Chrome's Proposed Feature - by uBlock Origin

Nicolas Merz, Zach Yale, Heather Geiger, Darrien Park, Kurt Blair, Daniel Kailly

November 30, 2018

Abstract

In this report, our team evaluates established software solutions meant to reduce eye strain and negative impact on sleep quality. The team proposes a new Night Mode for Chrome, which will include support for an OS-agnostic dark UI, and a toggleable 'smart invert'to give websites a dark scheme. A SAAM Analysis was conducted for the feature, where a set of Non-Functional Requirements are identified. Two different implementation approaches for the feature are introduced and compared from which an implementation method is selected for hypothetical development. A stakeholder impact assessment is conducted to evaluate potential positive and negative effects from implementing this feature. The feature's impact on Chrome's architecture is assessed, and feature impact on the UI, Browser, and Renderer subsystems are outlined. Two use cases are assessed- where a user toggles the Night Mode on while not on a web page, and where a user toggles the Night Mode on while on a web page. To illustrate the flow of these use cases, sequence diagrams are presented. A set of risks and limitations for the feature are identified, and finally the group identifies some of the team issues and lessons that were learned while defining the Night Mode feature.

Contents

1 Introduction and Background

People use their computers during a wide range of times throughout the day. The lighting of their environment changes throughout the day, so why should the design of the software they use remain static?

In the evening and at night, people continue to use their devices with the same bright, light-coloured interfaces designed for usage during the day. However, using screens at night can cause eye strain in dimmer lit environments, and can be detrimental to sleep quality and duration. It has been documented that exposure to blue light at night time " is part of the reason so many people don't get enough sleep" [1]. In the United States, the American Medical Association has recommended that "exposure to excessive light at night, including extended use of various electronic media, can disrupt sleep or exacerbate sleep disorders." [2]. However, some measures can be taken to improve sleep quality and duration, such as filtering out blue light or reducing intensity [3].

Upon the release of scientific findings, companies have begun to adapt their interfaces and operating systems to better accommodate technology usage at night. Apple, Google, and Microsoft have incorporated support into their major desktop and mobile operating systems for night-shift filters, which work by adjusting the colour temperature of the display to reduce visible blue light. However, the issue of light intensity remains as there are limits to how dim a display can become while remaining readable. To help make up for this, dark user interface designs help reduce the strain on a user's eyes, while keeping the display bright enough that it is still readable. With the release of Mac OS Mojave in September 2018, and the October 2018 update for Windows 10, both Mac and Windows now support a system-wide dark mode [4]. While the Chrome developer team has noted that they have plan on supporting Mac OS Mojave's dark mode in the future, they do not currently plan on introducing a dark mode for any other operating systems [5]. As such, our proposed feature addresses this area to incorporate night time browsing improvements for the Chrome browser.

2 Feature Overview

Given the established scientific findings surrounding technology usage at night, and the Chrome team's limited plans on supporting night-friendly UI modifications, the team proposes that the Chrome browser be adapted to include a new overall Night Mode feature. Our proposed Chrome Night Mode will be OS-agnostic, to support Mac, Windows, Linux, and Chrome OS operating systems. As well as adapting the UI colour scheme, it will also invert web pages to a darker theme to accommodate night time browsing.

Feature highlights of Night Mode include:

- Added toggle to apply Night Mode within the Settings tab on the UI toolbar (UI)
- An added Night Mode profile that is saved as default on the user's Google account when Night Mode is toggled on (Browser)
- The text and background is inverted for all rendered websites, forcing them to use light text on a dark background (Renderer)

By introducing support for a browser Night Mode that is OS-agnostic, the benefits of a Night Mode will no longer be limited to Mac OS users. This feature implements the scientific findings which emphasize that dark interfaces are a better alternative than light interfaces, and can improve sleep health.

[1] Harvard Health. (2018). Blue light has a dark side - Harvard Health. [online] Available at: https://www.health.harvard.edu/staying-healthy/blue-light-has-a-dark-side [Accessed 27 Nov. 2018].)

[2] Blask, D., Brainard, G., Gibbons, R., Lockley, S., Stevens, R. and Motta, M. (2012). Light Pollution: Adverse Health Effects of Nighttime Lighting. Report of the Council on Science and Public Health, 4, p.12.

[3] Rahman, S., Shapiro, C., Wang, F., Ainlay, H., Kazmi, S., Brown, T. and Casper, R. (2013). Effects of Filtering Visual Short Wavelengths During Nocturnal Shiftwork on Sleep and Performance. Chronobiology International, [online] 30(8), pp.951-962. Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3786545/ [Accessed 29 Nov. 2018].

[4] Smith, M. (2018). Turn off the lights. Dark Mode on Windows 10 and MacOS compared. [online] Digital Trends. Available at: https://www.digitaltrends.com/computing/dark-mode-macos-mojave-vs-windows-10/ [Accessed 30 Nov. 2018].

[5] (https://bugs.chromium.org/p/chromium/issues/detail?id=850098)

3 Derivation Process

The team made an initial effort to group together and brainstorm different ideas for a feature in Google Chrome, largely drawing from our own experiences with Chrome and what we would like to see implemented. After some discussion, our ideas were narrowed down to four options: tab suspension, a night/dark mode, browsing continuity between devices, and automatic cache clearing. The team found that many of these ideas were either an experimental feature or was already implemented.

