed to pink.

Fill color of the control

The fill color of the control can be changed through “Fill color…” when right-clicked. For this option to be enabled, the “editable” and “contextFillColor” field should be enabled.

In the editor, this can be changed through “controlFillColor”.

This setting will not work with transparent controls.

CLASS USED:

ENGINE:

CONTEXT MENU

- Defined in initPersistFields() of GuiControl (GuiControl.cpp)

  addProtectedField(“contextFillColor”);

CONTROL FILL COLOR

- Defined in initPersistFields() of GuiControl (GuiControl.cpp)

  addField(“controlFillColor”); under group (“Control”)

FUNCTIONS USED:

ENGINE:

CONTEXT MENU

1. void setContextFillColor(bool fillColor)

   When enabled, displays the option for changing the fill color.

2. bool canChangeContextFillColor()

   Returns whether the “context fill color” option is enabled or not.

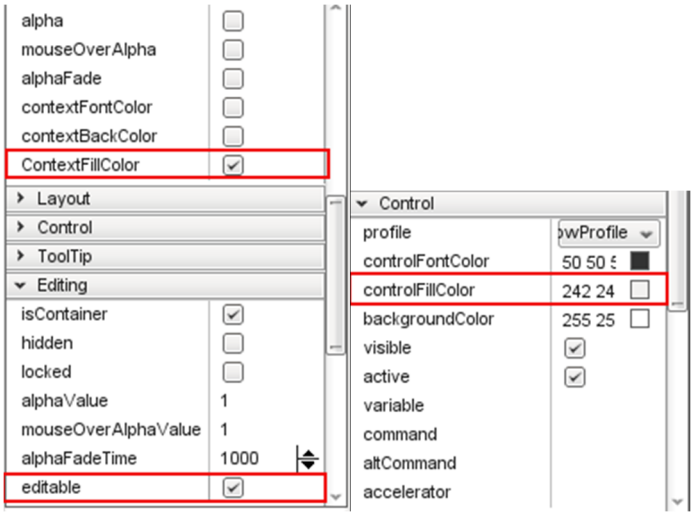
CONTROL FILL COLOR (CONSOLE FUNCTIONS)

1. void setControlFillColor(ColorI color)

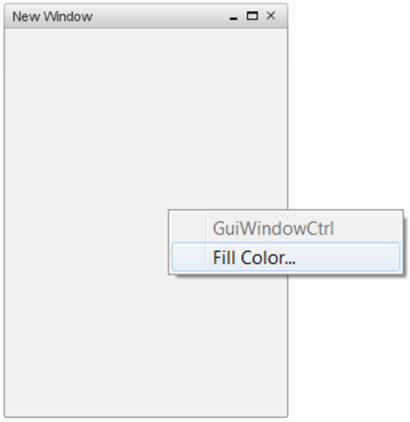
   Sets the fill color of the control.

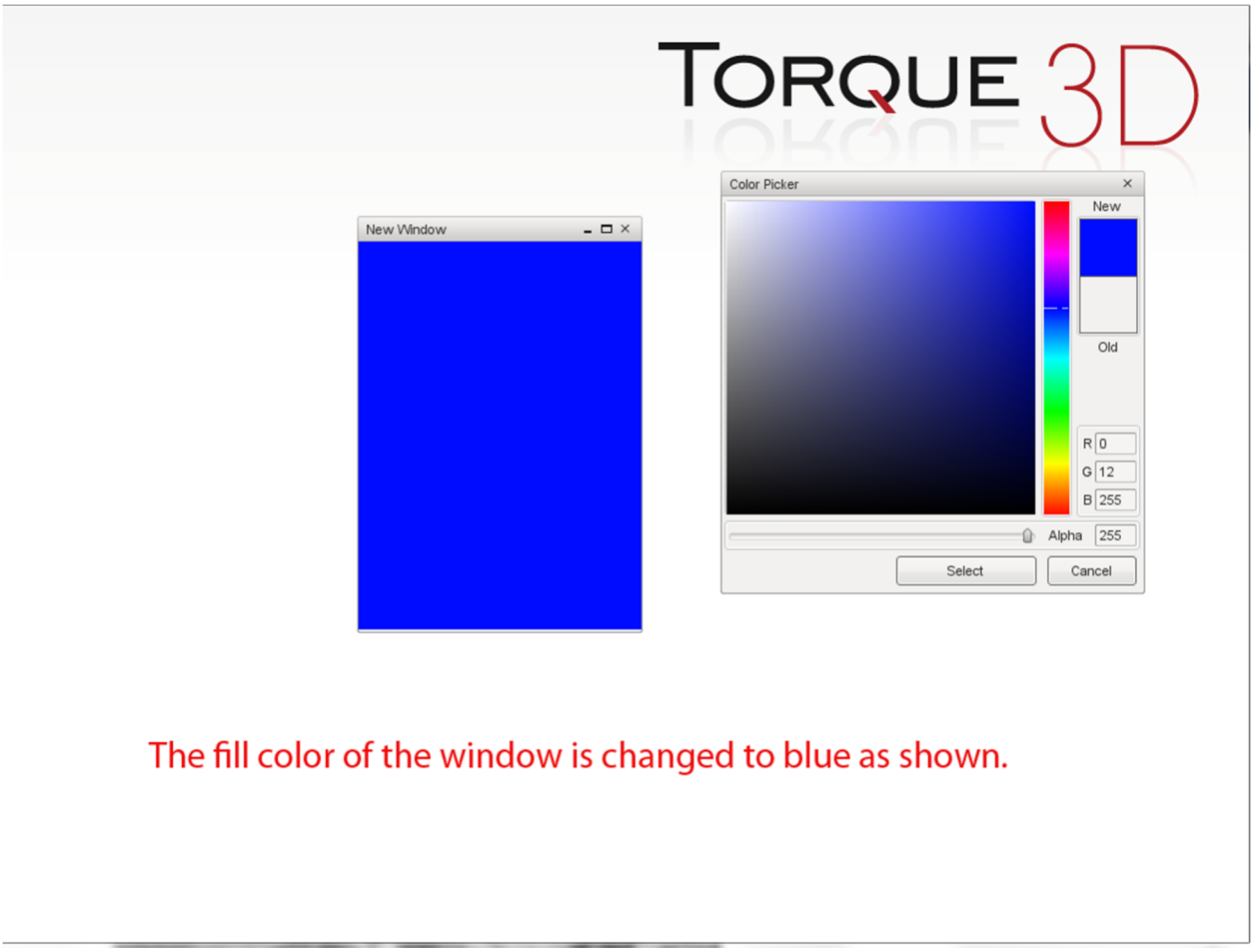
SCRIPT:

```
%fillColorPos = %popup.appendItem("Fill Color..." TAB "" TAB "getColorI(  %this.object.controlFillColor ,
%this.object @ \".setControlFillColor\", %this.object.getRoot(), %this.object @ \".setControlFillColor\",
%this.object @ \".setControlFillColor\" );");
```


alpha
mouseOverAlpha
alphaFade
contextFontColor
contextBackColor
ContextFillColor
Layout
Control
ToolTip
Editing
isContainer
hidden
locked
alphaValue 1
mouseOverAlphaValue 1
alphaFadeTime 1000
editable
Control
profile
controlFontColor
controlFillColor 242 24
backgroundColor 255 25
visible
active
variable
command
altCommand
accelerator

TORQUE 3D


New Window
GuiWindowCtrl
Fill Color...



