

Universidad Pontificia de Salamanca

Facultad de Informática

Grado en Informática

Trabajo Fin de Grado

**Desarrollo de una extensión de navegador para adaptación**

**de contenidos web a líneas Braille**

Gonzalo Sánchez Sánchez

Dr. Alberto Pedrero Esteban

Director

Salamanca, Julio de 2024

Resumen

En la era digital, la accesibilidad web es crucial para la inclusión de todas las personas, independientemente de sus capacidades. Este trabajo presenta el desarrollo de una extensión para el navegador Chrome que adapta el contenido de las páginas web al ancho máximo soportado por una línea braille.

La extensión permite a los usuarios de dispositivos braille leer correctamente el contenido web, mejorando su accesibilidad y usabilidad. Para el desarrollo de la extensión, se han utilizado tecnologías como HTML, CSS y JavaScript, además de la API de Chrome Extensions. El proceso de desarrollo incluyó el análisis de requisitos, el diseño, la implementación y la evaluación mediante pruebas con usuarios. Los resultados muestran una mejora significativa en la experiencia de lectura para los usuarios de líneas braille. Este proyecto no solo mejora la accesibilidad, sino que también fomenta una mayor inclusión digital.

En el futuro, se propone ampliar la compatibilidad de la extensión a otros navegadores y dispositivos móviles, así como añadir funciones adicionales y realizar evaluaciones continuas para mejorar la herramienta. Los resultados demuestran la viabilidad y la importancia de adaptar el contenido web para dispositivos braille.

Abstract

In the digital era, web accessibility is crucial for the inclusion of all people, regardless of their abilities. This work presents the development of a Chrome extension that adapts web page content to the maximum width supported by a braille display.

The extension allows braille device users to correctly read web content, enhancing accessibility and usability. For the development of the extension, technologies such as HTML, CSS, and JavaScript were used, along with the Chrome Extensions API. The development process included requirements analysis, design, implementation, and evaluation through user testing. Results show a significant improvement in the reading experience for braille display users. This project not only improves accessibility but also promotes greater digital inclusion.

Future proposals include extending the extension's compatibility to other browsers and mobile devices, adding additional features, and conducting continuous evaluations to enhance the tool. The results demonstrate the feasibility and importance of adapting web content for braille devices.

Descriptores

Accesibilidad web, Línea braille, Extensión Chrome, Adaptación de contenido, Inclusión digital.

Índice

[1. Introducción y antecedentes 1](#_Toc170651620)

[1. Introducción 1](#_Toc170651621)

[1.1.1 Objetivos 1](#_Toc170651622)

[2. Antecedentes 1](#_Toc170651623)

[1.2.1 Estudio del estado del arte 2](#_Toc170651624)

[2. Desarrollo de la extensión 3](#_Toc170651625)

[2.1 Especificación de requisitos 3](#_Toc170651626)

[2.1.1 Funcionalidades 3](#_Toc170651627)

[2.1.2 Interfaces 3](#_Toc170651628)

[2.2 Desarrollo de la extensión 4](#_Toc170651629)

[2.2.1 Tecnologías utilizadas 4](#_Toc170651630)

[2.2.2 Metodología de desarrollo 4](#_Toc170651631)

[2.2.3 Desafíos encontrados 4](#_Toc170651632)

[3. Resultados y conclusiones 7](#_Toc170651633)

[3.1 Resultados y conclusiones 7](#_Toc170651634)

[3.1.1 Resultados obtenidos 7](#_Toc170651635)

[3.1.2 Conclusiones 8](#_Toc170651636)

[3.2 Líneas Futuras 9](#_Toc170651637)

Índice de figuras

[Ilustración 1 - Interfaz de usuario de la extensión 6](file:////Users/Gonzalo/Desktop/IngenieriaInformatica/CUARTO/SEGUNDO%20CUATRI/TFG/TFG/Memoria%20del%20Proyecto.docx#_Toc170664040)

[Ilustración 2 - Prueba con braille computerizado 7](#_Toc170664041)

[Ilustración 3 - Prueba con braille integral 7](#_Toc170664042)

[Ilustración 4 - Página de muestra de Wikipedia 10](file:////Users/Gonzalo/Desktop/IngenieriaInformatica/CUARTO/SEGUNDO%20CUATRI/TFG/TFG/Memoria%20del%20Proyecto.docx#_Toc170664043)

Índice de tablas

# Introducción y antecedentes

## Introducción

En la era digital actual, la accesibilidad web es crucial para garantizar la inclusión de todas las personas, independientemente de sus capacidades físicas. Este trabajo se centra en el desarrollo de una extensión para el navegador Chrome, diseñada para adaptar el contenido de las páginas web al ancho de una línea braille. Este desarrollo surge de la necesidad de mejorar la experiencia de navegación para los usuarios de dispositivos braille, permitiéndoles acceder y leer el contenido web de manera eficiente.

