**OVERALL PROJECT DESCRIPTION**

Web Annotator

Overview

* Web Annotator is a Chrome extension that enables users to add, edit, and manage notes directly on any webpage.

Features

* Highlight Text: Select text and apply highlights using right-click or keyboard shortcuts.
* Customize Highlights: Change highlight colours to suit your preferences.
* Add Notes: Attach sticky notes to highlighted text for additional context.
* Delete Highlights: Easily remove highlights and associated notes.
* Manage Highlights: Access and organize all highlighted texts and notes from the extension's popup.
* Search and Sort: Find specific highlights using the search bar, and sort them by style or date.
* Export Highlights: Download all highlighted texts in HTML, PDF, or text format.

How It Works

* Highlighting Text:
* Select the text and right-click to choose "Highlight Text" or use keyboard shortcuts.
* Select your desired highlight colour from the submenu.
* Managing Highlights:
  + Click on a highlighted text to reveal options for changing colour, adding notes, or deleting.
* Using the Extension:
  + Click the Web Annotator icon in the Chrome toolbar.
  + Access all highlights and notes from the popup window.
  + Utilize search and sorting options for efficient organization.
  + Download highlights in your preferred format.

Installation

* Download the extension files.
* Open Chrome and go to `chrome://extensions/`.
* Enable "Developer mode" in the top right corner.
* Click "Load unpacked" and select the extension directory.
* The Web Annotator icon will appear in the Chrome toolbar.

**File Descriptions**

* **manifest.json**

Defines the extension's metadata, permissions, background scripts, and content scripts.

* **background.js**

Handles the background logic for context menu actions and keyboard shortcuts.

* **content.js**

Injects the script into web pages to handle text highlighting and notes functionality.

* **popup.html**

Defines the structure of the extension's popup window.

* **popup.js**

Handles the logic for the popup window, including displaying highlights, search, sort, and export functionality.

* **styles.css**

Contains the styling for the extension's popup window.

* Manifest.json

Overview:

The `manifest.json` file provides essential metadata and configuration details for the "Web Annotator" Chrome extension.

Key Components:

* Manifest Version: Version 3.
* Name: Web Annotator.
* Version: 1.0.
* Description: Seamlessly add, edit, and manage notes directly on any webpage.
* Permissions: Required permissions include accessing active tabs, scripting, storage, context menus, and web navigation.
* Background: Utilizes a service worker specified in `background.js`.
* Action: Defines the default popup HTML file as `popup.html`.
* Content Scripts: Injects scripts (`html2pdf.js` and `contents.js`) and styles (`style.css`) into all URLs.
* Host Permissions: Allows access to all URLs.

Commands:

* `toggle\_highlight`: Keyboard shortcut to highlight selected text (default: Alt+Shift+H, Mac: Command+Shift+H).
* `next\_highlight`: Navigate to the next highlight (default: Alt+Shift+N, Mac: Command+Shift+N).
* `previous\_highlight`: Navigate to the previous highlight (default: Alt+Shift+P, Mac: Command+Shift+P).
* `clear\_highlights`: Delete the selected highlight (default: Alt+Shift+D, Mac: Command+Shift+D).

Usage:

* Install the extension by loading the unpacked directory in Chrome's extension settings.
* Use right-click or keyboard shortcuts to highlight text.
* Manage highlights via the extension's popup window.
* Access notes and highlights across webpages seamlessly.
* Background.js

Overview

The `background.js` file is a critical component of the Web Annotator Chrome extension. It handles background logic, such as creating context menus, responding to user actions, and implementing keyboard shortcuts.

Key Functions

* `chrome.runtime.onInstalled.addListener(async () => { ... })`
* Purpose: Initializes context menus upon installation of the extension.
* Description:
* Creates context menus for highlighting text with various color options.
* Utilizes the `tldLocales` object to define the context menu titles.
* Populates the context menus with color options defined in the `highlightColors` array.
* `chrome.contextMenus.onClicked.addListener((info, tab) => { ... })`
* Purpose: Handles clicks on context menu items.
* Description:
* Determines the colour selected by the user and triggers text highlighting accordingly.
* Finds the corresponding colour code from the `highlightColors` array based on the clicked menu item ID.
* Executes the `highlightTextredirect` function with the selected colour as an argument.
* `highlightTextredirect(colour) { ... }`
* Purpose: Redirects the highlighting process to the content script.
* Description:
* Receives the selected highlight colour as an argument.
* Executes the `highlightText` function in the content script with the provided colour.
* `chrome.commands.onCommand.addListener(function (command) { ... })`
* Purpose: Implements keyboard shortcuts for various actions.
* Description:
* Listens for commands triggered by keyboard shortcuts.
* Sends messages to content scripts to perform corresponding actions such as toggling, navigating, or clearing highlights.