Background color of the control

The background color of the control can be changed through “Background color…” when right-clicked. For this option to be enabled, the “editable” and “contextBackColor” field should be enabled.

In the editor, this can be changed through “backgroundColor”.

This setting will not work with transparent controls.

CLASS USED:

ENGINE:

CONTEXT MENU

- Defined in initPersistFields() of GuiControl (GuiControl.cpp)

  addProtectedField(“contextBackColor”);

CONTROL BACKGROUND COLOR

- Defined in initPersistFields() of GuiControl (GuiControl.cpp)

  addField("backgroundColor"); under group ("Control")

FUNCTIONS USED:

ENGINE:

CONTEXT MENU

1. void setContextBackColor(bool backColor)

   When enabled, displays the option for changing the background color.

2. bool canChangeContextBackColor()

   Returns whether the "background color" option is enabled or not.

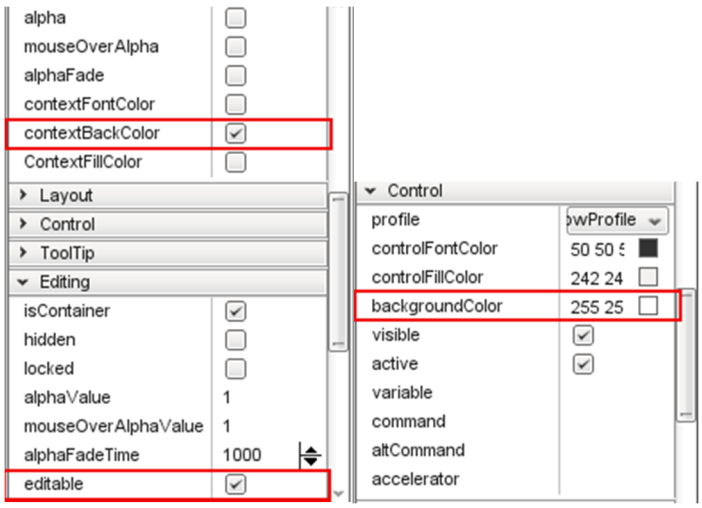
CONTROL BACKGROUND COLOR (CONSOLE FUNCTIONS)

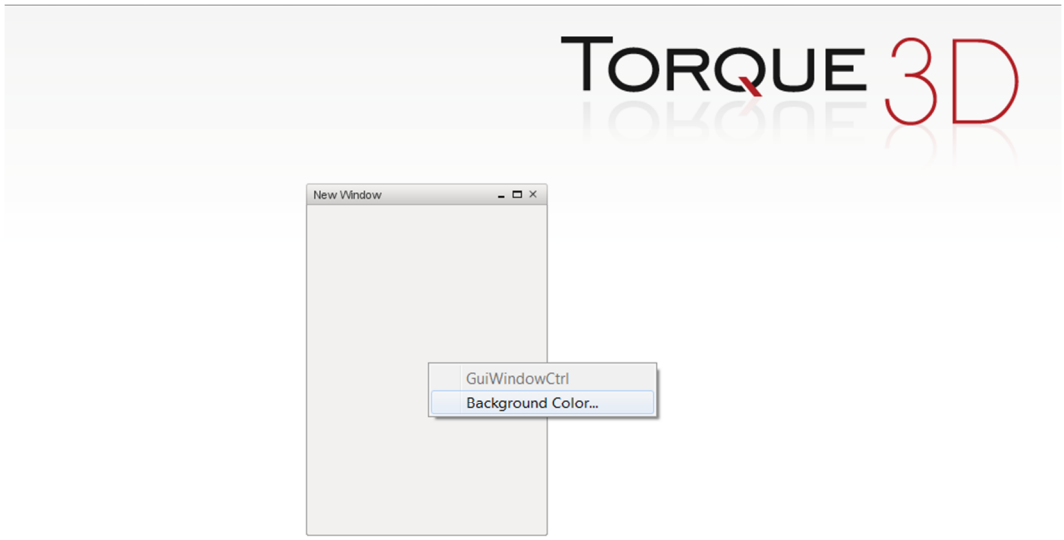
1. void setControlBackgroundColor(ColorI color)