Automatic tab suspension exists and is known as Tab Discarding and is enabled by default on Chrome (which can be toggled by changing a flag) [6], but the team did not realize this initially as it is somewhat subtle. Browser continuity also exists using a Google account (for example, you can be browsing on your desktop and can get the same tabs on your phone if you sync open tabs) [7]. Finally, automatic cache clearing does not exist exactly, but Chrome can be set so that it clears the cache on closing [8].

Because of this, Night Mode was chosen. Though it does partially exist as an experimental feature (Google is introducing a dark-skinned UI to integrate with the macOS dark mode), the team's proposed feature is distinct in that it has a standalone toggle, and has the further functionality of web page skinning which will be discussed later in the report.

4 SAAM Analysis

4.1 Overview of SAAM Analysis

1. Chose a set of Non-Functional Requirements (Quality Attributes) that would make Night Mode a success and which we could use to assess the architecture
2. Compare advantages and disadvantages of both implementation approaches to evaluate the degree to which each architecture provides support for each NFR
3. Propose some testing opportunities that prove Night Mode's success
4. Discuss the impacts on Stakeholders

4.2 Chosen Quality Attributes for Feature

We chose the following 6 NFRs as the key performance indicators for our feature: Performance, Stability, Availability, Maintainability, Evolvability, and Testability.

- Assessing **performance** is important to ensure that the addition of our feature does not hinder Chrome's performance for instance: Night Mode user's should observe load time's of at least 90% of Chrome's regular load speed. A slight slow down might be encountered as the Renderer works to parse the website code and invert the colour scheme
- **Stability** is important as we are making source code modifications. The changes we make must be tested in order to ensure that Night Mode does not cause additional crashes that would not be present in standard light Chrome.
- **Availability** is important as timing differs from usual applications. We need to ensure that night mode is available and functioning overnight when users need it most between hours of 4pm and 7am it should have peak functionality.

[6] Osmani, A. (2015). Tab Discarding in Chrome: A Memory-Saving Experiment. [online] Google Developers. Available at: https://developers.google.com/web/updates/2015/ 09/tab-discarding [Accessed 28 Nov. 2018].

[7] Kaufman, L. (2017). How To Sync Open Chrome Tabs View Them Across Devices. [online] trendblog.net. Available at: https://trendblog.net/sync-open-chrome-tabs-across-devices/ [Accessed 29 Nov. 2018].

[8] Megabyte. (2018). Automatically Clear Cache on a Site When Closing Chrome - Megabyte. [online] Available at: https://megabyterose.com/2018/ 01/clear-cache-on-exit/ [Accessed 29 Nov. 2018].

- **Maintanbility** is important because, as mentioned, this feature is new and it modifies source code, updates and changes are expected during beta releases while the implementation is perfected. Overnight tests and updates are not an option as this is when user's need Night Mode most, so we chose tests and updates to run between 7am and 9am to maintain the feature.
- **Evolvability** will be accounted for by ensuring that modifications to the source code are well-documented, and limited in scope where possible. This will ensure that the feature can evolve with Chrome as further enhancements are made to the browser over time. and easily changed over time with new updates and developments in Chrome.
- **Testability** is assessed when files are modified or added to the source code, new test cases will be added to existing set of automated unit and integration tests. This ensures that the addition of our feature does not further complicate the Chrome team's testing protocols.

4.3 Compare Implementation Approaches

To compare the two implementations, we outline the following advantages and disadvantages of each approach, as shown in Figure 1.

Chosen Implementation
Advantages:

- Better ability to test and optimize Performance
- More Portable as web page source code changes consider all OS functionality
- Guaranteed Availability between 4pm and 7am.

Disadvantages

- More invasive source code modification which could have Stability ramifications
- No developer input on site appearance
- Could cause inconsistencies, loss of text/image visibility, etc.

Alternative Implementation
Advantages:

- More Modifiable because no regular updates required by the Chrome team
- Likely favoured by site owners and developers as they can choose their own Night Mode features & degree of adoption

Disadvantages

- Heavily reliant on website developers to implement Night Mode on their sites which means:
  - No way to monitor Maintainability
  - No guarantee on Availability

Figure 1: Compare Chosen and Alternative Implementations

To reiterate, our chosen approach impacts the Browser, UI, and Renderer. When the user toggles on Night Mode, their Profile Flag used by Browser is changed to Night Mode. This is the same kind of profile flag that a Child/supervised user or Incognito mode would have. When the user opens a new Night Mode window, the UI is changed to the Native Dark Aura theme, the Renderer parses the site's CSS and performs a HEX inversion on the colours. The advantages of this approach is that it directly modifies website's source code. This means that we are better able to test and optimize the feature's performance since the implementation resides within Chrome's source code. It is more Portable as the feature is OS-agnostic and we have the ability to add mobile functionality. Finally, we can better guarantee the Availability of Night Mode in this implementation because the feature resides in Chrome and is independent of website updates and modifications made by developers. The invasive source code modifications could be seen as a disadvantage in case any Stability ramifications occur. We plan to do thorough testing of the enhancement before release. Furthermore, the site developers may not approve of how their site looks in Night Mode because the colour inversion could cause visual inconsistencies or impaired aesthetics as web designers may not have intended for the colours to be invertible.

