HTML Gauge

Wednesday, March 31, 2021 03:34

I'm on maybe my 20th HTML/JS gauge, and still don't have the comprehensive answers, but have working gauges and a reasonable workflow. To explain my workflow I've included the files.

HTML, CSS, JS code of an example gauge below which paints to a 256x256 `$flap_display` texture, in an "AS-33" aircraft.

The gauge files are in `html_ui/Pages/VCockpit/Instruments/AS-33/flap_indicator/`

The gauge uses two images "background_ok.png" and "background_nok.png" both of which are simply alternate plain LCD backgrounds 256x256, one normal, one pinkish.

The example only writes to the $texture - most of my HTML/JS gauges are also updating local variables ("L:..." vars) which are used in the `model/AS-33_interior.xml` to animate needles etc.

In terms of **workflow** (your actual question) the trick is to update the 'querystring' letters after the references to the `.js` and `.css` files in the `.html` any time you change either of those, and in the 'Dev' **Aircraft Selector** window click **load.** I.e. see `flap_indicator.html` below, the relevant lines (with redundant '?br' and '?dp' added to the url's) are:

Code:
```
<link rel="stylesheet" href="flap_indicator.css?br" />
```
and
Code:
```
<script type="text/html" import-script="/Pages/VCockpit/Instruments/AS-33/flap_indicator/flap_indicator.js?dp"></script>
```
Given I start with "?aa" on the end of those url's, you can see I do a lot of updates & reloads...

This method works because adding the harmless **?xx** to the end of the JS/CSS url will cause MSFS to think you're referencing a new file & won't re-use the current one from its 'cache'.

As a couple of extra notes on this gauge example:

The HTML contains multiple 'divs' layered on top of each other using absolute positioning.

For the gauge you do not define the entire page complete with <html> tags - MSFS controls that. See my .html file below.

You update the content of a div using standard JS, i.e. `document.getElementById("mydiv").innerHTML = "HELLO";` - this is how the example writes e.g. the FLAP INDEX as a text number into the div.

You hide/show divs using standard JS, i.e. `document.getElementById("mydiv").style.display='none'` (or 'block') - that's the way this example flips the background images.

There is a fairly complicated linkage between the references in the HTML/JS/CSS files and in most gauge examples the same reference is used for multiple different components. In my example below the JavaScript class is called **flap_indicator_class**, the MSFS-created special HTML element is called **flap_indicator-element** and the tag within your `.html` file that will embed your html inside that element is called **flap_indicator_script** so you can see where these things link to each other in the example - that's impossible to see when all those things are just called `flap_indicator`, so having different id's is better for an example (but ultimately not necessary).

The tiniest mis-step in the JS will cause the gauge to NOT DISPLAY AT ALL, and debugging then is difficult (there are tools that can help, but it's still difficult). so the **MOST IMPORTANT ELEMENT OF THE WORKFLOW** is to make changes in small steps so when the gauge blows up (it will...) you have pretty good idea of where the issue occurred. For future reference, putting a 'try..catch' around the updates called from within the "Update()" function, and writing an error code to the "#debug" div will help a lot, but I didn't want to make the example more complex.

The example below doesn't include the ability to 'click' on the gauge, i.e. as if it's a touch screen, but again that's standard JS `document.getElementById('mydiv').onclick = (your function)` but also you must have a class method "`get isInteractive() { return true; }`" for clicking to be permitted.

panel.cfg:

Code:
```
VCockpit08]
size_mm=256,256
pixel_size=256,256
texture=$flap_display
emissive          = 0
htmlgauge00=AS-33/flap_indicator/flap_indicator.html,0,0,256,256
```
flap_indicator.html

Code:
```
<link rel="stylesheet" href="flap_indicator.css?br" />
<script type="text/html" id="flap_indicator_script">
```

```
    <div id="background_ok"></div>
    <div id="background_nok"></div>
    <div id="display"></div>
    <div id="debug"></div>
</script>
<script type="text/html" import-script="/Pages/VCockpit/Instruments/AS-33/flap_indicator/flap_indicator.js?dp"></script>
```

flap_indicator.css

Code:

```
:root {
  --bodyHeightScale: 1; }
@keyframes TemporaryShow {
  0%, 100% {
    visibility: visible; } }
@keyframes TemporaryHide {
  0%, 100% {
    visibility: hidden; } }
html {
  height: 100%;
  width: 100%;
  overflow: hidden; }
  html body {
    -webkit-user-select: none;
    font-family: Roboto-Regular;
    font-size: calc(var(--viewportHeightRatio) * (36px / 21.6) * var(--currentPageHeight) / 100);
    color: white;
    height: 100%;
    width: 100%;
    margin: 0;
    padding: 0; }
flap_indicator-element {
  background-color: #121212;
  height: 100vh;
  width: 100vw;
  display: inline-block;
  overflow: hidden; }
/* Normal LCD background */
#background_ok {
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background-image: url("/Pages/VCockpit/Instruments/AS-33/flap_indicator/background_ok.png");
    background-repeat: no-repeat;
    /* background-size: cover; */
}
/* Warning LCD background */
#background_nok {
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    /* display: none; */
    background-image: url("/Pages/VCockpit/Instruments/AS-33/flap_indicator/background_nok.png");
    background-repeat: no-repeat;
    /* background-size: cover; */
}
#display {
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background: transparent;
    font-size: 200px;
    text-align: center;
    /* vertical-align: middle; */
    color: black;
    display: flex;
    justify-content: center;
    align-content: center;
    flex-direction: column;
}
flap_indicator-element #debug {
    position: absolute;
    top: 5%;
    left: 30%;
    width: 30%;
    height: 10%;
    font-size: 45px;
    color: black;
}
```