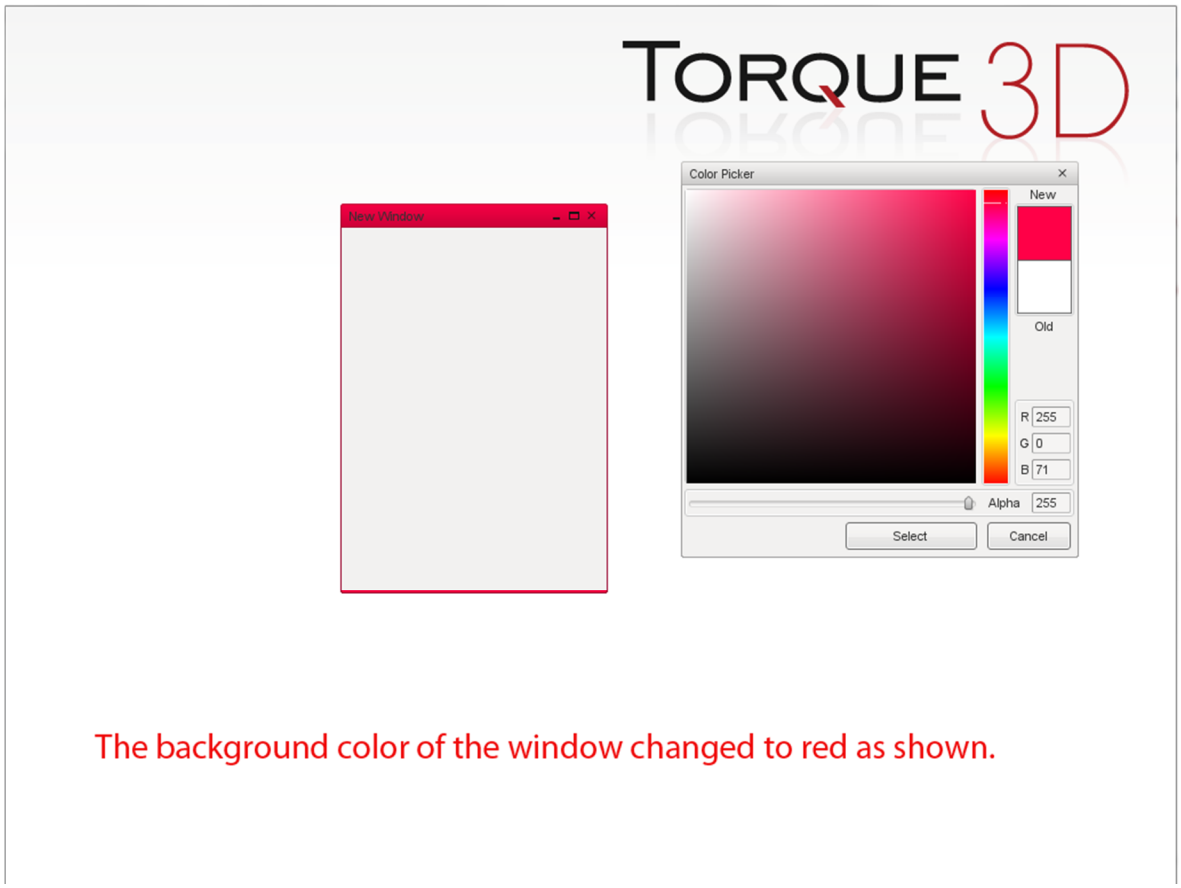
   Sets the background color of the control.

SCRIPT:

```
%backgroundColorPos = %popup.appendItem("Background Color..." TAB "" TAB "getColorI(  %this.object.backgroundColor ,
%this.object @ \".setControlBackgroundColor\", %this.object.getRoot(), %this.object @ \".setControlBackgroundColor\",
 %this.object @ \".setControlBackgroundColor\" );");
```







The background color of the window changed to red as shown.

Max extent of the control

The max extent of the control can be changed by stretching the control.

In the editor, this can be achieved through the field “maxExtent”.

CLASS USED:

ENGINE:

CONTROL MAX EXTENT

- Defined in initPersistFields() of GuiControl (GuiControl.cpp)

  addField(“maxExtent”); under group (“Layout”)

FUNCTIONS USED:

ENGINE:

CONTROL BACKGROUND COLOR (CONSOLE FUNCTIONS)

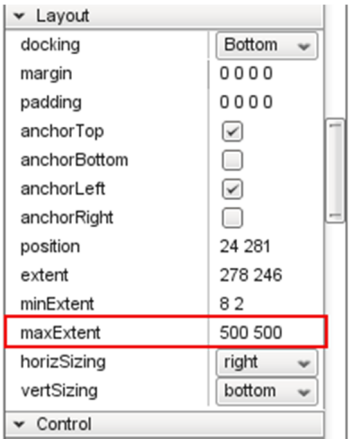
1. void setMaxExtent(Point2I maxExtent)

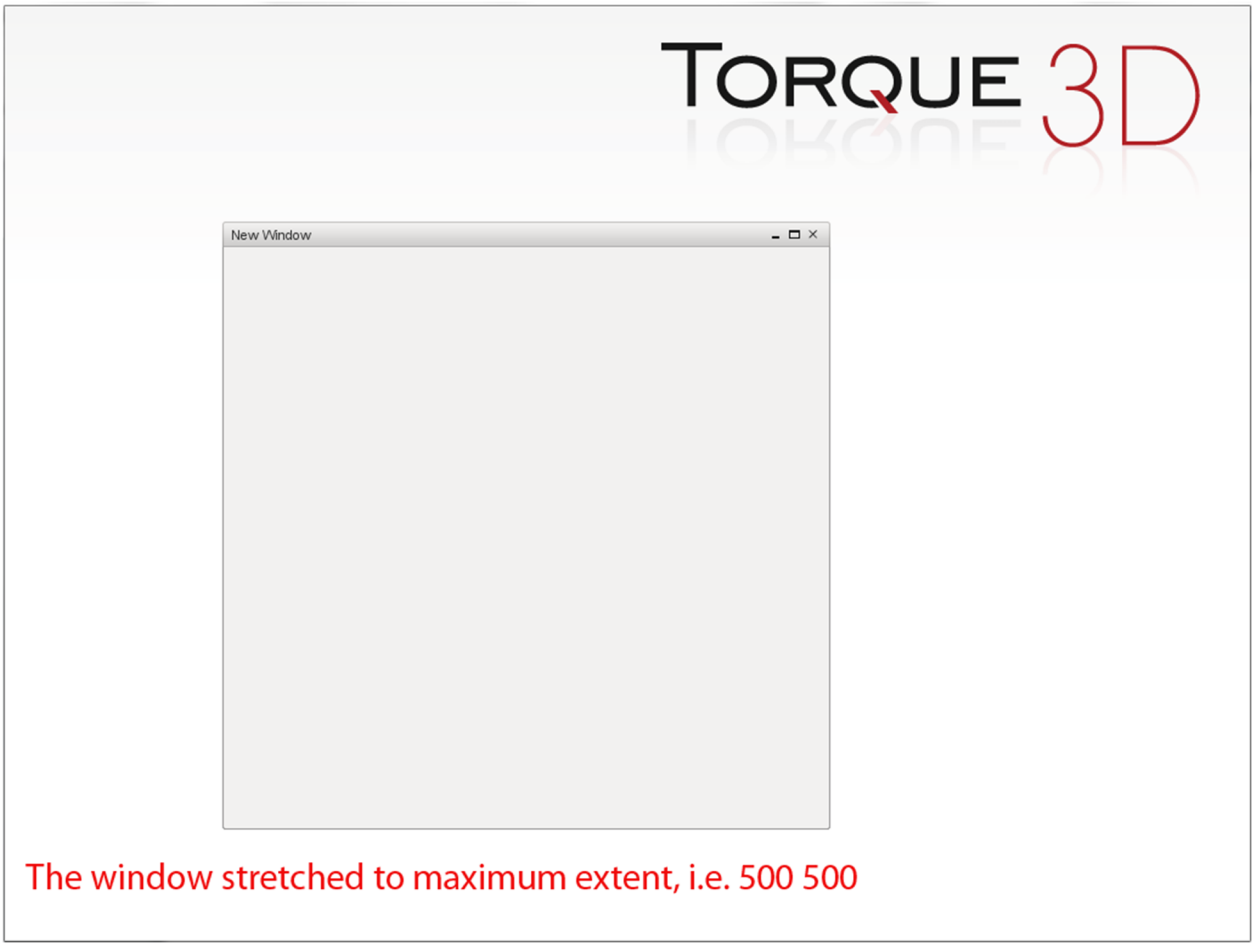
   Sets the maximum extent of the control.