Our proposed alternative is to release a Google HTML feature tag where sites can create a Night Mode version of their website's CSS. Chrome's browser detects the tag and displays the dark version of the site if the user has Night Mode toggled on. Ultimately, we decided against this implementation approach as

it is too reliant on website developers. The advantages of this approach are that it is more Modifiable for the Chrome team since the individual websites are altered by the site developers. This approach might be favoured by site developers as they could re-design the colour scheme, site layout and logos for the Night Mode profile. The primary disadvantage of this, and the reason we chose not to go with it, is that it relies too heavily on external parties. Also, we want to be able to run updates and tests to maintain the feature and we want to guarantee availability of the feature. If developers are slow to adopt the feature then many sites on Chrome would not offer Night Mode compatibility and users will be forced to browse without the benefits of Night Mode despite having Night Mode toggled on. Since the user is a primary stakeholder, we want Night Mode to be implemented on as many sites as possible.

4.4 Testing

Tests must be conducted to confirm that the proposed feature meets non-functional requirements, achieves positive user feedback, and remains stable. One large source of initial user feedback will be obtained by conducting A/B testing. Chrome users can be shown a number of currently existing web pages and how they would look with the Night Mode applied, and ask which mode they prefer [9]. Along with determining user preference, the A/B testing can be used to get comparative data on eye strain between the regular and Night Mode pages, which can be used to quantify the success of the implementation.

To test whether the feature meets the proposed performance requirements, timing tests must be written to compare the average performance of Chrome when loading normal, non-Night Mode pages to the Night Mode version. A non-trivial impact on performance could occur, given that the feature parses and modifies pages before they are displayed. These tests should be run while the browser loads individual pages and multiple pages at once. Stress tests should also be performed on both of these test cases, with resource-expensive processes running in the background. Finally, unit and integration tests should be added to Chrome's existing suite test cases so that Night Mode can be tested for correctness in a more isolated fashion.

4.5 Impact on Stakeholders

We have identified 5 groups of important stakeholders for our proposed feature and associated them with NFRs that would be of high priority for them.

- **Users:** Performance, Stability, and Availability Users are concerned with the Performance, Stability, and Availability because they want to be able to use Night Mode when they need it most - between the hours of 4pm and 7am, and they want it to be reliable and perform well while they use it.
- **Chrome Developers:** Performance, Stability, Evolvability, Maintainability, and Testability Chrome Developers are interested in the Performance and Stability of the feature because they are the ones responsible for creating a feature that performs well for users and doesn't compromise the stability of the rest of the browser. They are also concerned with the feature's testability because they will be required to test the feature to make sure it meets performance objectives. Finally, they are concerned with evolvability and maintainability because, should the feature need updating in order to satisfy new user requirements, they will be responsible for evolving the feature to comply.
- **First and Third Party Advertisers:** Stability and Availability First and Third Party Advertisers are concerned with Stability and Availability because they are paying to advertise on Google Chrome, and if the new feature compromises Chrome's stability, their ad revenues might fall as less people see their advertisements when Chrome is down. Furthermore, if they've designed Night Mode specific ads, they would want Night Mode to be available during the guaranteed available period of 4pm to 7am to maximize views on their advertisement.

[9] Patel, N. (2018). A Beginner's Guide To AB Testing: An Introduction. [online] Neil Patel. Available at: https://neilpatel.com/blog/ab-testing-introduction/ [Accessed 29 Nov. 2018].

- **Other Website Developers:** Stability Other Website Developers would be concerned with Stability because they dont want Night Mode to reduce traffic to their site. The feature has to be stable to satisfy these developers.
- **Google Shareholders, Investors, and Executives:** Performance and Evolvability Google Investors and Executives hold a stake in the feature because should the addition of Night Mode compromise Chrome's performance in any way, users may move away from using Chrome as a primary browser which would have negative impacts on Google's bottom line. Evolvability is also important for these stakeholders because, in order to attract the largest active user base, Chrome needs to update frequently to stay competitive with other browser and in-line with industry best practices.

Below, we outline the pros and cons of Night Mode to each group of stakeholders.
Pros:

Benefits to users who are in favour of Night Mode:Dark themed applications are widely favoured by programmers and other professionals to reduce eye strain and allow longer periods of concentration Can be extended to other users who wish for more productive workflow Unique feature to the browser will attract more users. Thereby more ad revenue from increased browsing traffic which is important for Google executives.

Cons:

Would be challenging and time consuming for Chrome Developers to implement a refined implementation of Night ModeChrome has lots of source code written by many teams of developers, so the implementation would require a team of experts with a good understanding of how the feature would impact other areas of architecture. Finally, advertising and marketing design choices are carefully chosen to influence specific aspects of human psychology An active Night Mode may cause a disruption in the delicate balance of the user's perception of the interface and change an ad's impact on a user.

5 Feature's Impact on Chrome

5.1 Impact on High-Level Architecture

The impacted subsystems from our revised conceptual architecture are the UI, Browser and Renderer. The architecture style remains a combination of object-oriented and implicit invocation since no modules or dependencies were added or removed. The modifications to the three affected subsystems are discussed below.

