Mobile UI Options GSoC 2016 Updates

Initial styling of the responsive design only with Stylus

During my first days of GSoG I used only Stylus to change the general look of the UI options for small screens (max-width: 640px). The reason for that was that I was not very familiar with the infusion framework and wanted to seen what could be made without changing the template. I managed to get the UI panel to be at the bottom of the page and made some adjustments to the panel with basic properties.

A new tool (to me) that I found very useful was the flexbox (https://css-tricks.com/snippets/css/a-guide-to-flexbox/). My mentor (Jon Hung) told be about it as an easy to change the ordering of the list elements from horizontal to vertical. I also used it to center elements and the flex-basis option was particularly useful when it came to redesigning the contrast panel. Overall, I think it is a great tool and with a little bit of thought and creativity it can be used to solve complex problems.

After I made the panels order vertically I had the problem that the framework fixed the height of the UI panel and the content was overflowing. I added so js to fix that, but I am still not really sure it this is a good place for the this code to be at (https://github.com/styki/infusion/blob/Responsive-Design/src/framework/preferences/js/SeparatedPanelPrefsEditor.js#L302-L325). I also added the scroll-to-next-panel option here, which was a simple use of scrollTop and rounding.

After these things were done, Jon and I decided to focus on the textfieldSliders, which had to be changed to textfieldButton. This required in-depth knowledge about the infusion framework.

Comments about the documentation of infusion

It took me more that a week to familiarize myself with the framework in order to be able to create a new template and make all the back-end connections. I think that anyone how sees the framework for the first time has to start by reading the tutorials. I made the mistake to start reading the ContextAwareness documentation and every time I saw something that I did not know I open the link. This link brought me to new links and so on until I got to the Core API documentation. The tutorials gave me a good basic knowledge and understanding about the framework. The examples were very useful and made everything absolutely clear. I think that I will be a good idea to add more examples in the general documentation, because there are a lot of options for every feature and it is hard to understand how everything works until you see it in practice.

Creating the new template for the TextfieldButtons

To make the responsive versions for the textSize and lineSpacing I first made a new template. I then had to make the StartedSchema switch to for small screens. I used to js and the ContextAwareness feature to make and remove the checks, but the problem was that the program responded to these changes only when it was first ran. If you resize the window is did not change the template. The reason for that is that once a component is built it is immutable. This problem with resizing still remains unsolved.

I then made the created the TextfieldButtons.js file and changed the Panels.js, so that it provided the appropriate content to initialize the model and make the connections. After that was done and it functioned properly I pulled from master to see what the changes are. There were some conflicts, because the slider was refactor to support a nativeHTML and a jQuery version. I then tried to add my changes without making major changes to the logic of the master branch.

The main issue that came up was that the we had to switch between 3 templates, but the slider checks have to be made only if we are on a large screen. One difficulty that I encountered was that options of a component don't get overridden. As a result a could not make 2 separate contextAwareness checks and it the small screen check is true to override the information provided by the other one. A had the idea to first make the small screen check. If it is true I add the textfieldButtons template. If it is false I go to another component. Inside of it there is another contextAwareness with the jQuery check. However, this contextAwareness nesting did not work. I still have no idea it does not function properly.

The only solution that I could think of was to make the window size check in the main component (https://github.com/styki/infusion/blob/Responsive-Design/src/framework/preferences/js/StarterSchemas.js#L158-L167) and the jQuery check in a different component defined in an if statement, which is true only for large screens (https://github.com/styki/infusion/blob/Responsive-Design/src/framework/preferences/js/StarterSchemas.js#L33-L51). This lead to the same problem when the screen is resized.

Implementing a blue color filter

Our next goal was to make a tool that can change the intensity of the blue color in all components in the body. To do that I first had to select all elements inside the body and store their initial colors. To do that I used a dictionary data structure. In it the key is element (js this object) and the values are the color and the background color. I could not find a build-in implementation of this data structure, so my version might not be the most effective. For each element it uses a linear search to go through the dictionary and find if there is a match. This results in an overall quadratic cost. For more complicated web pages this might be a big problem. Unless we can find a more efficient way to go through the records this was the solution I could think of.

