MouseEventManager set up.

for version 1.0.0

This is gives you a cleaner way of setting up button events, you no longer have to add specific event listeners to buttons, doesn't matter if they buttons are on the stage or not when the manager is initialized. No longer have to add or remove events, if you don't want the mouse event to fire on a button or display object just remove it from stage.

IMPORTS:

In the fla you only need to import 1 class, nothing gets added to the library.

```
import com.blt.managers.MouseEventManager;
```

PUBLIC PROPERTIES:

debug:Boolean = false;
Set this to get traces, it will tell you what movie clip you rolled over, rolled out, and clicked on. You can leave it true when it goes live.

PUBLIC METHODS:

var mouseManager:MouseEventManager = new MouseEventManager(_target:DisplayObject):void
When you initialize the constructor you need to pass the display object that you want to track mouse on. This object should be a movie clip, just pass the path to the actual object using it's instance name. I usually just use the stage, this way it will track all mouse events for any visible object on the stage. You can nest it also, make different instances of it for different types of navigation, if you have a lot it may be a good idea to handle it that way so you get less and cleaner code.

mouseManager.addButton(_btn:String, _click:Function = null, _over:Function = null, _out:Function = null):void
Adds a single button instance name, along with methods to call based on mouse event activity for that button.

- **_btn:** The instance name of the movie clip / button.
- **_click:** The on click function, pass this without the "()", this will get called when that button is clicked. It will also return the button movie clip or button object that was clicked on.
- **_over:** The on over function, pass this without the "()", this will get called when that button is rolled over. It will also return the button movie clip or button object that was clicked on.
- **_out:** The on out function, pass this without the "()", this will get called when that button is rolled out. It will also return the button movie clip or button object that was clicked on.

mouseManager.addButtons(_btns:Array, _click:Function = null, _over:Function = null, _out:Function = null):void
Adds a list of button instance names, along with methods to call based on mouse event activity for any of those buttons.

- **_btns:** An array of strings which is the instance name of each of the movie clips / buttons that are being added.
- **_click:** The on click function, pass this without the "()", this will get called when that button is clicked. It will also return the button movie clip or button object that was clicked on.
- **_over:** The on over function, pass this without the "()", this will get called when that button is rolled over. It will also return the button movie clip or button object that was clicked on.
- **_out:** The on out function, pass this without the "()", this will get called when that button is rolled out. It will also return the button movie clip or button object that was clicked on.

mouseManager.addNullButton(_click:Function = null, _over:Function = null, _out:Function = null):void
This adds a button methods that only get fired when no methods are set for whatever the mouse is over or clicking on.

- **_click:** The on click function, pass this without the "()", this will get called when that button is clicked. It will also return the button movie clip or button object that was clicked on.
- **_over:** The on over function, pass this without the "()", this will get called when that button is rolled over. It will also return the button movie clip or button object that was clicked on.
- **_out:** The on out function, pass this without the "()", this will get called when that button is rolled out. It will also return the button movie clip or button object that was clicked on.

mouseManager.destroy():void
This method will remove listeners and null our references to mouse event display object. This is fired when the display object that is passed through the constructor is removed from the stage, it can also be called from the fla.

SET UP:

// this is code from the true detective expand banner, has navigation buttons, video selector buttons, and social buttons. One of the more complex navigations we have done on a banner, i figured this was a good way to show how this works.

```
// add import
import com.blt.managers.MouseEventManager;

// initialize the constructor
var mouseManager:MouseEventManager = new MouseEventManager( this );

// shows traces for mouse events
mouseManager.debug = true;

// navigation buttons
mouseManager.addButton( "moreBtn_mc",  moreBtnClicked, selectorBtnOver, selectorBtnOut );
mouseManager.addButtons( ["aboutBtn_mc", "videoBtn_mc"], setContent, selectorBtnOver, selectorBtnOut );

// video buttons
mouseManager.addButtons( ["video1Btn_mc", "video2Btn_mc", "video3Btn_mc", "video4Btn_mc"], setVideo, videoBtnOver, videoBtnOut );

// replay button
mouseManager.addButton( "replayVideoBtn_mc", replayWSound );

// replay button
mouseManager.addButton( "closeBtn_mc", closePanel );

// social buttons
mouseManager.addButtons( ["facebookBtn_mc", "twitterBtn_mc"], handleSocialClick );

// main click through
mouseManager.addButton( "bg_mc",  generalClickThrough );

// main click through
mouseManager.addNullButton( generalClickThrough );


//######################################## MAIN CLICK BUTTON METHOD ########################################
```

```
function generalClickThrough( btn:MovieClip = null ):void
{
	trace( "MAIN CLICK FIRED" );
}


//######################################## SELECTOR BUTTON METHOD ########################################
function handleSocialClick( btn:MovieClip = null ):void
{
	switch ( btn.name )
	{
		case "facebookBtn_mc":
			trace( "FACEBOOK BUTTON CLICKED" );
			break;

		case "twitterBtn_mc":
			trace( "TWITTER BUTTON CLICKED" );
			break;
	}
}


//######################################## CLOSE PANEL BUTTON METHOD ########################################
function closePanel( btn:MovieClip = null ):void
{
	trace( "CLOSE CALLED" );
}


//######################################## SELECTOR BUTTON METHODS ########################################
function moreBtnClicked( btn:MovieClip ):void
{
	trace( "MORE BUTTON CLICKED" );
}


function selectorBtnOver( btn:MovieClip ):void
{
	TweenLite.to ( btn.bg_mc, .4, { alpha:1 });
}


function selectorBtnOut( btn:MovieClip ):void
{
	if ( btn == currentContentBtn ) return;
	TweenLite.to ( btn.bg_mc, .4, { alpha:.5 });
}


function setContent( btn:MovieClip ):void
{
	if ( btn.name == "videoBtn_mc" )
	{
		updateVideoThumbs();
	}
}


//######################################## VIDEO BUTTON METHODS ########################################
function videoBtnOver( btn:MovieClip ):void
{
	TweenLite.to ( btn.bg_mc, .4, { alpha:1 });
}


function videoBtnOut( btn:MovieClip ):void
{
	TweenLite.to ( btn.bg_mc, .4, { alpha:0 });
}


function setVideo( btn:MovieClip ):void
{

}
```