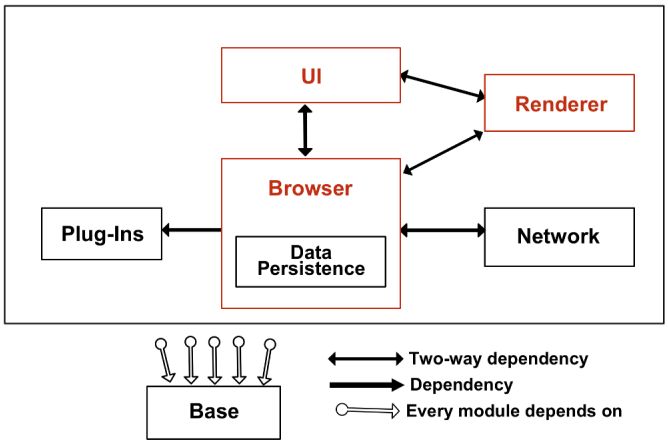

Figure 2: High-level impacted subsystems

5.2 Impact on Low-Level Architecture

5.2.1 UI

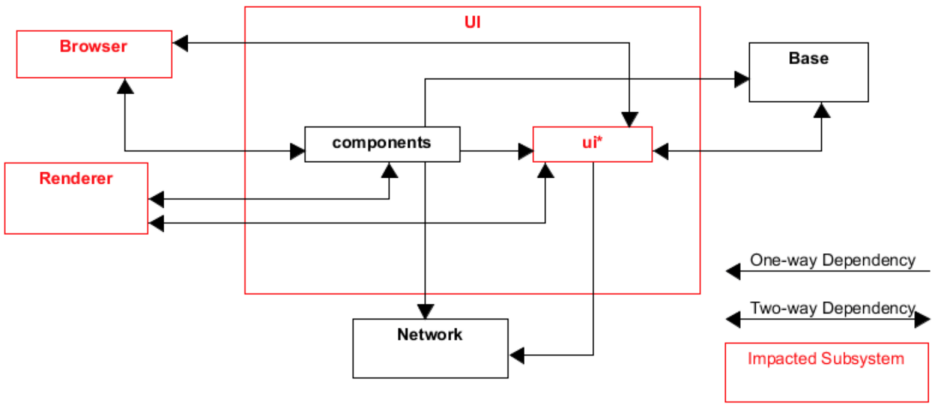

Figure 3: UI impacted subsystems

The UI module is modified slightly to implement the Night Mode feature, and these modifications are done in two files. The first file is **app_menu_model.cc**, located in the toolbar sub folder of the ui folder. This file contains code that is responsible for generating the menu when the "More Options" button (the three dots at the top right of the UI) is clicked. This file must be modified to add a label for toggling the Night Mode feature. Additionally, a command_id must be assigned to the label, so that the label can be identified and the label listener in the browser engine can execute the corresponding method.

The other file that is modified is browser_frame.cc, located in one of the views sub folders in the ui folder. This file contains the code that checks if the **Profile::INCOGNITO_PROFILE** flag is set when returning the Dark Aura (the library responsible for the UI) theme. If that incognito profile is set, an instance of NativeThemeDarkAura is returned and applied to the UI skin for Night Mode.

5.2.2 Browser

The Browser subsystem is impacted because we have to add the Night Mode profile to the set of Profile Flags and make a check for whether Night Mode is toggled on or off to provide the user with Night Mode functionality. The impacted sub-subsystems in the browser are chrome and components, as depicted in Figure 4.

Impacted directory: Browser/chrome

- **app_menu.cc** : contains listeners for the app_menu_model.cc call from the UI when the user selects "New Night Mode Window" from the menu dropdown on the UI toolbar. It changes the UI colouring to Native Theme Dark Aura.
- **browser_prefs.cc** : contains build flag for user preferences under a given profile. We would have to add a build flag for Night Mode in this file.
- **Night_Mode_prefs.cc and Night_Mode_policy_handler.cc** files would have to be created to so a User's Google Account preferences and parental controls can be checked to ensure the user is allowed to enable Night Mode before the change is implemented.
- **new_night_mode_tab.css and new_night_mode_tab.js** : would have to be added to provide web page formatting for the opening of new Night Mode tabs and windows. These css and js files exist both for regular browsing mode and Incognito mode, so we would need to specify actions for Night Mode in the same way.

