Styling an item with TAO

The TAO platform offers different ways to change the look of an item. The most obvious and simple way to work on the look is through the style editor. There is a whole chapter in this presentation - taken from the upcoming TAO User Guide - that explains how this works.

The CSS SDK is based on a set of files which use the same CSS selectors we have used inside TAO and allows you to easily overwrite the default theme. It is planned to expand this SDK in the future to allow adding the same stylesheet to a large number of items.

Using either the SDK or the style editor is a decision that depends what you are trying to achieve and your prior knowledge of CSS.

Simple design modifications

In this scenario you are probably best off with the built-in Style Editor. It allows you to change the colors of the background, the text and the borders. You can change the font and also the font size. You can also modify the item width but this is something we would not recommend - in the TAO platform the item is responsive which means it will automatically adapt to the size of the device you are using. On the other hand, having a setting for this is a requirement of the QTI standard and there might well be situations where a fixed item width will become necessary.

Pros

- Requires no CSS knowledge
- Instant results
- Covers the most common scenarios

Cons

- The possibilities are limited
- All modifications need to be repeated for each item

Major design modifications

This is in any case a scenario for the CSS SDK. You will need to have at least some prior knowledge of CSS. The HTMLdog tutorial will help you to get at least a basic understanding. Codrops' CSS reference is also very good but maybe not so much for beginners.

You will also need some understanding of the browser console. All examples in this context refer to the developer console in Firefox but the concepts are very similar in all modern browsers. If you are unfamiliar with the browser console start with the tutorial on the Mozilla website. There is one that focuses on the Page Inspector and another one that explains the Style Editor (not to be confused with the Style Editor inside the TAO platform!).

Pros

- You can profoundly change the look of an item
- You can apply your own corporate identity

Cons

- You need to be familiar with CSS
- You need to know your way around the browser console

Keep in mind that depending on the the style modifications you wish make it might be necessary to install additional software.

Built-in Style Editor

Note: this chapter has been taken from the upcoming TAO User Guide

To access the Style Editor of the TAO platform, click on the Style Editor button in the Action Bar above the Properties Panel.

This will turn the Properties Panel into a Style Editor panel. There are two parts to the editor, the Style Sheet Manager at the top, and the Item Style Editor down below.

If you have a style sheet ready for upload, click on the *Add Style Sheet* button. This will provide an interface similar to that of adding a graphic into a Graphic Interaction. In all other cases, if you will be editing styles in a way that is special for this Item only, focus on the Style Editor below.

Creating style sheets are to be handled in a whole other entry.

The Style Editor, meanwhile, has three segments, governing: (1) Color, (2) Font, and (3) Item width. These can be adjusted in any order, but for the sake of this tutorial, we will be going in order from top to bottom.

Adjusting colors

There are four color swatches that can be changed in accordance with the needs of the user: Background color, Text color, Border color, and Table headings.

Clicking on the swatch opens a color editor panel that consists of a square surrounded by a ring, and a text box below. The ring provides the user with a color wheel that allows changes to the color. The square provides the ability to adjust contrast (left and right) and brightness (up and down). The textbox provides the ability to save a desired specific color setting as portions of Red, Green, Blue (in RGB hexidecimal-percent of primary color density).

The four swatches cover specific parts of any Item and its interactions. The Background color provides the color backing the entire item. The text color covers all text within the Item and its constituent interactions. The border color governs that of the borders of interactions. Finally, the table heading color swatch provides color to those interactions that use tables (such as Match).

Adjusting fonts

There are two font controls governing the font family selection, and the font size. Adjusting the font family involves clicking on the appropriately marked box (which should start out reading *Default*). There are a number of fonts that are built into the system, and these are divided into "sans serif" fonts (fonts without "feet" at the ends of letters), "serif" fonts (those with "feet"), and "monospace" fonts (fonts that resemble a typewriter, with each letter whether it be an "i" or an "m", having the same width as the other). The large number of fonts provides for significant flexibility in Item page design, but within the Item style editor, these are the only fonts that are available.

The other control allows Item authors to change the font size, pretty much to any size of font desired. To set a font size, click into the box and type in a number (13 pixel is the default). To adjust further from there, the Item author has the choice of either typing in a new number into the box, or clicking on the bigger or smaller font icons. There is no limit to the size of font that can be selected, but of course when you get past a certain size, the font will simply be too large to even partially display.

Adjusting Item width

By default the Item width adapts to the width of the user's screen. Therefore it is highly recommended that you do not change this setting.

