# Changes to JQuery Plugins

## ContentFlow – contentflow\_src.js

1. Line 676 add call to init() after constructor of ContentFlow object.

This means that the init runs after the ContentFlow object has been created rather on the window load. This is required as need creation of the content flow object to happen after the coverflowFile has been loaded.

var ContentFlow = function (container, config) {

//alert(" contentflow\_src.js ContentFlow create ");

if (container) {

ContentFlowGlobal.Flows.push(this);

this.Container = container;

this.\_userConf = config?config:{};

this.conf = {};

this.\_loadedAddOns = new Array();

//gh have to add an additional call to init as it is not being initialized on page load

// alert(" contentflow\_src.js ContentFlow create have to add init");

**this.init();**

} else {

//alert("contentflow\_src.js creating content flow object error ");

throw ('ContentFlow ERROR: No flow container node or id given');

}

};

2. Line 26 : Added a check for iPad. This will then allow for registration of touch events as Browser will be treated as of type ‘iPhone’.

Browser: new (function () {

this.Opera = window.opera ? true : false;

this.IE = document.all && !this.Opera ? true : false;

this.IE6 = this.IE && typeof(window.XMLHttpRequest) == "undefined" ? true : false;

this.IE8 = this.IE && typeof(document.querySelectorAll) != "undefined" ? true : false;

this.IE7 = this.IE && ! this.IE6 && !this.IE8 ? true : false;

this.WebKit = /WebKit/i.test(navigator.userAgent) ? true : false,

this.iPhone = /iPhone|iPod|iPad/i.test(navigator.userAgent)? true : false;

this.Chrome = /Chrome/i.test(navigator.userAgent) ? true : false;

this.Safari = /Safari/i.test(navigator.userAgent) && !this.Chrome ? true : false;

this.Konqueror = navigator.vendor == "KDE" ? true : false;

this.Konqueror4 = this.Konqueror && /native code/.test(document.getElementsByClassName) ? true : false;

this.Gecko = !this.WebKit && navigator.product == "Gecko" ? true : false;

this.Gecko19 = this.Gecko && Array.reduce ? true : false;

})(),

3. To remove irritating and non-intuitive behavior of mouse over images:

Line 308: change the removeEvent from the window object, to the object that originally registered the event.

this.stopDrag = function(event) {

//alert("contentflow\_src stop drag start");

if (!event) var event = window.event;

if (this.Browser.iPhone) {

window.removeEvent('touchmove', onDrag, false);

if (!this.ontouchmove) {

var t = event.target;

if (t.firstChild) t = t.firstChild;

var e = document.createEvent('MouseEvents');

e.initEvent('click', true, true);

t.dispatchEvent(e);

}

}

else {

//window.removeEvent('mousemove', onDrag, false);

this.removeEvent('mousemove', onDrag, false);

}

afterDrag(event);

}.bind(this);

Line 340: On startDrag - Change the event to be over only the selected object rather than whole window. Change window.addEvent(..) to this.addEvent(..)

this.startDrag = function (event) {

if (!event) var event = this.event;

var stopDrag = this.stopDrag;

if (this.Browser.iPhone) {

var s = this;

s.ontouchmove = false

window.addEvent('touchmove', function (e) {

s.ontouchmove = true;

onDrag(e);

}, false);

event.preventDefault();

window.addEvent('touchend', stopDrag, false);

}

else {

//window.addEvent('mousemove', onDrag, false);

//window.addEvent('mouseup', stopDrag, false);

this.addEvent('mousemove', onDrag, false);

this.addEvent('mouseup', stopDrag, false);

}

if(event.preventDefault) { event.preventDefault() }

}.bind(this);

## contentflow.css

Make the ‘Slider’ size larger to allow for easier dragging (as at this point only the slider is draggable)

Change from:

.ContentFlow .scrollbar .slider{width:16px;height:16px;

to

.ContentFlow .scrollbar .slider{width:22px;height:22px;

Line 297 (This looks to be simple error)

Change the ontochmove to ontouchmove.

if (!event) var event = window.event;

if (this.Browser.iPhone) {

window.removeEvent('touchemove', onDrag, false);

// if (!this.ontochmove) {

if (!this.ontouchmove) {

var t = event.target;

if (t.firstChild) t = t.firstChild;

var e = document.createEvent('MouseEvents');

e.initEvent('click', true, true);

t.dispatchEvent(e);

}

}

## Timeline : Timeline for bundled scripts (i.e. all javascripts are local rather than being pulled from Simile site)

## Timeline-bundle.js (under js/timeline)

1. Line 37, registered the Touch events if this is an iPad

SimileAjax.DOM.registerEventWithObject(this.\_div, "mousedown", this, "\_onMouseDown");

var isiPad = navigator.userAgent.match(/iPad/i) != null;

if (isiPad)

{

//console.log('simile band.js: registering touchstart : ');

SimileAjax.DOM.registerEventWithObject(this.\_div, "touchstart", this, "\_onMouseDown");

SimileAjax.DOM.registerEventWithObject(this.\_div, "touchmove", this, "\_onMouseMove");

}

SimileAjax.DOM.registerEventWithObject(this.\_div, "mousemove", this, "\_onMouseMove");

SimileAjax.DOM.registerEventWithObject(this.\_div, "mouseup", this, "\_onMouseUp");