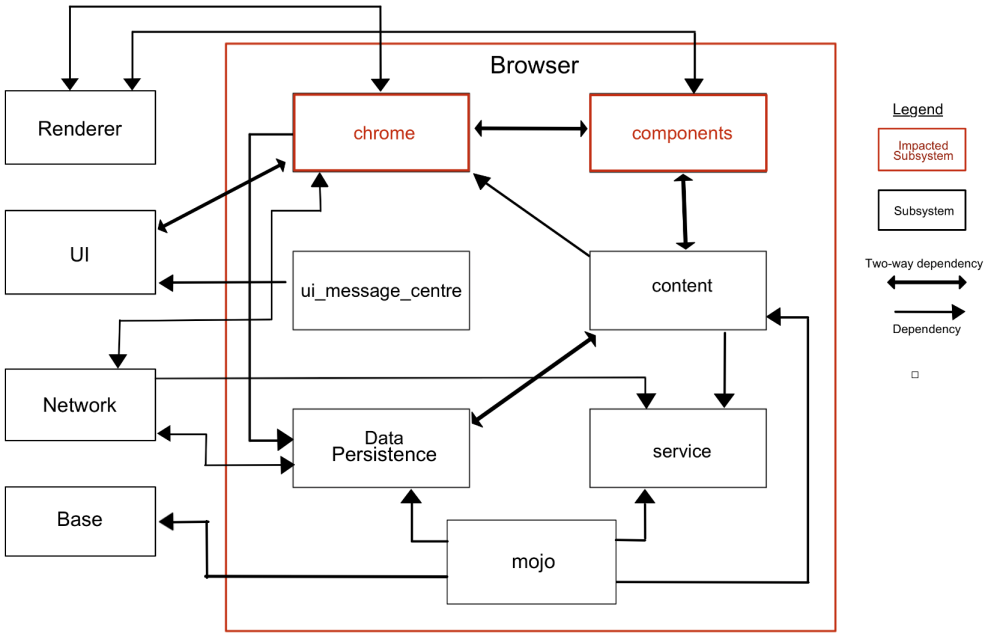

Figure 4: Browser impacted subsystems

- **pref_service_nightmode_whitelist** file would be created in order to determine which preferences to modify or keep consistent between Night Mode and standard Chrome browsing.
- **profile_manager.cc** is an existing file that is used to create and initialize new profiles or delete existing ones. We would have to add Night Mode functionality here - if it's a new profile, it will be synced to the user's Google account and set as default.

Impacted directory: Browser/components

- **user_manager_base.cc** would be modified to add a Boolean check for the Night Mode profile flag in order to apply the feature to the user's browsing window.

5.2.3 Renderer

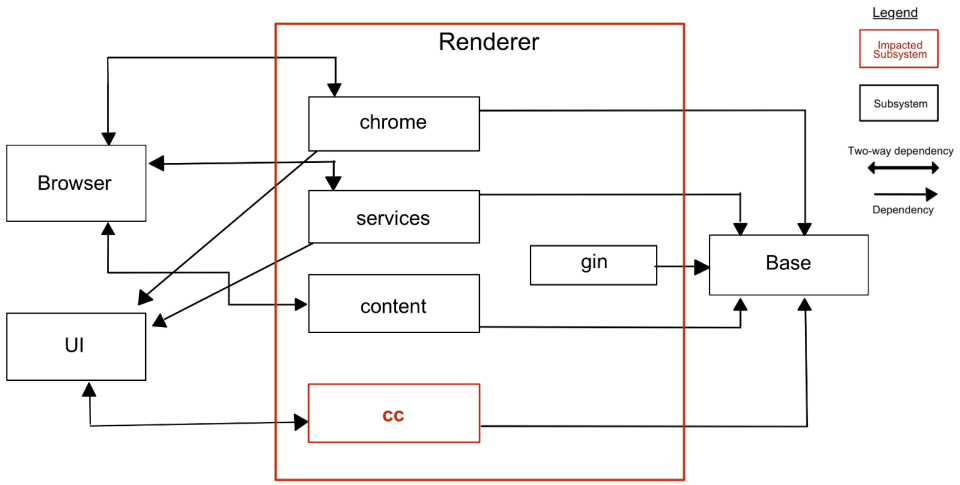

Figure 5: Renderer impacted subsystems

In the cc directory of our Renderer module, layer.cc is the file responsible for setting colours. Within the **SetBackgroundColor()** method, a custom method, **ReplaceColor()**, must be called to perform the dynamic colour replacement before the background colour is set. In the cc/paint directory exists files responsible for the colouring of text, fonts, and images, from where the **ReplaceColor()** method will also be called. Images are not going to be recoloured, as it was determined this would be very resource-intensive (as colours would have to be replaced pixel-by-pixel) and there are many images which should not be inverted, such as a picture of a dog. Determining which images should be replaced and which should not be replaced would be unreasonably difficult to implement for the scope of our proposed feature.

