Javascript THM Template
HOW TO

Constructor Call:

To get everything working correctly javascript needs all the scripts imported up front. All of the library javascript files are concatenate into one large file for ease of uploading and maintaining. This script file should be included in the <head> tags of the html.

```
<script type="text/javascript" src="./library/THP_Library.js"></script>
```

Once this scripts have been added the plugin needs to be embedded in the body:

```
<object id="myplugin" type="application/x-monoclegl" width="480" height="320"></object>
```

Then the plugin needs to be referenced while creating the template object or any other plugin object. This can be done by:

```
var p = document.getElementById("myplugin");
var thmDemo = new THP_Template(p, 545, 371, 3);
```

When creating a new demo you must tell the plugin the size of the demo which in this case is 545 by 371. Then the final parameter is how many questions are in the demo. This will return a instance of the plugin which is stored in thmDemo.

Media URL:

In every demo we declare the mediaURL to point to the directory where all the media is located. The purpose is to able to change the location of all the media with just a single line of code. For instance while developing it's note required that you have a server to host media. Instead set the mediaURL to the local location of the media on the hard drive. Then when the demo is submitted to Top Hat Monocle we can easily change the mediaURL to point to our server.

For example, if you wanted to load all the images off you hard drive:

```
var mediaURL = "file:///c:/path/to/"
var awesome = new Sprite(p, mediaURL + "awesome.png", 40, 140, 375, 120);
```

Scene Array:

Each question is stored in the sceneArray. They are stored in the order they are displayed (ie. 0 is the first question and then they continue in sequence. Each question has six callback function which are used to hook into the template:

initQuiz() is called once at the very start of the demo. This is the ideal place to create objects and add children to the quizzes sprite.
loadQuiz() is called when the instruction button is pressed or a new question is selected. Typically this function is display information and adding event listeners.

showCorrectAnswer() is called after 3 wrong attempts or 1 right answer. Typically this function is for answer animations.
resetQuiz() is ran whenever the user presses the reset button. This function is obsolete and loadQuiz() is called if resetQuiz doesn't exist.
cleanUp() quiz is ran right before a loadQuiz functions is called. Typically this function removes any event listeners or freeing memory.
checkAnswer() is called whenever the submit button is press. Typically this function is for logic tests to return true if correct or false otherwise.

For each step a quiz class is created which holds all the callback function, boolean flags and display objects. The quiz array can be access from thmDemo.sceneArray but to figure out exactly what number the quiz you want to a handy little function called thmDemo.getScene has been added. At the beginning of a question it's good practice to call thmDemo.getScene to get the exact quiz you want in case steps or questions numbers change.

In the following example the program saves the internal quiz number for question #2. Once you have this number you can access and shortcut the scene. Anything added to this shortcuts will only show up while question #2 is being shown. This saves the programmer from control visibility flags.

```
// Get a shortcut to the quiz scene
var scene = thmDemo.getScene(1, 0);
```

It's possible to automatically go to any quiz with the setScene() command. It's meant to help debugging later questions without going through the whole quiz. None of the flags are changed and it's up to the programmer to get rid this setScene() command before the demo is considered done. NOTE: getScene() is slightly different from it's flash counterpart getQuiz(). As in the example below if want to jump to scene 3 you must subtract one from each.

```
// Jump to question 3
thmDemo.gotoScene(2, 0);
```

To add objects to a quiz each scene has it's own layer .bgLayer. All children should be added to this layer and when the template loads that scene the layer is placed in the correct order of the display list. (ie. Behind the right and bottom panels.) Currently labels can only be added to scene or they won't appear.

```
// Add someSprite to the bgLayer
scene.bgLayer.addChild(someSprite);

// Add someLabel directly to the scene
scene.addChild(someLabel);
```

Demo Labels:

All the demo label can be set inside the html file:

```
thmDemo.setInstructionText();
```

thmDemo.setTitle();

Each question need a question label created for it.

The Curtain:

The curtain is used to signal the student that an animation is happening and they can't interact with the demo. The curtain itself is a 50% alpha black layer that blocks the mouse from the sending events to objects beneath it.

// when animating use curtain with
thmDemo.showCurtain();

// when animating use curtain with
thmDemo.hideCurtain();

Sliding Panels:

A locking mechanism has been added. The panels can now be locked and unlocked by the bottomPanelLock and rightPanelLock flags. locked = true and the panels will not respond to an extend or retract commands:

To extend right and bottom panel and lock them example:

```
thmDemo.rightPanelExtend();
thmDemo.rightPanelLock = true;
thmDemo.bottomPanelExtend();
thmDemo.bottomPanelLock = true;
```

Remember to unlock the panels afterwards with:

```
thmDemo.rightPanelLock = false;
thmDemo.bottomPanelLock = false;
```

Explore Mode:

This mode is to give students a chance to explore more complicated demos without having to submit anything. For explore to run at the beginning of the demo the Boolean flag thmDemo.boolExplore must be set to true. Otherwise the demo will skip explore mode and jump to question one. Similar to how each quiz has it's own explore mode scene can be referenced by thmDemo.scnExplore. Explore mode has typically needs three callback hooks for initialization, loading and clean up:

```
thmDemo.scnExplore.initQuiz();
thmDemo.scnExplore.loadQuiz();
thmDemo.scnExplore.cleanUp();
```

To add objects to explore mode use the layer thmDemo.scnExplore.bgLayer. Object can be added directly to thmDemo.scnExplore but they will not be displayed in the correct order and will be drawn on top the panels.

```
// Add someSprite to the explore layer
thmDemo.scnExplore.bgLayer.addChild(someSprite);

// Add someLabel directly to the explore scene
thmDemo.scnExplore.(someLabel);
```