### Objetivos

* Desarrollar una extensión para Chrome que adapte el contenido web al ancho de una línea braille.
* Mejorar la accesibilidad y usabilidad de las páginas web para los usuarios de dispositivos braille.
* Evaluar la efectividad de la extensión mediante pruebas con usuarios.

## Antecedentes

El sistema braille, creado por Louis Braille en el siglo XIX, ha sido fundamental para la alfabetización y la educación de las personas ciegas. Con el avance de la tecnología, las líneas braille electrónicas han permitido a estos usuarios interactuar con dispositivos digitales. Sin embargo, la mayoría de las páginas web no están optimizadas para su lectura en estas líneas, lo que presenta un desafío significativo para la accesibilidad web.

### Estudio del estado del arte

* **Líneas braille:** Dispositivos que permiten a las personas ciegas leer texto mediante un sistema de puntos en relieve que se actualizan dinámicamente.
* **Extensiones de navegador:** Herramientas que añaden funcionalidades específicas a los navegadores web, facilitando la personalización y mejorando la experiencia del usuario.
* **Accesibilidad web:** Conjunto de prácticas y tecnologías diseñadas para hacer que el contenido web sea accesible para todas las personas, incluyendo aquellas con discapacidades.

# Desarrollo de la extensión

A continuación, se aborda de manera detallada el proceso de creación de la extensión para el navegador Chrome, cuyo objetivo es adaptar el contenido de las páginas web al ancho máximo soportado por una línea braille. Este proceso se divide en dos partes fundamentales: la especificación de requisitos y el desarrollo de la extensión junto con los desafíos y dificultades encontrados durante el mismo.

Cabe destacar que se han desarrollado dos versiones, una de desarrollo y otra para el usuario final debido a que no se cuenta con el conocimiento para leer en braille ni con una línea braille para la realización de pruebas durante el desarrollo.

Para acceder a fragmentos concretos del código mencionados durante este capítulo consultar [Anexo B: Código Fuente](#_Anexo_B:_Código). El código fuente también se encuentra comentado para un mejor entendimiento del mismo.

## Especificación de requisitos

### Funcionalidades

* Detectar y adaptar el contenido de las páginas web al ancho máximo soportado por una línea braille.
* Proveer una interfaz amigable y fácil de usar para la configuración de preferencias del usuario.
* Compatibilidad con los principales lectores de pantalla y dispositivos braille.

### Interfaces

* **Interfaz de usuario:** Panel de configuración accesible desde el menú de extensiones de Chrome.
* **Interfaz técnica:** Módulo que interactúa con el DOM (Document Object Model) de las páginas web para adaptar el contenido según la configuración establecida en la interfaz de usuario.