6 Sequence Diagrams for Feature Use Cases

6.1 User Toggles on Night Mode Feature

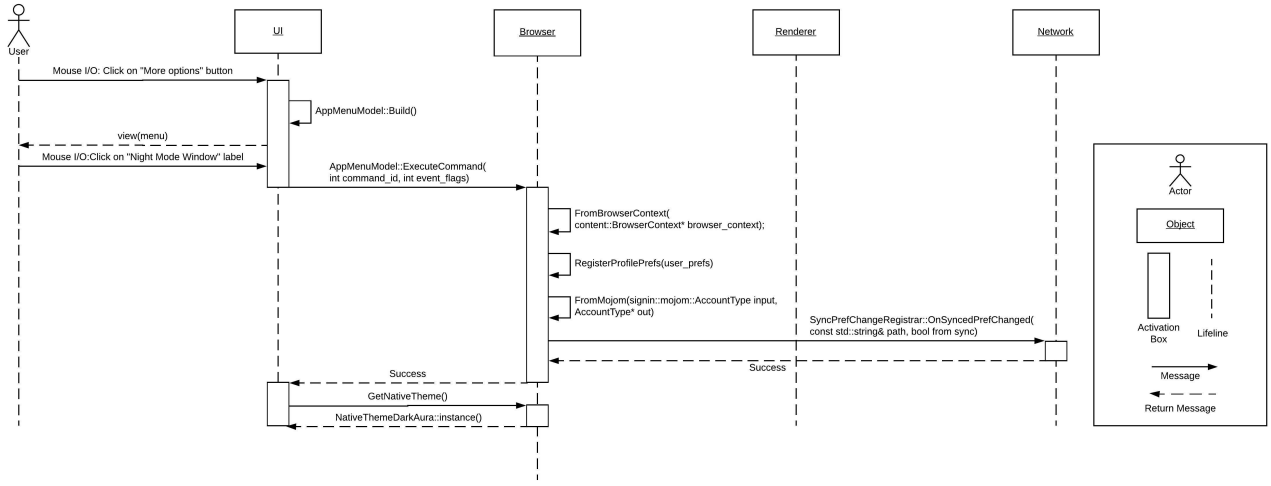

Figure 6: Use Case: User Toggles on Night Mode

In this use case, a user toggles on Night Mode, with the assumption that they're not on a web page, they are logged into a google account, and the macOS Dark Mode experimental feature is not turned on. First, the user clicks on the "More Options" menu. In the UI, AppMenuModel::Build() is called to construct the actual menu (this is modified by the feature's code to include an option to toggle Night Mode). Then, the user clicks on the "Night Mode Window" label, which causes the ExecuteCommand to be called, passing the label's corresponding command_id. The Browser then gets the browser context using FromBrowserContext(). The browser context contains some profile and user information. The profile is modified to set the Night Mode flag, and the Night Mode flag in the user's local setting are also changed. This information is saved locally using RegisterProfilePrefs(). FromMojom is used to retrieve whether or not the user is logged into a Google account. In this case they are, so SyncPrefChangeRegistrar() is called, which syncs the preference change across the user's account.

Finally, after some returns, GetNativeTheme() is called. This must be called explicitly as the profile flag was updated, but nothing specifically caused the UI to be updated (except on restart of Chrome). Thus, this method is called, and since the profile flag is now set, an instance of NativeThemeDarkAura is returned.

6.2 Render a page with Night Mode toggled on

This sequence diagram illustrates the use case of loading a web page after activating Night Mode. When the user clicks on a link to a distinct URL, the UI sends a request to the Browser Engine to load the web page. The URL associated to the link is the parameter used to specify the web page to be loaded and rendered. The Browser Engine then sends a request to the web site's server through Chrome's Network stack. After

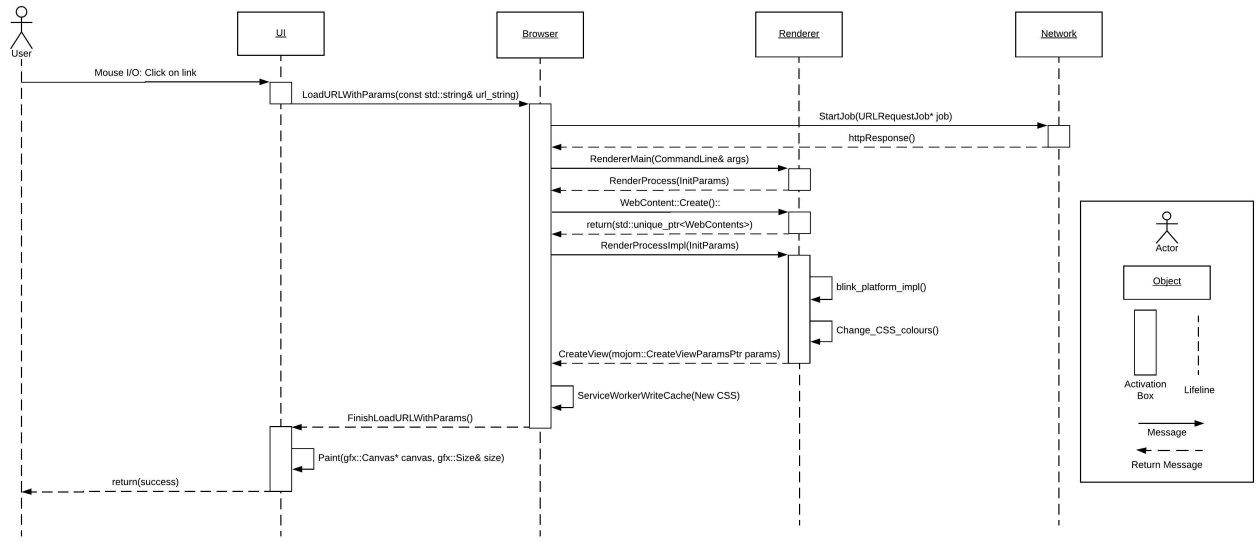

Figure 7: Use Case: Render a page with Night Mode toggled on

receiving an HTTP response from Network, The Browser Engine and Renderer subsystems begin to apply the Night Mode changes to the web page.