SimileAjax.DOM.registerEventWithObject(this.\_div, "mouseout", this, "\_onMouseOut");

SimileAjax.DOM.registerEventWithObject(this.\_div, "dblclick", this, "\_onDblClick");

2. Line 229, Make changes to MouseMove event to allow for TouchMove event with iPad.

Timeline.\_Band.prototype.\_onMouseMove=function(D,A,E){

if(this.\_dragging){

var isiPad = navigator.userAgent.match(/iPad/i) != null;

if (isiPad)

{

A.preventDefault();

}

var C=A.clientX-this.\_dragX;

var B=A.clientY-this.\_dragY;

if (isiPad)

{

//console.log('simile band.js: \_onTouchMove : diff x ' + diffX + ' y ' + diffY);

if(A.touches.length == 1)

{ // Only deal with one finger

//console.log('simile band.js: \_onTouchMove : touches found ');

var touch = A.touches[0]; // Get the information for finger #1

var node = touch.target; // Find the node the drag started from

node.style.position = "absolute";

node.style.left = touch.pageX + "px";

node.style.top = touch.pageY + "px";

//console.log('simile band.js: \_onTouchMove : touches found x ' + touch.pageX + ' y ' + touch.pageY );

C = touch.pageX - this.\_dragX;

B = touch.pageY - this.\_dragY;

this.\_dragX = touch.pageX;

this.\_dragY = touch.pageY;

}

}

else

{

this.\_dragX=A.clientX;

this.\_dragY=A.clientY;

}

this.\_moveEther(this.\_timeline.isHorizontal()?C:B);

this.\_positionHighlight();

}};

## Timeline : Alternate Timeline for scripts downloaded dynamically at runtime from Simile Web site

## Band.js (under js/timeline/scripts)

1. Line 93, registered the Touch events if this is an iPad

SimileAjax.DOM.registerEventWithObject(this.\_div, "mousedown", this, "\_onMouseDown");

var isiPad = navigator.userAgent.match(/iPad/i) != null;

if (isiPad)

{

//console.log('simile band.js: registering touchstart : ');

SimileAjax.DOM.registerEventWithObject(this.\_div, "touchstart", this, "\_onMouseDown");

SimileAjax.DOM.registerEventWithObject(this.\_div, "touchmove", this, "\_onMouseMove");

}

SimileAjax.DOM.registerEventWithObject(this.\_div, "mousemove", this, "\_onMouseMove");

SimileAjax.DOM.registerEventWithObject(this.\_div, "mouseup", this, "\_onMouseUp");

SimileAjax.DOM.registerEventWithObject(this.\_div, "mouseout", this, "\_onMouseOut");

SimileAjax.DOM.registerEventWithObject(this.\_div, "dblclick", this, "\_onDblClick");

2. Line 468, Make changes to MouseMove event to allow for TouchMove event with iPad.

Timeline.\_Band.prototype.\_onMouseMove = function(innerFrame, evt, target) {

if (this.\_dragging) {

var isiPad = navigator.userAgent.match(/iPad/i) != null;

if (isiPad)

{

evt.preventDefault();

}

var diffX = evt.clientX - this.\_dragX;

var diffY = evt.clientY - this.\_dragY;

if (isiPad)

{

//console.log('simile band.js: \_onTouchMove : diff x ' + diffX + ' y ' + diffY);

if(evt.touches.length == 1){ // Only deal with one finger

//console.log('simile band.js: \_onTouchMove : touches found ');

var touch = evt.touches[0]; // Get the information for finger #1

var node = touch.target; // Find the node the drag started from

node.style.position = "absolute";

node.style.left = touch.pageX + "px";

node.style.top = touch.pageY + "px";

//console.log('simile band.js: \_onTouchMove : touches found x ' + touch.pageX + ' y ' + touch.pageY );

diffX = touch.pageX - this.\_dragX;

diffY = touch.pageY - this.\_dragY;

this.\_dragX = touch.pageX;

this.\_dragY = touch.pageY;

}

}

else

{

this.\_dragX = evt.clientX;

this.\_dragY = evt.clientY;

}

this.\_moveEther(this.\_timeline.isHorizontal() ? diffX : diffY);

this.\_positionHighlight();

}

};

## RTE Editor – jquery.rte.js

1. Line 259 and 286 change the <li></li> to <span></span> as <li> breaks the tab functionality.

for (var key in controls){

if(controls[key].separator) {

//li= $("<li></li>").append(obj.addClass(key));

li = $("<span></span>").addClass('separator');

} else {

if(controls[key].init) {

try {

controls[key].init.apply(controls[key], [this]);

} catch(e) {

}

}

if(controls[key].select) {

obj = $(controls[key].select)

.change( function(e) {

self.event = e;

self.toolbar\_click(this, controls[this.className]);

return false;

});

} else {

obj = $("<a href='#'></a>")

.attr('title', (controls[key].hint) ? controls[key].hint : key)

.attr('rel', key)

.click( function(e) {

self.event = e;

self.toolbar\_click(this, controls[this.rel]);

return false;

})

}

li = $("<span></span>").append(obj.addClass(key));

//li= $("<li></li>").append(obj.addClass(key));

}

2. Change all references to li in the css file to span (see jquery.rte.css)