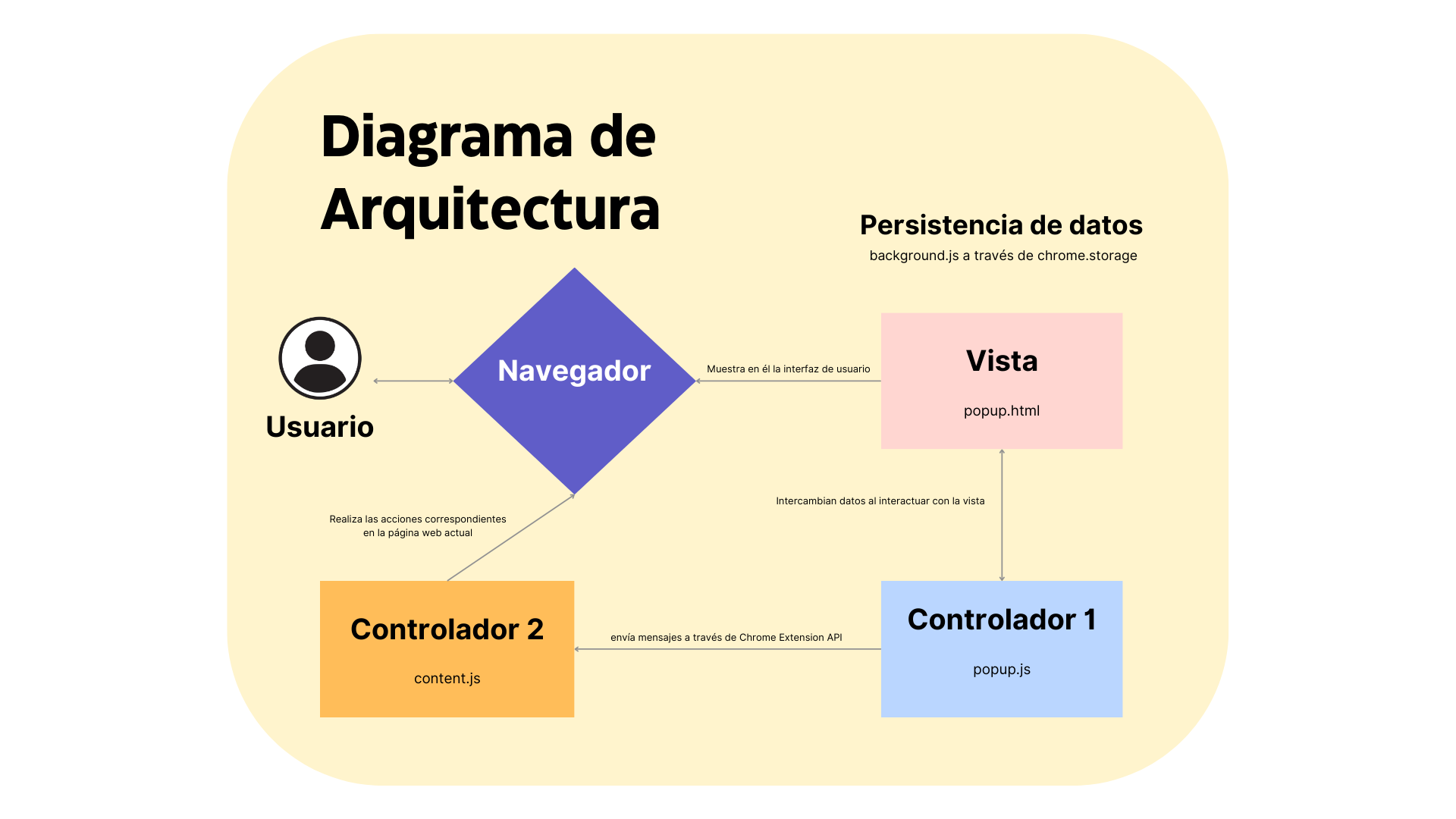
## Desarrollo de la extensión

### Tecnologías utilizadas

* **HTML, CSS y JavaScript:** Para la creación de la interfaz de usuario y la manipulación del DOM.
* **API de Chrome Extensions:** Para la integración y funcionamiento dentro del navegador Chrome.
* **Frameworks y Librerías:** Uso de librerías específicas para la accesibilidad y la manipulación de contenido web.

### Metodología de desarrollo

1. **Análisis de Requisitos:** Identificación de las necesidades del usuario y especificación de las funcionalidades.
2. **Diseño:** Creación de prototipos y diseño de la arquitectura de la extensión.



1. **Implementación:** Desarrollo del código y pruebas iniciales.

***Front-end* o interfaz de usuario:** La interfaz de usuario de la extensión ha sido diseñada para ser intuitiva y fácil de usar, especialmente pensando en las necesidades de los usuarios de dispositivos braille. La pantalla principal de la extensión permite a los usuarios configurar rápidamente sus preferencias de lectura y las especificaciones del dispositivo concreto que estén usando. A continuación, se describen los elementos principales de la interfaz:

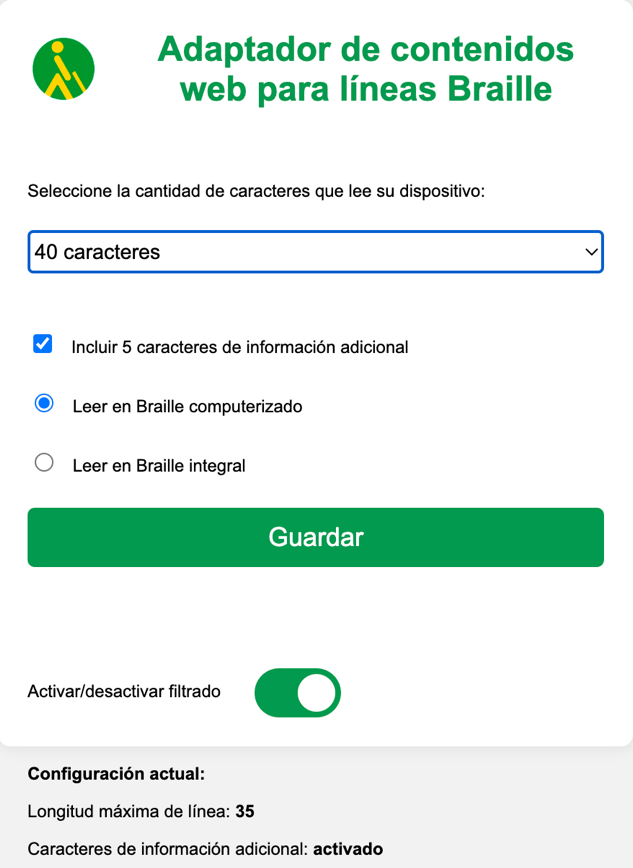
* Selector de caracteres: Un menú desplegable permite a los usuarios seleccionar la cantidad de caracteres que su dispositivo permite leer como máximo, tiene como opciones 20, 40 y 80 caracteres, que son los máximos de caracteres de las líneas braille existentes, como opción predefinida se encuentra la de 40 caracteres, ya que es el dispositivo más común.
* Opción de incluir caracteres de información adicional: Una casilla de verificación permite a los usuarios decidir si desean incluir 5 caracteres adicionales de información. Son caracteres comúnmente utilizados en este tipo de dispositivos que sirven para indicar el tipo de elemento sobre el que está.
* Modo de lectura: Los usuarios pueden elegir entre leer en "Braille computadorizado" o "Braille integral", proporcionando flexibilidad según sus preferencias.
* Botón de guardado: Un botón grande y visible para guardar las configuraciones seleccionadas.
* Interruptor de activación: Un interruptor para activar o desactivar el filtrado de contenido, proporcionando un control rápido sobre la funcionalidad de la extensión.
* Configuración actual: Un área informativa que muestra la configuración actual del usuario, incluyendo la longitud máxima de línea, el estado de los caracteres de información adicional y el modo de lectura seleccionado.

Ilustración 1 - Interfaz de usuario de la extensión

La interfaz de usuario está integrada por el archivo popup.html, el cual, como está indicado en el esquema de la arquitectura, se encarga de mostrar la vista de la extensión y en función de las interacciones del usuario con ella, envía una u otra información al script popup.js.

**Back-end o interfaz técnica:** Este módulo de la extensión está diseñado para interactuar con el DOM de las páginas web, filtrando y adaptando el contenido según las configuraciones definidas por el usuario.

De este modo, los datos introducidos por el usuario y que recibe el popup.js, son reenviados al script de fondo content.js y background.js. En estos se utiliza la API chrome.storage para almacenar configuraciones de manera persistente y mostrarlas.

El archivo popup.js además de contener la lógica para el envío de información a los scripts de fondo dependiendo de las interacciones que realice el usuario con la extensión, también se encarga de realizar acciones por defecto y sin ninguna interacción con la extensión por parte del usuario. Por ejemplo, cuando se carga una página, obtiene el estado actual de configuración y realiza el filtrado teniendo en cuenta la misma.

Por otro lado, el script content.js es más complejo y contiene casi toda la lógica de la extensión. A continuación, se detalla su funcionamiento:

**Funciones principales**

1. **chrome.runtime.onMessage.addListener()**:

Esta función utiliza la API chrome.runtime para manejar mensajes y registra un *“listener”* para manejar mensajes enviados desde el archivo popup.js o desde el entorno del navegador. En la versión de desarrollo esta función también se encarga de mostrar los datos de configuración y el contenido dividido del DOM respetando esa misma configuración por consola para poder comprobar su funcionamiento.

1. **dividirLineas(texto)**:

Esta función divide el texto en líneas según la longitud máxima definida en la configuración y el tipo de braille seleccionado.

1. **filtrarElemento(elemento)**:

Esta función que actúa como capa intermedia entre la función dividirLineas descrita anteriormente y la función filtrarContenido la cual es llamada cuando está activada la extensión es crucial para filtrar el contenido de texto dentro de los elementos del DOM de manera que se mantenga la estructura y usabilidad original de la página.