Usage

* Installation: This file is part of the Web Annotator Chrome extension. Install the extension to enable its functionality in your browser.
* Initialization: Upon installation, the extension initializes context menus and keyboard shortcuts defined in this file.
* User Interaction: Users can right-click on selected text to access context menus for highlighting and choose from various colour options.
* Keyboard Shortcuts: Users can utilize predefined keyboard shortcuts to perform actions such as toggling, navigating, or clearing highlights.
* contents.js

Overview

The `contents.js` file is a crucial part of the Web Annotator Chrome extension, responsible for handling user interactions within the webpage, such as highlighting text, adding comments, and managing existing highlights.

Key Functions

* Event Listeners
* Purpose: Listens for user interactions like clicks and double-clicks.
* Description:
  + Detects when a user clicks on text to highlight it.
  + Manages the display of option buttons for modifying highlights.
  + Handles double-click events to initiate text highlighting.
* Highlighting Text
  + Purpose: Highlights selected text with customizable colours.
  + Description:
  + Dynamically creates and applies span elements to selected text for highlighting.
  + Calculates offsets and XPath for proper positioning and identification of highlighted text.
  + Saves highlight details including colour, URL, text content, and offsets to local storage.
* Restoring Highlights
  + Purpose: Restores previously saved highlights upon page load.
  + Description:
  + Retrieves highlight data from local storage and applies them to the corresponding text elements.
  + Handles various scenarios for properly restoring highlights within the page's structure.
* Keyboard Shortcuts
* Purpose: Implements keyboard shortcuts for quick actions.
* Description:
  + Listens for commands triggered by keyboard shortcuts such as toggling, navigating, or clearing highlights.
  + Executes corresponding functions to perform the desired actions based on user input.

Usage

* Installation: This file is part of the Web Annotator Chrome extension. Install the extension to enable its functionality in your browser.
* Interaction: Interact with the webpage by selecting text to highlight, adding comments, or managing existing highlights using provided option buttons.
* Keyboard Shortcuts: Utilize predefined keyboard shortcuts to perform actions like toggling, navigating, or clearing highlights for enhanced user experience.
* popup.html

Overview

The `popup.html` file serves as the structure for the popup window of the Web Annotator Chrome extension. It provides users with a user interface to manage highlights, search for specific highlights, and sort them based on different criteria.

Key Components

* Navbar
* Purpose: Houses essential controls and options for managing highlights.
* Description:
  + Contains elements for downloading highlights as a PDF and a sticky search bar for searching and sorting highlights.
* Search Bar
  + Purpose: Allows users to search for specific highlights by keywords.
  + Description:
  + Includes an input field where users can enter keywords to search for relevant highlights.
  + Provides real-time filtering of highlights based on the entered keywords.
* Sorting Options
  + Purpose: Enables users to sort highlights based on different criteria.
  + Description:
  + Offers a dropdown menu with options to sort highlights by date or style.
  + Allows users to customize the display order of highlights for better organization.

Usage

* Opening the Popup: Click on the extension icon in the browser toolbar to open the popup window.
* Search and Filter: Use the search bar to enter keywords and filter highlights based on the entered text.
* Sorting Highlights: Select a sorting option from the dropdown menu to rearrange highlights based on date or style.

Dependencies

* html2pdf.js: JavaScript library for converting HTML content to PDF format.
* popup.js: JavaScript file containing logic for the popup window.
* popup.css: CSS file defining styles for the popup window.
* popup.css

Overview

The `popup.css` file contains styles for the popup window of the Web Annotator Chrome extension. It defines the visual appearance and layout of elements within the popup, including the navbar, search bar, highlight display, and buttons.