2. Point2I getMaxExtent()

   Returns the maximum extent of the control.

SCRIPT:





Save to XML file and reload

The settings of any control can be saved into an XML file and reloaded.

CLASS USED:

ENGINE:

XML READ

- Defined in initPersistFields() of ReadXML (ReadXML.cpp)

FUNCTIONS USED:

ENGINE:

READ XML(CONSOLE FUNCTIONS)

1. bool readFile()

Reads the XML file and sets the field values for found controls. The controls which are not saved will be loaded with default settings.

WRITE XML (CONSOLE FUNCTIONS)

1. saveToXML(SimXMLDocument *document, const char *profileName)
   Writes the attributes of the control. If the control is a simGroup (consists of several child controls), every control will be saved.

SCRIPT:

```
%currentObject = %ctrl.getRootControl();
%saved = %currentObject.saveToXML( $appName, $appName @ ".xml");
if( %saved )
   echo (" Saved " @ $appName @ ".xml" );
else
   echo( "Saved failed" );

new ReadXML($appName) { fileName = "" @ $appName @ ".xml"; };
$appName.readFile();
```

Make sure the profileName and the ReadXMLObject name (**THE HIGHLIGHTED** ONES) are the same.

Window pop-out

Pop the windows out

Any GuiWindowCtrl can be popped. **Make sure you name the GuiWindowCtrl**.

CLASS USED:

ENGINE:

Defined in initPersistFields() of GuiWindowCtrl (GuiWindow.cpp)

addGroup("PopOut");

addField("ShowTitle") : Whether the Title Ctrl will show.

addField("AllowPopWindow") : Whether the window can be poped out.

addField("isInPopup") : If the windowctrl is being shown on own canvas.

addField("LastWindowPosition") : The last known position for the window.

FUNCTIONS USED:

ENGINE:

(CONSOLE FUNCTIONS)

1. void ClosePopOut()

   Puts the guiwindow back on the main canvas.

2. void OpenPopOut()

   Puts the guiwindow on a new canvas.

SCRIPT:

```
if( %ctrl.closePopOut )
   %ctrl->ClosePopOut();

if( %ctrl.openPopOut )
   %ctrl->OpenPopOut();
```

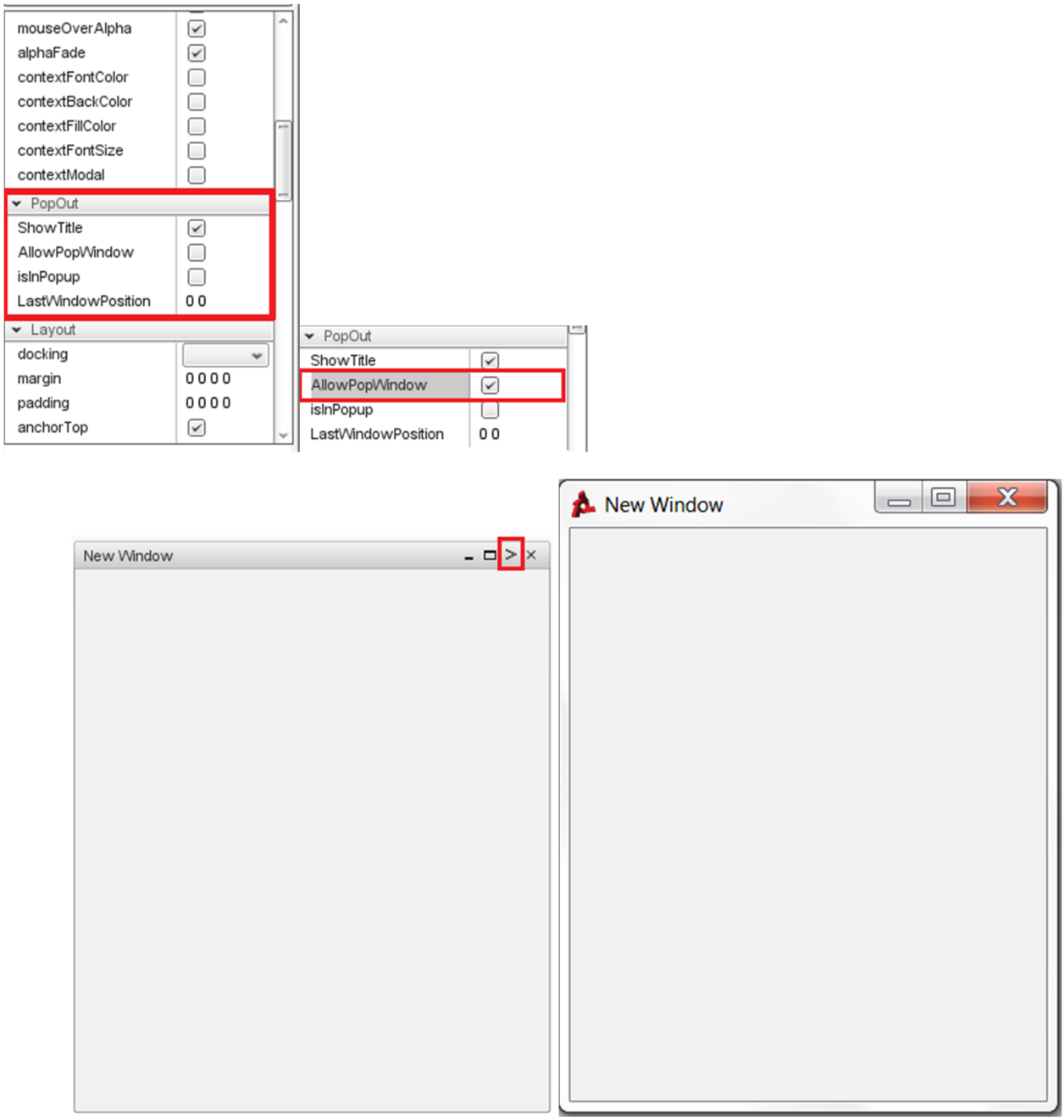


Fig: New popout window

SimObject class

CONSOLE FUNCTIONS

void setEditable(bool b)

DESCRIPTION: When true, an object can be editable in the game.