En este script también se incluyen las equivalencias de los caracteres naturales con el número de caracteres que ocupan tanto en braille integral como en braille computerizado y las respectivas funciones que realizan los conteos de caracteres de cada palabra que se les pasa.

1. **Pruebas y Evaluación:** Realización de pruebas con usuarios y ajustes basados en su *feedback*. También se realizaron pruebas a través de la creación de archivos html de prueba, los cuales se desplegaban y se probaba su correcto funcionamiento a través de la consola del navegador. A continuación se muestra un ejemplo:

En esta primera imagen, podemos observar cómo se nos muestra la configuración por consola y a continuación el texto de la web de prueba dividido en líneas respetándola:

Primera línea: 18 “A” + 17 “ “ = 35 caracteres en braille computerizado.



Ilustración 2 - Prueba con braille computerizado

En esta segunda, nos encontramos con el mismo caso, pero esta vez hacemos el conteo de caracteres en braille integral:  
Primera línea: 12 “A” + 11 “ “ = (12x2) + (11x1) 35 caracteres en braille integral.



Ilustración 3 - Prueba con braille integral

1. **Despliegue:** Publicación de la extensión en la *“Chrome Web Store”*.

### Desafíos encontrados

* **Compatibilidad:** Asegurar que la extensión funcione correctamente con diferentes versiones de Chrome y dispositivos braille.
* **Optimización:** Optimizar la extensión para que no afecte negativamente la velocidad de carga de las páginas web.
* **Contabilización Precisa de Caracteres en Braille Integral**: Dada la existencia de múltiples caracteres y normas en el sistema braille, es crucial que la extensión contemple las equivalencias correctas para cada carácter. A través del siguiente código se garantiza que se cumplen casi todas las normas a excepción de algunas que dependen de configuraciones adicionales ya que pueden transcribirse de varias formas y algunos caracteres especiales. Quedan así disminuidos los posibles errores a casos muy excepcionales y que considero despreciables.

const equivalenciasBrailleIntegral = {

' ': 1, 'a': 1, 'b': 1, 'c': 1, 'd': 1, 'e': 1, 'f': 1, 'g': 1, 'h': 1,

// ... (otros caracteres y equivalencias)

};

function contarCaracteresBrailleIntegral(texto) {

let contador = 0;

for (const char oftexto) {

if (equivalenciasBrailleIntegral[char] !== undefined) {

contador += equivalenciasBrailleIntegral[char];

} else {

contador += 3; // Por defecto si no lo encontramos lo contamos como 3 para evitar problemas

}

}

return contador;

}

Para la realización de la constante equivalenciasBrailleIntegral se utilizó como fuente de información el *“Documento técnico B 2: Signografía básica de las lenguas*

*cooficiales españolas* (Comisión Braille Española, 2023)*”*. Para la realización de la constante equivalenciasBrailleComputerizado se utilizó como fuente el *“Documento técnico V 1: Tabla ANSI española para braille computarizado* (Comisión Braille Española, 2023)*”*.

* **Cumplimiento de las Normas de Seguridad de Google Chrome**: La extensión debe adherirse estrictamente a las normas de seguridad y especificaciones técnicas establecidas por Google Chrome para garantizar su funcionalidad y seguridad.
* **Versatilidad en el Desarrollo**: Debido a que no se cuenta con el conocimiento para leer en braille ni con una línea braille para la realización de pruebas, se hace necesario desarrollar dos versiones de la extensión: una para usuarios y otra para desarrollo. A través de la versión de desarrollo se permite la verificación del funcionamiento mediante la consola del navegador o la modificación directa de la apariencia de las páginas.
* **Mantenimiento de la Apariencia y Usabilidad de las Páginas**: Para que la extensión opere sin alterar la apariencia y la usabilidad de las páginas web, permitiendo que otros usuarios que no utilicen línea braille puedan utilizar el mismo dispositivo con la extensión activada sin inconvenientes, se desarrolló una función intermedia entre

filtrarContenido() y dividirLineas() llamada filtrarElemento.

async function filtrarElemento(elemento) {

if (elemento.nodeType === Node.TEXT\_NODE) {

const texto = elemento.nodeValue;

const lineasFiltradas = await dividirLineas(texto);

elemento.nodeValue = lineasFiltradas.join('\n');

} else {

for (let i = 0; i < elemento.childNodes.length; i++) {

await filtrarElemento(elemento.childNodes[i]);

}

}

}

# Resultados y conclusiones

## Resultados y conclusiones

### Resultados obtenidos

* Implementación exitosa de la extensión con las funcionalidades especificadas.
* Mejora significativa en la experiencia de lectura para los usuarios de líneas braille. Para demostrar esta mejora, se cogió un fragmento del DOM de la Wikipedia.