RendererMain serves as the main method when running renderer processes, so it must always be the first method called by Renderer for any use cases concerning rendering of web pages. After invoking this method, a RenderProcess method is called, with its initial parameters for web page rendering being sent to Browser Engine. A request is then sent to Renderer to create the contents of the web page. With the input parameters specified from earlier in the rendering pipeline, the Night Mode transformation is applied.

Change_CSS_colours performs the smart inversion of the web page's CSS, and parameters specifying the new UI view are returned to the Browser Engine. The Data Persistence module, within Browser, writes the CSS modifications to Chrome's cache. This is to prevent Night Mode rendering from occurring at the beginning of the sequence diagram whenever the user visits the URL. The URL and its display parameters are then returned to the UI, where the inverted CSS, along with the rest of the website, is drawn in Paint.

7 Concurrency

Concurrency is an important issue to Chrome that is not greatly affected by the Night Mode feature. Architecturally, there are no changes being made, and none Chrome's sandboxing, Site Isolation (maintaining that only tabs on the same site can be in the same process), or threading processes are touched. Most of the implementation would be carried out by slightly modifying currently existing concurrency mechanisms with the addition of flags and boolean checks. However, modifications to the files in Renderer which perform the hex colour inversion could have a negative impact on Chrome's concurrency model. For instance, if there is a critical bug in this code which triggers a crash, all rendered processes would also crash. On a more severe note, this could lead to complete unresponsiveness of the browser. While this isn't an issue directly violating sandboxing, it would go against Chrome's goal to isolate tab instances to limit crashes.

8 Risks and Limitations of our Implementation

8.1 Limitations

A limitation of our feature implementation is that Night Mode might not be compatible with every website. For instance, many news outlets and legacy websites do not use CSS formatting. Since our implementation relies on the inversion of website CSS, our feature would not be supported on these sites and the site would

appear unchanged. Furthermore, some sites may have implemented tight security measures which prevent the modification of their CSS, so our feature would not work on these sites either.

8.2 Risks

A risk of our feature implementation is the potential of bloating existing software in Chrome and compromising its performance. By modifying a page before it gets rendered, we are adding an additional work for the browser, which can inevitably cause Chrome to load web pages perceivably slower than it would without Night Mode. When expanding a code base, it's important to always focus on reducing coupling while increasing cohesion where possible to prevent bloating or over-engineering the system.

Our feature might pose security risks to Chrome. When changing or adding code to a complex software project, the risk of introducing vulnerabilities may accidentally leave the system exposed to attacks. The most invasive modifications from our implementation are done on a site's rendered CSS. Each tab opened in Chrome is securely sandboxed this limits the reach of a cyber attack. Therefore, a compromised tab would not affect the security of other tabs in the browsing session.

Another risk of our approach is that for certain use cases, such as for teaching purposes in education, it is not effective to invert the colours. According to an experiment conducted at the University of Missouri, it was concluded that colours with a greater contrast ratio generally lead to greater readability. And although white text on a black background shared the same contrasting ratio as black text on a white background, the unfamiliarity of the darker background resulted in a lower readability rating overall. [10].

Many UX designers specifically choose website colours to appeal to certain behavioural queues in human psychology. Altering the colours for these sites may potentially have the risk of affecting sales figures for businesses and Google may be legally liable for "tort of negligent interference" which is when a party damages the contractual relationship between other parties and results in economic harm.

9 Chrome Team Issues

Considering that we used the Chrome Team's project of Night Mode for MacOS as inspiration for our project, we thought that looking into their development issues would be reflective of problems we would encounter when implementing the feature. For reference, this Chrome developer discussion can be found at Issue # 850098 on the Chromium project's website.

When changing the color of the Night Mode window, a developer pointed out it was coloured too similarly to an Incognito Mode window. Not only would this be confusing for a developer if they wished to debug and test UI-specific issues, but it would also be a hindrance to the end-user. If a user is searching or performing an action in a non-Incognito window, this could lead to privacy issues, on top of confusing user experience. The colouring of Night Mode and Incognito windows should be unambiguous when a user glances at the colouring of the window, it should be immediately apparent what type of Chrome window they are using.

Another issue that the development team has encountered is the Night Mode colouring of extension icons. Currently, extension developers aren't allowed to provide Night Mode versions of their icon. The solution that the Chrome development team proposed for this would be allowing 3rd party developers to make an alternate Night Mode version of the icons and making Chrome able to detect the correct extension icon, depending on whether Night Mode is activated or not. The significant takeaway from this example is that even with our selected approach of a native, universal Night Mode, we still would need to reply on external developers to ensure that Night Mode changes would apply to the entire browsing experience.

[10] [http://lite.mst.edu/media/research/ctel/documents/LITE-2003-04.pdf]

10 Our Lessons Learned and Limitations

10.1 Lessons Learned

For this assignment, a different approach had to be taken from previous assignments. Previously, the team was responsible for documenting an already existing system, but this time, we had to take our previous learning and now apply it to an original idea. Thus, designing the feature both helped solidify our knowledge of Chrome's components and their interactions, as well as helping us understand how to handle the daunting task of contributing to a colossal open-source project such as Chromium.