flap_indicator.js

Code:

```
/*
  flap_indicator
```

```
*/
class flap_indicator_class extends BaseInstrument {
constructor() {
        super();
        this._isConnected = false;
    }
get templateID() { return "flap_indicator_script"; } // ID of <script> tag in flap_indicator.html
// *********************************************************************
    // ********** CONNECTED CALLBACK        ****************************
    // *********************************************************************
    connectedCallback() {
        super.connectedCallback();
this._isConnected = true;
    }
// *********************************************************************
    // ********** DISCONNECTED CALLBACK        *************************
    // *********************************************************************
    disconnectedCallback() {
        super.disconnectedCallback();
    }
// *********************************************************************
    // ********** GAUGE UPDATE CALLED ON SIM UPDATE CYCLE     ************
    // *********************************************************************
    Update() {
        // We read the sim variables into local vars for efficiency if multiple use.
        this.ALTITUDE_M = SimVar.GetSimVarValue("A:INDICATED ALTITUDE", "meters");
this.flap_index = SimVar.GetSimVarValue("A:FLAPS HANDLE INDEX","number");
        this.spoilers_out = SimVar.GetSimVarValue("A:SPOILERS HANDLE POSITION","percent") > 0;
        this.gear_down = SimVar.GetSimVarValue("A:GEAR HANDLE POSITION", "bool") ? true : false;
        //DEBUG BALLAST NOT IMPLEMENTED
        this.carrying_ballast = false; //get(DATAREF_BALLAST_KG) > 20;
        this.time_now_s = SimVar.GetSimVarValue("E:ABSOLUTE TIME","seconds");
        this.alt_agl_ft = SimVar.GetSimVarValue("PLANE ALT ABOVE GROUND", "feet");
this.update_flap_indicator();
    }
// *********************************************************************
    // ********** GAUGE UPDATE CALLED ON SIM UPDATE CYCLE     ************
    // *********************************************************************
// Runs once on startup
    flap_indicator_init() {
        if (this.flap_indicator_init_complete == null) {
this.WHEEL_DOWN_ALT_LIMIT_FT = 2000; // warning if gear down above 2000 feet AGL
this.MAX_WARNING_DURATION_S = 5; // only leave warning on indicator for max time
            this.warning_time_s = 0.0; // record time of issuing warning
            this.warning_gear_up = false;
            this.warning_gear_down = false;
            this.warning_ballast = false;
this.background_nok_el = this.getChildById("background_nok");
            this.display_el = this.getChildById("display");
            this.display_el.innerHTML = "AB";
            this.flap_indicator_init_complete = true; // prevent further runs
        }
    }
update_flap_indicator() {
        this.flap_indicator_init();
        // SPOILERS / GEAR UP WARNING
        if (this.spoilers_out && ! this.gear_down) {
            if (! this.warning_gear_up) {
                this.background_nok_el.style.display = 'block';
                this.display_el.style.fontSize = '50px';
                this.display_el.innerHTML = 'SPOILERS<br/>GEAR UP';
                this.warning_time_s = this.time_now_s;
            }
            this.this.warning_gear_up = true;
        } else {
            this.warning_gear_up = false;
        }
// AT HEIGHT / GEAR DOWN WARNING
        if (this.gear_down && (this.alt_agl_ft > this.WHEEL_DOWN_ALT_LIMIT_FT)) {
            if (! this.warning_gear_down) {
                this.background_nok_el.style.display = 'block';
                this.display_el.style.fontSize = '70px';
                this.display_el.innerHTML = 'GEAR<br/>DOWN';
                this.warning_time_s = this.time_now_s;
            }
            this.warning_gear_down = true;
        } else {
            this.warning_gear_down = false;
        }
let on_ground = this.alt_agl_ft < 10;
// SPOILERS / BALLAST WARNING
        if (! on_ground && (this.spoilers_out || this.gear_down) && this.carrying_ballast) {
            if (! this.warning_ballast) {
                this.background_nok_el.style.display = 'block';
                this.display_el['font-size'] = '40px';
                this.display_el.innerHTML = 'SPOILERS<br/>BALLAST';
                this.warning_time_s = this.time_now_s;
            }
            this.warning_ballast = true;
        } else {
```

```
            this.warning_ballast = false;
        }
let warning_expired = this.time_now_s > (this.warning_time_s + this.MAX_WARNING_DURATION_S);
let no_warning = ! this.warning_gear_down && ! this.warning_gear_up && ! this.warning_ballast;
if (no_warning || warning_expired) {
            // All seems fine, so set flap_indicator to flaprqst
            let display_text = '-';
            let font_size = '200px';
            switch (this.flap_index) {
                case 0:
                case 1:
                case 2:
                    display_text = ''+(this.flap_index+1);
                    break;
                case 3:
                    display_text = '4A';
                    font_size = '180px';
                    break;
                case 4:
                    display_text = 'T1';
                    font_size = '180px';
                    break;
                case 5:
                    display_text = 'T2';
                    font_size = '180px';
                    break;
                case 6:
                    display_text = 'Land';
                    font_size = '100px';
                    break;
                default:
                    break;
            }
            this.background_nok_el.style.display = 'none';
            this.display_el.style.fontSize = font_size;
            this.display_el.innerHTML = display_text;
        }
    }
}
registerInstrument("flap_indicator-element", flap_indicator_class);
```