<https://es.wikipedia.org/wiki/Universidad>

El fragmento del DOM correspondiente al texto seleccionado se tomó como muestra. A través de un script en Python se contaron el número de errores de lectura, es decir, el número de palabras que se quedarían divididas o incompletas al final de una línea, al utilizar una línea braille para leer ese fragmento y para cada longitud de línea braille que se puede configurar en la extensión.

Ilustración 4 - Página de muestra de Wikipedia

#### Script.py

def count\_word\_splits(text, line\_length):

words = text.split()

current\_length = 0

split\_count = 0

for word in words:

word\_length = len(word)

if current\_length + word\_length <= line\_length:

current\_length += word\_length + 1 # sumamos 1 por el espacio

else:

split\_count += 1

current\_length = word\_length + 1 # reiniciamos la longitud actual

return split\_count

def main():

# Texto de la wikipedia

text = """

Aquí iria el texto de la wikipedia

"""

# Longitudes de linea posibles

line\_lengths = [15, 20, 35, 40, 75, 80]

# Calcular el numero de palabras cortadas para cada longitud de linea

split\_counts = {length: count\_word\_splits(text, length) for length in line\_lengths}

# Resultados

print("Resultados de errores por longitud de línea:")

for length, count in split\_counts.items():

print(f"Longitud {length} caracteres: {count} palabras cortadas")

if \_\_name\_\_ == "\_\_main\_\_":

main()

#### Resultados script.py

|  |  |
| --- | --- |
| **Longitud máxima de caracteres por línea** | **Palabras cortadas totales en el fragmento** |
| 15 | 156 |
| 20 | 115 |
| 35 | 60 |
| 40 | 52 |
| 75 | 27 |
| 80 | 25 |

### Conclusiones

El desarrollo de esta extensión y la realización de pruebas demuestra la viabilidad y la importancia de adaptar el contenido web para dispositivos braille. Como se ha comprobado a través del script, el número de posibles errores de lectura que se pueden provocar al leer cualquier fragmento de texto, especialmente en líneas que soportan pequeñas longitudes de caracteres es muy elevado, por lo que la extensión facilita mucho la lectura cómoda de los usuarios y por tanto mejora la accesibilidad y fomenta una mayor inclusión digital.

## Líneas Futuras

* **Ampliación de Compatibilidad:** Adaptar la extensión para otros navegadores y para dispositivos móviles.
* **Funciones Adicionales:** Podría ser útil incorporar soporte para diferentes idiomas y configuraciones personalizadas más avanzadas. Un ejemplo de esto podría ser la inclusión en cada línea de etiquetas personalizadas descriptivas para indicar el tipo de elemento sobre el que se encuentra la línea
* **Evaluaciones continuas:** Realizar estudios de usabilidad periódicos y tener en cuenta la retroalimentación aportada por los usuarios para seguir mejorando la herramienta.

# Anexos

## Anexo A: Manual de Usuario

Aquí incluiré las instrucciones sobre cómo instalar y utilizar la extensión tanto con el código como desde la Chrome web store.

## Anexo B: Código Fuente

Enlace al repositorio con el código de la extensión listo para su despliegue.

<https://github.com/Gonzarrrion/TFG/tree/release>

## Anexo C: Pruebas y Evaluaciones

Resultados de las pruebas realizadas con usuarios y análisis de la efectividad de la extensión.

Bibliografía

Comisión Braille Española. (junio de 2023). *Documento técnico B 2: Signografía básica de las lenguas cooficiales españolas.* Obtenido de www.once.es: https://www.once.es/servicios-sociales/braille/comision-braille-espanola/documentos-tecnicos/documentos-tecnicos-relacionados-con-el-braille

Comisión Braille Española. (octubre de 2023). *Documento técnico V 1: Tabla ANSI española para braille computarizado.* Obtenido de www.once.es: https://www.once.es/servicios-sociales/braille/comision-braille-espanola/documentos-tecnicos/documentos-tecnicos-relacionados-con-otras-materias/documentos-tecnicos-relacionados-con-otras-materias