Key Styles

* General Styles
* Purpose: Provides basic styling for all elements, ensuring consistent padding, margin, and box-sizing.
* Description:
* Sets a subtle background colour for the entire page.
* Defines a keyframe animation for twinkling background colours.
* Body Styles
* Purpose: Styles specific to the body element.
* Description:
* Allows scrolling within the popup window.
* Specifies the font family and light background colour for better readability.
* Scrollbar Styles
* Purpose: Customizes scrollbar appearance.
* Description:
* Hides scrollbars for a cleaner interface using `::-webkit-scrollbar`.
* Highlighted Text Styles
* Purpose: Styles for displaying highlighted text.
* Description:
* Sets width, margin, padding, and background color for the highlighted text container.
* Defines styles for individual spans within the highlighted text.
* Button Styles
* Purpose: Defines styles for buttons.
* Description:
  + Specifies width, height, background color, border radius, cursor, and transition effects for buttons.
  + Adjusts styles on hover to provide visual feedback.
* Navbar Styles
* Purpose: Styles for the navigation bar.
* Description:
  + Sets background color, position, width, z-index, padding, and box-shadow for the navbar.
  + Defines animation for the twinkling effect (optional).

Usage

* Apply the provided CSS styles to the HTML elements within the popup window to achieve a consistent and visually appealing interface.
* Customize the styles as needed to match the design requirements of your Chrome extension.

Dependencies

* popup.html: HTML structure of the popup window.
* popup.js: JavaScript logic for the popup window.
* popup.js

Overview

The `popup.js` file contains JavaScript code for the Web Annotator Chrome extension's popup window. It handles interactions with the DOM, such as displaying highlighted text, sorting elements, searching, and copying content to the clipboard.

Key Functions

* `setDOMInfo(info)`
* Purpose: Updates the DOM with highlighted text information.
* Description:
  + Clears the `Dates` array.
  + Iterates through the provided `info` array to create div elements for each highlighted text.
  + Sets background color, text color, and other styles based on the provided information.
  + Appends buttons, paragraphs, and spans to each div.
  + Attaches event listeners to buttons and input fields.
* `attachEventListeners()`
  + Purpose: Attaches event listeners to buttons and input fields.
  + Description:
  + Adds event listeners to buttons for copying text to the clipboard.
  + Adds event listeners to input fields for searching and sorting.
* `popup()`
  + Purpose: Handles the popup actions, such as downloading content in different formats.
  + Description:
  + Prompts the user to choose a download format (txt, html, or pdf).
  + Collects content based on the chosen format and initiates the download process.
* `searchDivs()`
  + Purpose: Filters and displays highlighted text based on the search keyword.
  + Description:
  + Retrieves the search keyword from the input field.
  + Filters div elements based on whether their text content contains the keyword.
  + Shows or hides div elements accordingly.
* `sortDivs()`
  + Purpose: Sorts highlighted text divs based on the selected sorting criteria.
  + Description:
  + Retrieves the selected sorting criteria (date or style).
  + Sorts div elements based on their background color (for style) or resets the DOM (for date).

Usage

* Include this JavaScript file in your Chrome extension project to enable the functionality of the popup window.
* Customize the functions as needed to fit your specific requirements or extend the functionality of the extension.

Dependencies

* popup.html: HTML structure of the popup window.
* popup.css: Stylesheet for the popup window.
* style.css

Overview

The `style.css` file contains CSS styling rules for the Web Annotator Chrome extension's popup window. It defines the appearance and behavior of various elements within the popup interface.

Key Styles

* `.active`
* Purpose: Defines the styling for an active element.
* Description:
  + Adds a dashed black border to visually indicate the active state.
* `.main`
  + Purpose: Styles the main container element.
  + Description:
  + Sets the height, width, and background color.
  + Arranges child elements using flexbox with space-evenly distribution.
  + Adds padding and border-radius for visual appeal.
  + Applies cursor pointer for interaction.
* `.main2`
  + Purpose: Styles an alternative main container element.
  + Description:
  + Similar to `.main` but with different dimensions and alignment for child elements.
* `.hover-option-button`
  + Purpose: Styles a hover option button.
  + Description:
  + Sets dimensions and border-radius to create a circular button.
  + Defines background color for the button.
  + Applies hover effect to scale up the button when hovered.
* `.hover-option-button-hover`
* Purpose: Styles another hover option button with a different hover effect.
* Description:
  + Similar to `.hover-option-button` but with a different hover effect.
* `.hover-textarea`
  + Purpose: Styles a hoverable textarea element.
  + Description:
  + Sets dimensions, border, and box-shadow properties for the textarea.
  + Defines a z-index to ensure it appears above other elements when hovered.

Usage

* Include this CSS file in your Chrome extension project to apply the specified styles to the popup window.
* Customize the styles as needed to achieve the desired appearance and behavior of the interface elements.

Dependencies

* popup.html: HTML structure of the popup window.
* popup.js: JavaScript functionality for the popup window.