PARAMETERS:

- b: The boolean value to be set.

```
%ctrl.setEditable( true );
```

bool isEditable()

DESCRIPTION: Returns if an object is editable or not in the game.

PARAMETERS: NONE

```
%editable = %ctrl.isEditable();
```

bool saveToXML(const char *profileName, const char *fileName)

DESCRIPTION: Saves the object's properties/attributes to an XML file. Returns true if successful.

PARAMETERS:

- profileName: The name of the XMLObject for testing purposes.
- fileName: The name of the file to be saved in.

```
%currentObject = %ctrl.getRootControl();
%saved = %currentObject.saveToXML( $appName, $appName @ ".xml");
if( %saved )
   echo (" Saved " @ $appName @ ".xml" );
else
   echo( "Saved failed" );
```

GuiControl class

NON-CONSOLE FUNCTIONS

void updateRenderAlpha(bool mMouseOver)

DESCRIPTION: Updates the alpha value of a control depending upon the mouse-over value. If mouse-over, the **mouse-over alpha** is updated, otherwise the **normal alpha** is updated.

PARAMETERS:

- mMouseOver: If mouse-over, the value is true. If not, the value is false.

void applyProfileSettings()

DESCRIPTION: Applies the profile settings, i.e. font color, fill color, background color, etc. to a control.

PARAMETERS: NONE

void copyProfileSettings()

DESCRIPTION: Copy the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

void resetProfileSettings()

DESCRIPTION: Resets the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

void setProfileAlpha(F32 alphaValue)

DESCRIPTION: Sets the alpha value of a control and of its immediate children.

PARAMETERS:

- alphaValue: The alpha value to be set.

void fadeControl()

DESCRIPTION: Fades the control when mouse enter or leave a control.

PARAMETERS: NONE

void setControlColor(const char *colorName, ColorI color)

DESCRIPTION: Sets the value of the named **colorname**.

PARAMETERS:

- colorName: The name of the color.
- color: The color to be set.

void setMaxExtent(const Point2I &newMaxExtent)

DESCRIPTION: Sets the maximum extent of a control.

PARAMETERS:

- newMaxExtent: The value of the extent to be set.

CONSOLE FUNCTIONS

void setAlphaValue(F32 alphaValue):

DESCRIPTION: Sets the alpha value of the object to the specified value.

PARAMETERS:

- alphaValue: The alpha value to be set.

```
ctrl.setAlphaValue( %value );
```

F32 getAlphaValue( ):

DESCRIPTION: Returns the current alpha value of the object.

PARAMETERS: NONE

```
%this.oldAlphaValue = %ctrl.getAlphaValue();
```

void setMouseOverAlphaValue(F32 mouseOverAlpha):

DESCRIPTION: Sets the mouse-over alpha value of the object to the specified value.

PARAMETERS:

- mouseOverAlpha: The mouse-over alpha value to be set.

```
%this.ctrl.setMouseOverAlphaValue( %value );
```

F32 getMouseOverAlphaValue( ):

DESCRIPTION: Returns the current mouse-over alpha value of the object.

PARAMETERS: NONE

```
%this.oldMouseOverAlphaValue = %ctrl.getMouseOverAlphaValue();
```

void setAlphaFadeTime(F32 alphaFadeTime):

DESCRIPTION: Sets the alpha fade time of the object to the specified value.

PARAMETERS:

- alphaFadeTime: The alpha fade time to be set.

```
%this.ctrl.setAlphaFadeTime( %value );
```

F32 getAlphaFadeTime( ):

DESCRIPTION: Returns the current alpha fade time of the object.

PARAMETERS: NONE

```
%this.oldAlphaFadeTime = %ctrl.getAlphaFadeTime();
```

void setContextAlpha(bool alpha)

DESCRIPTION: If enabled, displays the option to change the alpha value of a control in the game.

PARAMETERS:

- alpha: The boolean value to set or reset the context alpha setting.

```
%ctrl.setContextAlpha( true );
```

bool isContextAlphaEnabled()

DESCRIPTION: Returns whether the option for changing the alpha of a control is enabled or not, during the game.

PARAMETERS: NONE

```
%alphaEnabled = %ctrl.isContextAlphaEnabled();
```

void setContextMouseOverAlpha(bool mouseOverAlpha)

DESCRIPTION: If enabled, displays the option to change the mouse-over alpha value of a control in the game.

PARAMETERS:

- mouseOverAlpha: The boolean value to set or reset the context mouse-over alpha setting.

```
%ctrl.setContextMouseOverAlpha( true );
```

bool isContextMouseOverAlphaEnabled()

DESCRIPTION: Returns whether the option for changing the mouse-over alpha of a control is enabled or not, during the game.

PARAMETERS: NONE

```
%mouseOverAlphaEnabled = %ctrl.isContextMouseOverAlphaEnabled();
```

void setContextAlphaFade(bool alphaFade)

DESCRIPTION: If enabled, displays the option to change the alpha fade time of a control in the game.

PARAMETERS:

- alphaFade: The boolean value to set or reset the context alpha fade setting.

```
%ctrl.setContextAlphaFade( true );
```

bool isContexAlphaFadeEnabled()

DESCRIPTION: Returns whether the option for changing the alpha fade time of a control is enabled or not, during the game.

PARAMETERS: NONE

```
%alphaFadeEnabled = %ctrl.isContextAlphaFadeEnabled();
```

void setContextFontColor(bool fontColor)

DESCRIPTION: If enabled, displays the option to change the font color of a control in the game.

PARAMETERS:

- fontColor: The boolean value to set or reset the context font color setting.

```
%ctrl.setContextFontColor( true );
```

bool canChangeContextFontColor()

DESCRIPTION: Returns whether the option for changing the font color of a control is enabled or not, during the game.

PARAMETERS: NONE

```
%fontColorChange = %ctrl.canChangeContextFontColor();
```

void setContextFillColor(bool fillColor)

DESCRIPTION: If enabled, displays the option to change the fill color of a control in the game.

PARAMETERS:

- fillColor: The boolean value to set or reset the context fill color setting.

```
%ctrl.setContextFillColor( true );
```

bool canChangeContextFillColor()

DESCRIPTION: Returns whether the option for changing the fill color of a control is enabled or not, during the game.