Because the QTI standard requires the ability to set an Item width on screens, the option to set a width is made available on TAO. However, setting a width that does not adapt to screen size will mean that different sized computer screens may have problems with the setting. There really is no need to adjust this under normal test authoring situations.

If you do not like any of the settings you have selected, click on the eraser icon on the right of the control (the swatch or the text box), and the item will be restored to its default setting.

This can be particularly useful if in playing around with the item, the end result turns out to be something wholly unreadable. Default settings are always a click away.

Working with item styles

If you compare an online test to a paper based one you could say the item is the paper you are writing on. Or if you prefer word processors you could say it is an empty document. An item will in any case contain at least one interaction and probably some portions of text or pictures. There is an extra chapter that explains how to edit interaction styles.

Modifying the item style as such would typically concern the background color, the text color, the font or even the background picture.

The CSS SDK uses the selector `html body div.qti-item` to modify the item directly. A custom style for an item could look like this:

```
html body div.qti-item {
    background-color: #eeeeee;
    color: #333333;
    font-family: Arial, Helvetica, sans-serif;
    border: 2px #dddddd solid;
}
```

We recommend that you prefix every selector with `html body div.qti-item` to ensure you do not interfere with the appearance of the TAO platform itself.

Adding background pictures or fonts

In a regular web environment you would use the following code to add a background image:

```
html body div.qti-item {
    background-image: url(my-image.png);
}
```

Inside the TAO platform this is currently not possible. All external resources need to be compiled into the stylesheet by encoding them in `base64`. OAT provides a tool to convert resources to base64. With an encoded image the above code looks like that:

```
// code has been shortened for readability
html body div.qti-item {
    background-image: url(data:;base64,iVBORw0KGgoAAAANSUhEUgAAAAMAAAADCA
    YAAABWKLW/AAAAEUlEQVR42mNgYGD4z4AGMAUANfEC/pPYlKYAAAAASUVORK5CYII=);
}
```

The same rules apply in the scenario where you want to add an external font with the `@font-face` rules. They will also need to be encoded in `base64`. Given the browser support of TAO, you will only need the `*.woff` format. Using `*.woff2` in addition to that could help to make the item more future safe though.

Including external stylesheets with @import

As already mentioned above TAO does not support the inclusion of external resources, at least not in the way you would have expected. It is therefore not possible to include stylesheets with the `@import`-rule.

The structure of the interactions

All interactions share - as far as possible - a similar architecture.

The following example is based on an *order interaction* but its basics are the same across most interactions.

```
    <div class="qti-interaction qti-blockInteraction qti-orderInteraction">
        <div class="qti-prompt-container">
            <p class="qti-prompt">Prompt</p>
        </div>
        <div class="order-interaction-area">
            <ul class="choice-area">
                <li class="qti-choice">...</li>
            </ul>
            <ul class="result-area">
                <li class="qti-choice">...</li>
            </ul>
        </div>
    </div>
```

As you can see the interaction sits in a container with the following CSS classes:

- `qti-interaction`
- `qti-blockInteraction` (resp. `qti-inlineInteraction`)
- `qti-orderInteraction` (or any other type)

Below this level sits the `prompt`, which is embedded in a `qti-prompt-container`. The prompt is followed by two blocks, the `choice-area` and the `result-area`. This is typical for all interactions that consist of two blocks. The actual choices in this example have the class `qti-choice`.

If you want to change for instance the background of the choices only in the result area of this particular interaction your selector would be as follows:

```
html body div.qti-item .qti-orderInteraction .result-area .qti-choice {
    background: grey;
}
```

To modify them for both areas you would write instead:

```
html body div.qti-item .qti-orderInteraction .qti-choice {
    background: grey;
}
```

And finally if the style should apply to all block interactions:

```
html body div.qti-item .qti-blockInteraction .qti-choice {
    background: grey;
}
```

Working with the browser console

Probably one of the most important tools for a CSS developer is the browser console. In the context of this article we will concentrate on the one that comes with Firefox but the consoles in other modern browsers work very much the same way.

There are two panels on the console that are of particular interest, the Inspector and the Style Editor (not to be confused with the Style Editor inside the TAO platform!).

The Mozilla website has a tutorial for the browser console that pretty much focuses on the Inspector and there is also a page on the Style Editor.

It is important to note that the item and the interactions are different at design time and at runtime. When you are in the Item Creator these elements contain a number of additional parts such as toolbars, context menus etc. that are needed to edit the interaction. Inspecting the source code in this environment can be tricky or even misleading. A better way would be to display the item preview and to start the inspection there.