After the program finds the element in the records it takes its colors, which are stored in an array of their rgb or rgba components. It then multiplies the blue components by the model value of the blue color filter component. The value is then stored and the elements gets its colors changed with jQuery. The blue color filter component is similar to the lineSpace and textSize ones. It has a textfieldSlider for large screens and textfieldButtons for small screens. They both have a min value of 0, a max value of 1 and a step of 0.1. The default value is 0, which mean no change, and the max 1 means 100% blue color filter or no blue component. Specific icons for the component need to be added.

Known bugs:

I managed to solve most issues associated with the new component, but there was one that I still don't know how and why it occurs. It is when you have the table of content opened and leave the blue color filter to a value different than 1 (let's say 0.5). When you reload the page it remembers the blue color filter value and applies the change upon loading. The problem is that the header of the ToC does not get the right original color stored in the dictionary. In the records we get to store the actual color multiplied by the model value (or in the example 0.5 of the real blue component). This is a huge problem if the model value is left at 0 and then the page is reloaded. We then end up with a 0 as the blue component no matter to what value we change the model after that. This issue is caused by the fact that we first apply the change in colors once before storing the elements in the dictionary. However, this should be impossible, because an element has to be found in the records in order for the program to change its colors.

I faced the problem that the blue color filter has to be applied whenever an element is removed or added to the DOM. To do that I used a Mutation Observer (https://developer.mozilla.org/en/docs/Web/API/MutationObserver). This was a new feature for since it has been released in 2015. I found it quite useful and quickly solved my problem. I even used it to track an element's attribute to determine if the UI panel is opened and then add/remove a class to the responsive hide/show button to style it properly. However, there is a small issue with Safari. It throws and error even if you a webkit prefix for the MutationObserver. This issue has to be solved by someone on a Mac, because the latest version of Safari for Windows is 5.1 and MutationObservers are supported in 6+.

Adding image filters for contrast themes

I used the CSS property filter to change the color of the images in the page to make them match the contrast theme. Two issue came up as I was implementing the feature.

When a contrast theme is applied the blue color filter has to be disable, because they are not compatible.When a theme is selected the slider and the textfield disappear and a message is shown. The problem is that the blue color filter panel gets updated only when the UI panel is closed and reopened or the page is reloaded. The code that makes the image appear is vanilla CSS and is strange that it does get applied immediately. I talked to Jon and he said that there was an old issue, which involved the iframe and its changes not being applied at run time. I tried using some js to add a class to the blue color filter when it had to display the message, but jQuery could not detect the elements in the iframe. I can go through the code and find where the iframe is being build, but this will make the program really messy and I am not sure if it will fix anything.

The other issue is on webkit browsers. When you apply the filter option to multiple images some of them disappear. I have no I idea why this happens. I saw that people have had this issue for years, but no one had a working fix.

Implementing a mute button

The last feature that I implemented was a button to mute all audio and video tags in a web page. This was simply done by having an on/off button similar to the table of content one. When it is turned on it adds a mute property to all audio and video tags. When it is turned off it unmutes them. The implementation targets only elements that are not initially muted. The idea behind that is that if you are for example in Facebook and only one video is play sound, so you turn of the mute button. If you turn it off and it unmutes all elements in the page this will be a problem. However, there is a small bug associated with that. If you have the button turned off and you reload the page it will function properly. If it is left on and you reload it will detect every video as initially unmuted, so it will mute it and add it to the array of elements to be unmuted later on. When the mute option is remove every video will start playing audio, even the ones that had a muted property initially. This can be see and tested in the UI demo (https://github.com/styki/infusion/tree/Responsive-Design/demos/uiOptions). I think that this bug is similar to the one with the blue color filter and the table of content heading. I think that when the infusion components are being build js detects that the elements are there, but it ignores their attributes.

Final thoughts

I had a great time working for Inclusive Design Institute. During this summer I had the opportunity to talk to new people and deepen my knowledge in the computer science field. The people that I worked with were amazing. They always helped me and made it easier for me to continue working and to implement new features. At first I did not understand how the infusion framework worked and had to study it for about 10 days until I started to get some understanding. Although I sometimes felt like I am write a little bit of unnecessary code, I realized that the infusion framework helps developers have a universal tool for building components. It is really systematized and it provides consistency for the code development and conventions.

Overall, this was a great experience and I would strong recommend the company to any of my friends if they have the opportunity to work for Inclusive Design Institute. I will continue to contribute to the project whenever I have time and if there is another project next year I will apply again.