PARAMETERS: NONE

```
%fillColorChange = %ctrl.canChangeContextFillColor();
```

void setContextBackColor(bool backColor)

DESCRIPTION: If enabled, displays the option to change the background color of a control in the game.

PARAMETERS:

- backColor: The boolean value to set or reset the context background color setting.

```
%ctrl.setContextBackColor( true );
```

bool canChangeContextBackColor()

DESCRIPTION: Returns whether the option for changing the background color of a control is enabled or not, during the game.

PARAMETERS: NONE

```
%backgroundColorChange = %ctrl.canChangeContextBackColor();
```

void setShowContextWindowSettings(bool showSettings)

DESCRIPTION: If enabled, displays the option to change the window settings of a control in the game.

PARAMETERS:

- showSettings: The boolean value to set or reset the context window setting.

```
%ctrl.setShowContextWindowSettings( true );
```

bool canShowContextWindowSettings()

DESCRIPTION: Returns whether the option for changing the window settings of a control is enabled or not, during the game.

PARAMETERS: NONE

```
%windowSettingsShown = %ctrl.canShowContextWindowSettings();
```

void setContextFontSize(bool fontSize)

DESCRIPTION: If enabled, displays the option to change the font size of a control in the game.

PARAMETERS:

- fontSize: The boolean value to set or reset the context font size.

```
%ctrl.setContextFontSize( true );
```

bool canChangeContextFontSize()

DESCRIPTION: Returns whether the option for changing the font size of a control is enabled or not, during the game.

PARAMETERS: NONE

```
%changeFontSizeShown = %ctrl.canChangeContextFontSize();
```

void setContextLockControl(bool lock)

DESCRIPTION: If enabled, displays the option to love a control in the game. When unset, the control can be moved.

PARAMETERS:

- lock: The boolean value to set or reset the context lock setting.

```
%ctrl.setContextLockControl( true );
```

bool isContextLockable()

DESCRIPTION: Returns whether the option for locking a control is enabled or not, during the game.

PARAMETERS: NONE

```
%lockEnabled = %ctrl.isControlLockable();
```

void setContextMoveControl(bool moveControl)

DESCRIPTION: If enabled, displays the option to move a control in the game. When unset, the control is locked.

PARAMETERS:

- moveControl: The boolean value to set or reset the context move setting.

```
%ctrl.setContextMoveControl( true );
```

bool isContextMovable()

DESCRIPTION: Returns whether the option for moving a control is enabled or not, during the game.

PARAMETERS: NONE

```
%moveEnabled = %ctrl.isContextMovable();
```

void setControlLock(bool locked)

DESCRIPTION: When enabled, locks a control.

PARAMETERS:

- locked: The boolean value to lock a control.

```
%lockPos = %popup.appendItem("Locked" TAB "" TAB "%this.object.setControlLock(!%this.object.getControlLock());");
```

bool getControlLock()

DESCRIPTION: Returns whether the control is locked or not.

PARAMETERS: NONE

```
%popup.checkItem(%lockPos, %ctrl.getControlLock());
```

void setControlFontColor(ColorI color)

DESCRIPTION: Sets the font color of a control and of its immediate children to the **color** specified.

PARAMETERS:

- color: The color to be set.

```
%fontColorPos = %popup.appendItem("Font Color..." TAB "" TAB "getColorI(  %this.object.controlFontColor , %this.object
 @ \".setControlFontColor\", %this.object.getRoot(), %this.object @ \".setControlFontColor\", %this.object
@ \".setControlFontColor\" );");
```

void setControlFillColor(ColorI color)

DESCRIPTION: Sets the fill color of a control to the **color** specified. Will not work for transparent controls.

PARAMETERS:

- color: The color to be set.

```
%fillColorPos = %popup.appendItem("Fill Color..." TAB "" TAB "getColorI(  %this.object.controlFillColor ,
%this.object @ \".setControlFillColor\", %this.object.getRoot(), %this.object @ \".setControlFillColor\",
 %this.object @ \".setControlFillColor\" );");
```

void setControlBackgroundColor(ColorI color)

DESCRIPTION: Sets the background color of a control to the **color** specified. Will not work for transparent controls.

PARAMETERS:

- color: The color to be set.

```
%backgroundColorPos = %popup.appendItem("Background Color..." TAB "" TAB "getColorI(  %this.object.backgroundColor ,
%this.object @ \".setControlBackgroundColor\", %this.object.getRoot(), %this.object @ \".setControlBackgroundColor\",
 %this.object @ \".setControlBackgroundColor\" );");
```

void setControlFontSize(S32 fontSize)

DESCRIPTION: Sets the font size of the control.

PARAMETERS:

- fontSize: The font size of the control

```
%this.ctrl.setControlFontSize(%value);
```

S32 getControlFontSize()

DESCRIPTION: Returns the font size of the control.

PARAMETERS: NONE

```
%this.oldFontSize = %ctrl.getControlFontSize();
```

void setControlTexture(const char *fileName)

DESCRIPTION: Sets the texture of the control.

PARAMETERS:

- fileName: The filename of the texture.

```
%texPos = %popup.appendItem("Set Texture..."
      TAB "" TAB
      "getLoadFilename( \"All Image Files|*.png;*.jpeg;*.jpg;*.tga;*.jng;*.mng;*.gif;*.bmp;*.dds|png|*.png|jpeg
*.jpeg|jpg|*.jpg|tga|*.tga|jng|*.jng|mng|*.mng|gif|*.gif|bmp|*.bmp|dds|*.dds\", %this.object @ \".setControlTexture\",
\"\" );");
```

String getControlTextureFile()

DESCRIPTION: Returns the filename of the texture.

PARAMETERS: NONE

```
%textureFilename = %ctrl.getControlTextureFile();
```

bool controlIsSibling(GuiControl *sibling)

DESCRIPTION: Returns true if a control is a sibling of the given control or not.

PARAMETERS:

- sibling: The control to be tested.

```
%isSibling = %ctrl.controlIsSibling( %captured );
```

GuiControl *getRootControl()

DESCRIPTION: Returns the root control which is a child of the canvas.

PARAMETERS: NONE

```
%currentObject = %ctrl.getRootControl();
```

Point2I getMaxExtent()

DESCRIPTION: Returns the maximum extent of a control.

PARAMETERS: NONE

```
%maxExtent = %ctrl.getMaxExtent();
```

bool transparentControlCheck()

DESCRIPTION: Checks if the control is transparent or not. If yes, returns true.