Additionally, the team tried to consider how to implement the feature in a way that Google would be most likely to. The comparison made was to Google's Tag Manager (specifically its use for Google Analytics) [11]. From this, we believed that Google would likely implement an API that developers could add to their website to automatically integrate with the Night Mode feature. Though we concluded, as explained in the SAAM Analysis, that it would likely be difficult to get all sites to adopt this API, the team did learn about one of the methods Google uses to convince developers to use their product.

Finally, the team learned how to not just consider our own preferences and feelings when it comes to designing a feature, but also how to incorporate scientific studies and user feedback. Much research was done on the effect of blue light and light intensity when considering the justifications of the Night Mode feature and what it should really accomplish. User feedback such as A/B Testing was also researched, giving the team some insights into how to conduct surveys.

10.2 Limitations

The team also encountered some limitations when deriving our proposed enhancement. First and foremost, Chrome already has many features either implemented in the source code, via an extension, or currently in development. This made the derivation process longer than we anticipated as we worked to find a novel and useful idea for Chrome. Additionally, Chrome's source code is vast, so finding files to modify and function calls required for our use cases was time consuming and challenging.

11 Conclusion

After researching the effects of web browsing at night time, our team determined that Chrome does not have any existing features which address the health concerns of exposure to blue light at night time. Our team proposes a solution to this problem: developing a Night Mode feature for Chrome. To determine the best approach for implementing our enhancement, a SAAM analysis was conducted. The analysis defines 6 key Non-Functional Requirements: performance, stability, availability, maintainability, evolvability, and testability. Two implementation approaches were evaluated against the NFRs. Following the SAAM analysis, an optimal implementation approach was selected. Stakeholder impacts were assessed. Each stakeholder was assigned a set of the chosen NFRs which aligned best with their interests. We then look at the feature's impact on Chrome's architecture. The impacted subsystems are the UI, Browser, and Renderer. Two use cases relating to the feature were presented. The toggling on of Night Mode the loading of a webpage with Night Mode is toggled on. These use cases evaluated the impact of the feature on the architecture through sequence diagrams, and demonstrated how the feature could be integrated into the Chrome workflow. No major changes to Chrome's concurrency were identified. The risks and limitations of our implementation were identified and discussed. Despite the immense scale of Chrome, this assignment provided the team with valuable insight into the process for contributing a new feature to a large, open-source project. As a result, the team was able to better understand how to define the scope for a new feature, how to evaluate the best implementation approach, identify feature impacts, and develop sample use cases.

[11] Google Developers. (2018). Complete Web Upgrade: Using Google Tag Manager. [online] Available at: https://developers.google.com/analytics/devguides/collection/upgrade/reference/gtm [Accessed 26 Nov. 2018].

References

[1] Harvard Health. (2018). Blue light has a dark side - Harvard Health. [online] Available at: https://www.health.harvard.edu/staying-healthy/blue-light-has-a-dark-side [Accessed 27 Nov. 2018].

[2] Blask, D., Brainard, G., Gibbons, R., Lockley, S., Stevens, R. and Motta, M. (2012). Light Pollution: Adverse Health Effects of Nighttime Lighting. Report of the Council on Science and Public Health, 4, p.12.

[3] Rahman, S., Shapiro, C., Wang, F., Ainlay, H., Kazmi, S., Brown, T. and Casper, R. (2013). Effects of Filtering Visual Short Wavelengths During Nocturnal Shiftwork on Sleep and Performance. Chronobiology International, [online] 30(8), pp.951-962. Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3786545/ [Accessed 29 Nov. 2018].

[4] Smith, M. (2018). Turn off the lights. Dark Mode on Windows 10 and MacOS compared. [online] Digital Trends. Available at: https://www.digitaltrends.com/computing/dark-mode-macos-mojave-vs-windows-10/ [Accessed 30 Nov. 2018].

[5] https://bugs.chromium.org/p/chromium/issues/detail?id=850098

[6] Osmani, A. (2015). Tab Discarding in Chrome: A Memory-Saving Experiment. [online] Google Developers. Available at: https://developers.google.com/web/updates/2015/09/tab-discarding [Accessed 28 Nov. 2018].

[7] Kaufman, L. (2017). How To Sync Open Chrome Tabs View Them Across Devices. [online] trendblog.net. Available at: https://trendblog.net/sync-open-chrome-tabs-across-devices/ [Accessed 29 Nov. 2018].

[8] Megabyte. (2018). Automatically Clear Cache on a Site When Closing Chrome - Megabyte. [online] Available at: https://megabyterose.com/2018/01/clear-cache-on-exit/ [Accessed 29 Nov. 2018].

[9] Patel, N. (2018). A Beginner's Guide To AB Testing: An Introduction. [online] Neil Patel. Available at: https://neilpatel.com/blog/ab-testing-introduction/ [Accessed 29 Nov. 2018].

[10] http://lite.mst.edu/media/research/ctel/documents/LITE-2003-04.pdf

[11] https://www.jstor.org/stable/1072614?origin=crossrefseq=1metadata$_i nfo_t ab_c ontents$

[12] Google Developers. (2018). Complete Web Upgrade: Using Google Tag Manager. [online] Available at: https://developers.google.com/analytics/devguides/collection/upgrade/reference/gtm [Accessed 26 Nov. 2018].