```
import fl.controls.Slider;
import fl.events.SliderEvent;

// Set up specific heat capacity values
var cpWater:Number = 4.18;
var cpMetal:Number = 0.385;
// Set ranges for water and metal values
var minWaterTemp:Number = 15;
var maxWaterTemp:Number = 45;
var minWaterAmt:Number = 65;
var maxWaterAmt:Number = 150;
var minMetalTemp:Number = 90;
var maxMetalTemp:Number = 200;
var minMetalAmt:Number = 10;
var maxMetalAmt:Number = 45;

// Initial Variables
var initWaterTemp:Number;
var initWaterAmt:Number;
var initMetalTemp:Number;
var initMetalAmt:Number;
var finalTemp:Number;
var tempTimer:Timer;

// Set up sliders
sliderWaterTemp.minimum = minWaterTemp;
sliderWaterTemp.maximum = maxWaterTemp;
sliderWaterTemp.tickInterval = Math.round((maxWaterTemp - minWaterTemp)/5);
sliderWaterTemp.snapInterval = 1;
sliderWaterTemp.liveDragging = true;

sliderWaterAmt.minimum = minWaterAmt;
sliderWaterAmt.maximum = maxWaterAmt;
sliderWaterAmt.tickInterval = Math.round((maxWaterAmt - minWaterAmt)/10);
sliderWaterAmt.snapInterval = 1;
sliderWaterAmt.liveDragging = true;

sliderMetalTemp.minimum = minMetalTemp;
sliderMetalTemp.maximum = maxMetalTemp;
sliderMetalTemp.tickInterval = Math.round((maxMetalTemp - minMetalTemp)/10);
sliderMetalTemp.snapInterval = 1;
sliderMetalTemp.liveDragging = true;
```

```
sliderMetalAmt.minimum = minMetalAmt;
sliderMetalAmt.maximum = maxMetalAmt;
sliderMetalAmt.tickInterval = Math.round((maxMetalAmt - minMetalAmt)/10);
sliderMetalAmt.snapInterval = 1;
sliderMetalAmt.liveDragging = true;

// Slider event listeners
sliderWaterTemp.addEventListener(SliderEvent.CHANGE, changeWaterTemp);
sliderWaterAmt.addEventListener(SliderEvent.CHANGE, changeWaterAmt);
sliderMetalTemp.addEventListener(SliderEvent.CHANGE, changeMetalTemp);
sliderMetalAmt.addEventListener(SliderEvent.CHANGE, changeMetalAmt);

// Button event listeners
start_btn.addEventListener(MouseEvent.CLICK, onStart);
stop_btn.addEventListener(MouseEvent.CLICK, onStop);
reset_btn.addEventListener(MouseEvent.CLICK, onReset);

// Initialize Stage
initializeStage();

// Slider event handlers
function changeWaterTemp(event:SliderEvent):void {
        initWaterTemp = event.value;
        multimeter_mc.ch1_txt.text = String(formatDecimals(initWaterTemp,2));
}
function changeWaterAmt(event:SliderEvent):void {
        water_mc.height = event.value;
        initWaterAmt = event.value;
        multimeter_mc.ch2_txt.text = String(formatDecimals(initWaterAmt,2));
}
function changeMetalTemp(event:SliderEvent):void {
        initMetalTemp = event.value;
        multimeter_mc.ch3_txt.text = String(formatDecimals(initMetalTemp,2));
}
function changeMetalAmt(event:SliderEvent):void {
        metal_mc.scaleX = event.value/maxMetalAmt;
        metal_mc.scaleY = event.value/maxMetalAmt;
        initMetalAmt = event.value;
        multimeter_mc.ch4_txt.text = String(formatDecimals(initMetalAmt,2));
}
```

```
// Button event handlers
function onStart(event:MouseEvent):void {
        reset_btn.enabled = false;
        start_btn.enabled = false;
        sliderWaterTemp.enabled = false;
        sliderWaterAmt.enabled = false;
        sliderMetalTemp.enabled = false;
        sliderMetalAmt.enabled = false;
        stop_btn.enabled = true;

        // Calculate final temperature
        finalTemp = calculateFinalTemp();

        // Set up timer for animating final temperature
        tempTimer = new Timer(1000);
        tempTimer.addEventListener(TimerEvent.TIMER, animateFinalTemp);
        tempTimer.start();
}
function onStop(event:MouseEvent):void {
        reset_btn.enabled = true;
        stop_btn.enabled = false;
        tempTimer.removeEventListener(TimerEvent.TIMER, animateFinalTemp);
}
function onReset(event:MouseEvent):void {
        start_btn.enabled = true;
        sliderWaterTemp.enabled = true;
        sliderWaterAmt.enabled = true;
        sliderMetalTemp.enabled = true;
        sliderMetalAmt.enabled = true;
        stop_btn.enabled = false;
        initializeStage();
}
```

```
function animateFinalTemp(event:TimerEvent):void {
	if(initWaterTemp < finalTemp){
		initWaterTemp += Math.random()*(finalTemp+.03-initWaterTemp)/2;
		tempTimer.delay = Math.random()*1500;
		multimeter_mc.ch1_txt.text = String(formatDecimals(initWaterTemp, 2));
	} else if(initWaterTemp > finalTemp) {
		initWaterTemp -= Math.random()*(initWaterTemp+.03-finalTemp)/2;
		tempTimer.delay = Math.random()*1500;
		multimeter_mc.ch1_txt.text =  String(formatDecimals(initWaterTemp, 2));
	} else {
		tempTimer.removeEventListener(TimerEvent.TIMER, animateFinalTemp);
	}
}

// Format number (num) to given number (digits) of decimal places.
function formatDecimals(num, digits):String {
	if (digits <= 0)
		return String(Math.round(num));
	if (num < 0) {
		var isNegative = true;
		num *= -1;
	}

	var tenToPower = Math.pow(10, digits);
	var cropped = String(Math.round(num * tenToPower));

	if (num < 1) {
		while (cropped.length < digits+1)
			cropped = "0" + cropped;
	}

	if (isNegative) cropped = "-" + cropped;

	var roundedNumStr = cropped.slice(0, -digits) + "." + cropped.slice(-digits);
	return roundedNumStr;
}
```

```
function initializeStage(){
        metal_mc.scaleX = minMetalAmt/maxMetalAmt;
        metal_mc.scaleY = minMetalAmt/maxMetalAmt;
        water_mc.height = minWaterAmt;
        // Assign initial values
        initWaterTemp = minWaterTemp;
        initWaterAmt = minWaterAmt;
        initMetalTemp = minMetalTemp;
        initMetalAmt = minMetalAmt;
        // Set slider values
        sliderWaterTemp.value = initWaterTemp;
        sliderWaterAmt.value = initWaterAmt;
        sliderMetalTemp.value = initMetalTemp;
        sliderMetalAmt.value = initMetalAmt;
        // Display output for initial values
        multimeter_mc.ch1_txt.text = String(formatDecimals(initWaterTemp,2));
        multimeter_mc.ch2_txt.text = String(formatDecimals(initWaterAmt,2));
        multimeter_mc.ch3_txt.text = String(formatDecimals(initMetalTemp,2));
        multimeter_mc.ch4_txt.text = String(formatDecimals(initMetalAmt,2));
        // Button state
        stop_btn.enabled = false;
}

function calculateFinalTemp():Number{
        // cpWater, cpMetal, initWaterTemp, initMetalTemp, initWaterAmt, initMetalAmt
        var temp:Number = ??;
        return temp;
}
```