PARAMETERS: NONE

```
%transparent = %ctrl.transparentControlCheck();
```

GuiSwatchButtonCtrl class

NON-CONSOLE FUNCTIONS

void applyProfileSettings()

DESCRIPTION: Applies the profile settings, i.e. font color, fill color, background color, etc. to a control.

PARAMETERS: NONE

void copyProfileSettings()

DESCRIPTION: Copy the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

void resetProfileSettings()

DESCRIPTION: Resets the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

GuiColorPickerCtrl class

NON-CONSOLE FUNCTIONS

void applyProfileSettings()

DESCRIPTION: Applies the profile settings, i.e. font color, fill color, background color, etc. to a control.

PARAMETERS: NONE

void copyProfileSettings()

DESCRIPTION: Copy the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

void resetProfileSettings()

DESCRIPTION: Resets the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

GuiButtonBaseCtrl class

NON-CONSOLE FUNCTIONS

bool setProtectedTexture(void *object, const char* index, const char* data)

DESCRIPTION: Sets the texture of the control. It relates to the texture function of the **GuiControl** class**.**

PARAMETERS:

- object: The object whose texture is to be changed.
- index: Index, if array.
- data: The data (filename) of the texture.

const char* getProtectedTexture(void* object, const char* data)

DESCRIPTION: Returns the data (filename) of the texture.

PARAMETERS:

- object: The object whose texture file is to be returned.
- data: The data (filename) of the texture.

GuiMessageVectorCtrl class

NON-CONSOLE FUNCTIONS

void applyProfileSettings()

DESCRIPTION: Applies the profile settings, i.e. font color, fill color, background color, etc. to a control.

PARAMETERS: NONE

void copyProfileSettings()

DESCRIPTION: Copy the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

void resetProfileSettings()

DESCRIPTION: Resets the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

GuiClockHud class

NON-CONSOLE FUNCTIONS

void applyProfileSettings()

DESCRIPTION: Applies the profile settings, i.e. font color, fill color, background color, etc. to a control.

PARAMETERS: NONE

void copyProfileSettings()

DESCRIPTION: Copy the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

void resetProfileSettings()

DESCRIPTION: Resets the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

GuiCrossHairHud class

NON-CONSOLE FUNCTIONS

void applyProfileSettings()

DESCRIPTION: Applies the profile settings, i.e. font color, fill color, background color, etc. to a control.

PARAMETERS: NONE

void copyProfileSettings()

DESCRIPTION: Copy the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

void resetProfileSettings()

DESCRIPTION: Resets the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

GuiHealthBarHud class

NON-CONSOLE FUNCTIONS

void applyProfileSettings()

DESCRIPTION: Applies the profile settings, i.e. font color, fill color, background color, etc. to a control.

PARAMETERS: NONE

void copyProfileSettings()

DESCRIPTION: Copy the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

void resetProfileSettings()

DESCRIPTION: Resets the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

GuiHealthTextHud class

NON-CONSOLE FUNCTIONS

void applyProfileSettings()

DESCRIPTION: Applies the profile settings, i.e. font color, fill color, background color, etc. to a control.

PARAMETERS: NONE

void copyProfileSettings()

DESCRIPTION: Copy the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

void resetProfileSettings()

DESCRIPTION: Resets the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

GuiShapeNameHud class

NON-CONSOLE FUNCTIONS

void applyProfileSettings()

DESCRIPTION: Applies the profile settings, i.e. font color, fill color, background color, etc. to a control.

PARAMETERS: NONE

void copyProfileSettings()

DESCRIPTION: Copy the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

void resetProfileSettings()

DESCRIPTION: Resets the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

GuiSpeedometerHud class

NON-CONSOLE FUNCTIONS

void applyProfileSettings()

DESCRIPTION: Applies the profile settings, i.e. font color, fill color, background color, etc. to a control.

PARAMETERS: NONE

void copyProfileSettings()

DESCRIPTION: Copy the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

void resetProfileSettings()

DESCRIPTION: Resets the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

GuiGradientCtrl class

NON-CONSOLE FUNCTIONS

void applyProfileSettings()

DESCRIPTION: Applies the profile settings, i.e. font color, fill color, background color, etc. to a control.

PARAMETERS: NONE

void copyProfileSettings()

DESCRIPTION: Copy the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

void resetProfileSettings()

DESCRIPTION: Resets the profile settings of a control like font color, fill color, background color, etc.

PARAMETERS: NONE

GuiCanvas class

NON - CONSOLE FUNCTIONS

class GuiCanvas_Close : public SimEvent

DESCRIPTION: Deletes the window control and posts an event.

PARAMETERS:

- None.

void onWindowClose()

DESCRIPTION: Sends out a callback when the window is closed.

PARAMETERS:

- None.

CONSOLE FUNCTIONS

void onResize(const char* width, const char* height)

DESCRIPTION: Callback to the script when the control is resize. Replaced the old executef callback.

PARAMETERS:

- width: Width of the new control.
- height: Height of the new control.

```
function NewControl::onResize( %this, %width, %height)
{
   // Do something
}
```

void onCreateMenu()

DESCRIPTION: Callback to the script when a menu bar is created. Replaced the old executef callback.

PARAMETERS:

- None.

```
function NewControl::onCreateMenu( %this )
{
   // Do something
}
```

void onDestroyMenu()

DESCRIPTION: Callback to the script when a menu bar is destroyed. Replaced the old executef callback.

PARAMETERS:

- None.

```
function NewControl::onDestroyMenu( %this )
{
   // Do something
}
```

void onLoseFocus()

DESCRIPTION: Callback to the script when a control loses focus. Replaced the old executef callback.

PARAMETERS:

- None.

```
function NewControl::onLoseFocus( %this )
{
   // Do something
}
```

void onGainFocus()

DESCRIPTION: Callback to the script when a control gains focus. Replaced the old executef callback.

PARAMETERS:

- None.

```
function NewControl::onGainFocus( %this )
{
   // Do something
}
```

void onWindowClose()

DESCRIPTION: Callback to the script when a window is closed. Replaced the old executef callback.

PARAMETERS:

- None.

```
function NewControl::onWindowClose( %this )
{
   // Do something
}
```

void setPopupShown(bool shown)

DESCRIPTION: Enabled when a popup is shown in the canvas.

PARAMETERS:

- shown: True if the popup is shown.

```
Canvas.setPopupShown( true );
```

bool isPopupShown()

DESCRIPTION: Returns whether the popup is shown or not.

PARAMETERS: NONE

```
%popup = Canvas.isPopupShown();
```

GuiWindowCtrl class

NON - CONSOLE FUNTIONS

void setVisible(bool value)

DESCRIPTION: If enabled, shows the window on the canvas.