Finding the styles for an element

There are several strategies to find the right CSS selector to apply a particular style. The rules in the SDK are well named and commented but they cannot address every possible modification.

The most straight-forward way to find a selector is to use the Inspector panel. Right-click the element and select *Inspect Element* (or similar, depending on your browser) to display it in the Inspector. It happens often that the element you have chosen does not have the style you are looking for. In this case select the parent or child elements until you see the style in the *Rule* panel. You can use the Rule panel also to try out a style before you edit your actual style sheet.

Testing a stylesheet

The complicated solution to test a stylesheet is to go to TAO's Style Editor and upload your CSS file. You will then need to reload the page, select the item again and enter into authoring mode. This is obviously very impractical and this is where the browser's style editor comes to rescue. We found that the best way to start is to use an empty stylesheet, say `my-custom-styles.css` and then to use this to test your work in the browser console. Mind you that you should not call your stylesheet `tao-user-styles.css` because this is a reserved name. While nothing will break, this could still cause confusion.

If you compile your stylesheets with a pre-processor such as SASS you need at this point to perform an additional step: On the top right of the console click on the little gear ("Toolbar Options") and check the box that says "Show original sources" ("Show source maps" in some other browsers). This will help you to identify which of the many source files contains the style you want to edit.

When you scroll over the list of stylesheets on the left hand side of the panel, you will notice some files that start with *getFile*. One of them will have `my-custom-styles.css` at the end, this is the file you need to select. From this point you can copy/paste your new styles into this panel and the styles will be applied immediately.

What can you do if a style is not applied?

There is of course always the possibility that you have an invalid rule, a spelling mistake, a missing semicolon or anything like that. If you can rule all of this out you will need to check whether your selector is strong enough. Inspect the element in question in the browser console and find out which selector has been applied in the original CSS. It might well be that your style is there but struck through. In this case you will need to edit the selector and make it more specific. There is a good article on CSS Tricks to illustrate this. That said, there also an easy way out - you can simply use a Specificity Calculator to tackle the problem. Adding `!important` to a rule - as it is sometimes recommended - should always be the very last resort and only be used with great care.

Using a pre-processor for your stylesheets

In recent years it has become increasingly popular to generate stylesheets with a pre-processor such as *SASS*, *LESS* or *Stylus*. TAO uses *SASS* in the *SCSS* flavor for its stylesheets and this works very well. All source files within the SDK are available either as pure `*.scss` or as `*.css` files. If you have the chance to use SASS on your computer you should definitely do this. Of course any other pre-processor can be used but we do not provide any sources other than *SASS/SCSS*.

To work with SASS, open a command line window, navigate to the *SCSS* folder of your project and type

```
sass --watch .:../css
```

Working directly with CSS

The SDK contains a complete stylesheet with the same selectors found in the `*.scss` files. It contains no *CSS* rules but only empty selectors and comments. For the practical work you should not edit this stylesheet directly because you would essentially create a rather large CSS file of which most lines are unused. A better approach would be to start with an empty stylesheet and copy/paste the lines that you need for the rule you are working on from the original file. This way your code remains small and readable whilst keeping the SDK documentation intact.

Software and resources

The Firefox Style Editor at this point is the most suitable on the market. Chrome on the on the other hand requires no administration rights for installation.

We recommend that you compress all pictures you create for a stylesheet. PngOptimizer for Windows does this very well but there are many others too. TinyPNG is just one of the many online services that can achieve the same. There is currently no standalone compressor for JPEG we could advise - you will need to rely on your graphics editor.

When you work with fonts via `@font-face` you should not use them as an external resource, for instance via Google Fonts. The reason is that you have no guarantee that a student who takes a test has also unrestricted access to the internet. Fontsquirrel has a nice generator for web fonts that allows to download a font in the right format and to include it in the CSS. Remember that in order to match TAO's browser requirements you need the font only in the *.woff* format. Using *.woff2* in addition to that could help to make the item more future safe though.

If you work with pictures and fonts you need to convert them into a *base64 data URI*. We have a converter on the TAO Style Guide that you can use to generate data URIs.

If you need a tutorial on CSS, HTMLdog has a good one. It may be a bit outdated but certainly covers most of your needs. Codrops' CSS reference is an excellent reference for those who already know their way around stylesheets.

We found this Calculator for CSS Specifity to work nicely.

The Sass website has a beginner's guide that should help you to get started. There is also a page that explains the installation of SASS. You will need administrator rights for this installation.

The Mozilla website has a tutorial for the browser console that pretty much focuses on the aspects we are interested in. There is also a page on the Style Editor.