PARAMETERS:

- value: The boolean value to set or reset the visibility of the window.

void OnWindowPopOut()

DESCRIPTION: Pops out a new window.

PARAMETERS:

- None

void updateRendering()

DESCRIPTION: Updates the window after popout.

PARAMETERS:

- None

void PopUpClosed()

DESCRIPTION: Closes the popup and replaces the window onto the game canvas.

PARAMETERS:

- None

void ClosePopOut()

DESCRIPTION: Closes the pop-out. Calls the PopUpClosed.

PARAMETERS:

- None

CONSOLE FUNTIONS

void setContextTitle(bool value)

DESCRIPTION: If enabled, displays the option to set the title of a window control in the game.

PARAMETERS:

- value: The boolean value to set or reset the context title setting.

```
%ctrl.setContextTitle( true );
```

bool isTitleSet()

DESCRIPTION: Returns true if the title of a window control can be changed in the game.

PARAMETERS: NONE

```
%titleEnabled = %ctrl.isTitleSet();
```

void setWindowTitle(const char *title)

DESCRIPTION: Sets the window title of a window control

PARAMETERS:

- title: The title to be set.

```
%this.ctrl.setWindowTitle(%value);
```

const char *getWindowTitle()

DESCRIPTION: Returns the window title of a window control.

PARAMETERS: NONE

```
%this.oldTitleName = %ctrl.getWindowTitle();
```

void ClosePopOut()

DESCRIPTION: Puts the GuiWindowCtrl on the main canvas.

PARAMETERS: NONE

```
if( %ctrl.closePopOut )
   %ctrl->ClosePopOut();
```

void OpenPopOut ()

DESCRIPTION: Puts the GuiWindowCtrl on a new canvas.

PARAMETERS: NONE

```
if( %ctrl.openPopOut )
   %ctrl->OpenPopOut();
```

ReadXML class

NON-CONSOLE FUNCTIONS

bool readLayer(SimXMLDocument *document)

DESCRIPTION: Parses the XML file layer by layer for Group and Settings.

PARAMETERS:

- document: The document to be parsed.

CONSOLE FUNCTIONS

bool readFile()

DESCRIPTION: Parses the XML file to read the settings of a control. Returns true, if parsed correctly.

PARAMETERS: NONE

```
new ReadXML($appName) { fileName = "" @ $appName @ ".xml"; };
$appName.readFile();
```

SimFieldDictionary class

Entry *findField(StringTableEntry fieldname)

DESCRIPTION: Finds the dynamic field with the fieldname **FieldName**. Returns the field if found, else NULL.

PARAMETERS:

- fieldName: The fieldname to be searched.

Console Edits

TOOLS/INGAMEEDITOR

main.cs

initializeInGameEditor()

DESCRIPTION: Initializes the in-game editor and loads all the necessary scripts and gui's.

destroyInGameEditor()

DESCRIPTION: Destroys the in-game editor.

changeFontSize.ed.gui

setTitle.ed.gui

windowSettings.ed.gui

changeFontSize.cs

loadDialog(%this, %ctrl)

DESCRIPTION: Loads the font size gui.

PARAMETERS:

- %this: The changeFontSize object.
- %ctrl: The control which is passed for font size change.

resetValue(%this)

DESCRIPTION: Resets the value of the control's font size.

PARAMETERS:

- %this: The changeFontSize object.

setValue(%this, %value)

DESCRIPTION: Sets the value (font size) of the control.

PARAMETERS:

- %this: The changeFontSize object.
- %value: The value to be set.

setTitle.cs

loadDialog(%this, %ctrl)

DESCRIPTION: Loads the set title gui.

PARAMETERS:

- %this: The setTitle object.
- %ctrl: The control which is passed for title change.

resetValue(%this)

DESCRIPTION: Resets the value of the control's title.

PARAMETERS:

- %this: The setTitle object.

setValue(%this, %value)

DESCRIPTION: Sets the value (title) of the control.

PARAMETERS:

- %this: The setTitle object.
- %value: The value to be set.

windowSettings.cs

loadDialog(%this, %ctrl)

DESCRIPTION: Loads the window settings gui.

PARAMETERS:

- %this: The windowSettings object.
- %ctrl: The control which is passed for window settings change.

resetValue(%this)

DESCRIPTION: Resets the controls' window settings.

PARAMETERS:

- %this: The windowSettings object.

setAlphaValue(%this, %value)

DESCRIPTION: Sets the value (alpha value) of the control.

PARAMETERS:

- %this: The windowSettings object.
- %value: The value to be set.

setMouseOverAlphaValue(%this, %value)

DESCRIPTION: Sets the value (mouse-over alpha value) of the control.

PARAMETERS:

- %this: The windowSettings object.
- %value: The value to be set.

setAlphaFadeTime(%this, %value)

DESCRIPTION: Sets the value (alpha fade time) of the control.

PARAMETERS:

- %this: The windowSettings object.
- %value: The value to be set.

checkAllowedDialogs(%this)

DESCRIPTION: Checks which settings are enabled for the current control

PARAMETERS:

- %this: The windowSettings object.

ingameContext.cs

onRightMouseUp (%this, %ctrl)

DESCRIPTION: Executes the popup menu when right clicked.

PARAMETERS:

- %this: The ingameContext object.
- %ctrl: The control on which the right click context menu is to be popped.

xmlOutput (%this, %ctrl)

DESCRIPTION: Outputs the control information into the XML output.

PARAMETERS:

- %this: The ingameContext object.
- %ctrl: The control whose information is to be stored.

onAdd (%this)

DESCRIPTION: Performs the function when added to the system, which loads the XML setting if present.

PARAMETERS:

- %this: The ingameContext object.

transparentControlCheck (%this, %ctrl, %colorName)

DESCRIPTION: Checks if the control is transparent/textureless or not.

PARAMETERS:

- %this: The ingameContext object.
- %ctrl: The control whose transparency is to be checked.
- %colorName: If not transparent, changes the specified color named **colorName**.

game/core/art/gui/console.gui

onLoseFocus ()

DESCRIPTION: Pops the console when the focus is lost.

PARAMETERS:

- None.

onGainFoucs ()

DESCRIPTION: Pushes the console window again to the canvas.

PARAMETERS:

- None.

onResize (%this, %width, %height)

DESCRIPTION: Resizes the console window.

PARAMETERS:

- %this: The consoleDlg object.
- %width: The width of the control.
- %height: The height of the control.