

# PRÁCTICA RA4: USABILIDAD Y ACCESIBILIDAD

Módulo: Desarrollo de Interfaces / Olga Marco Ugarte DAM2

## FASE 1: EVALUACIÓN DE LA INTERFAZ

### HEURÍSTICAS DE NIELSEN

#### **1. Falta de navegación por teclado**

- Problema: Los botones de la barra de herramientas tienen desactivada la propiedad de foco (focusTraversable=false). Es imposible navegar entre ellos tabulando.
- Heurística violada: Flexibilidad y eficiencia de uso.
- Impacto: Impide el uso eficiente a usuarios expertos y bloquea totalmente el uso a personas que no pueden utilizar un ratón.

#### **2. Invisibilidad de los comandos de voz**

- Problema: El sistema acepta comandos de voz específicos (Pro ejemplo "flores", "invierte"), pero no existe ninguna ayuda visual ni lista en pantalla que informe al usuario de qué palabras puede decir.
- Heurística violada: Saber que decir antes de dudar qué tienes que decir. (Falta un apartado donde ponga que cosas puedes decirle al detector de voz)
- Impacto: Obliga al usuario a memorizar comandos o adivinarlos, aumentando innecesariamente la carga cognitiva y frustrando al usuario nuevo.

#### **3. Falta de confirmación en acciones destructivas**

- Problema: El botón "Borrar" elimina todo el contenido del editor instantáneamente sin mostrar un mensaje de confirmación.
- Heurística violada: Prevención de errores.
- Impacto: Alto riesgo de pérdida de datos accidental por un clic erróneo, generando ansiedad en el usuario.

#### **4. Ausencia de función "Rehacer"**

- Problema: Existe la función "Deshacer" (El botón de atrás), pero si el usuario deshace cambios por error, no existe forma de recuperarlos (El botón de rehacer).
- Heurística violada: Control y libertad del usuario (User control and freedom).
- Impacto: El usuario se siente inseguro al usar el historial, sabiendo que "Atrás" es una acción unidireccional sin vuelta atrás.

#### **5. Etiquetas de botones confusas**

- Problema: Botones como "M/m" no sigue convenciones estándar de interfaces de texto.
- Heurística violada: Consistencia y estándares.
- Impacto: Confusión inicial sobre la funcionalidad de los controles. "M/m" no es un estándar reconocido universalmente para alternar mayúsculas.

### ACCESIBILIDAD WCAG 2.2

#### **6. Operabilidad con teclado (sin ratón)**

- Problema: El programa ahora mismo es imposible de usar si no tienes ratón. Se ha desactivado la opción de saltar de botón en botón con la tecla Tabulador, por lo que una persona con discapacidad motora en las manos no puede pulsar "Negrita" ni "Guardar", solo puede escribir texto.

## 7. Visibilidad del foco

- Problema: Como no se puede navegar con el teclado, tampoco se ve qué botón está seleccionado. Aunque se arreglara el punto anterior, si no se añade un borde azul o un cambio de color al pasar por encima con las teclas, el usuario estaría "ciego" sin saber qué botón va a pulsar.

## 8. Claridad de mensajes y estados

- Problema: Algunos botones tienen nombres confusos o muy cortos, como "M/m" para mayúsculas o solo "B" para negrita. Esto es un problema para todos, pero especialmente para las personas ciegas que usan lectores de pantalla, ya que el aparato les leerá "Botón B" y no sabrán qué hace. Deberían tener etiquetas ocultas explicativas.

### CHECKLIST RA4

- Cumplen razonablemente: Distribución general (Layout), Contraste de texto base (negro sobre blanco).
- No cumplen: Navegación por teclado, Feedback de errores, Ayuda y documentación (comandos de voz), Prevención de errores.

## FASE 2: PROPUESTAS DE MEJORA

### Habilitar la navegación por teclado (Solución a problemas 1 y 6)

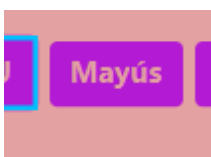
- Solución: Eliminar la instrucción `setFocusTraversable(false)` en el código de inicialización de todos los botones.
- Justificación: Permite que cualquier usuario navegue por la interfaz secuencialmente usando el tabulador, cumpliendo el criterio de operabilidad básico.

Al tabular pasa del botón "I" al botón "U" así consecutivamente.



### Estandarización de etiquetas e iconos (Solución a problemas 5 y 8)

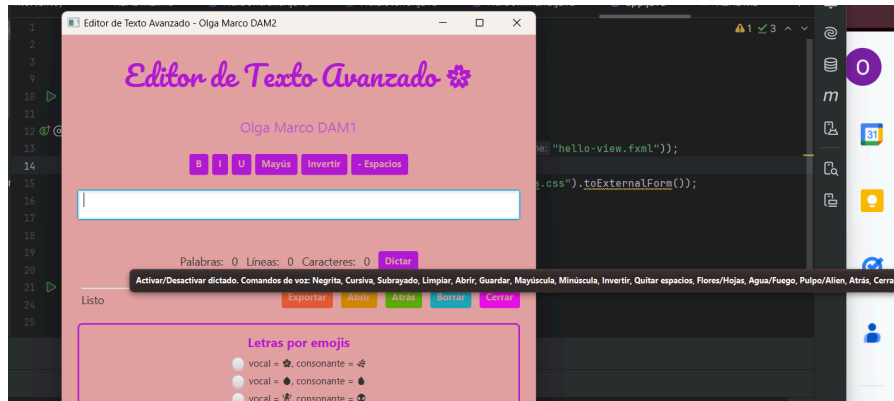
- Solución: Cambiar el texto "M/m" por "Mayús" o un icono estándar "Aa". Añadir Tooltips más descriptivos y propiedades de accesibilidad (`setAccessibleText`) a los botones de formato ("B" -> "Poner Negrita").
- Justificación: Facilita la comprensión visual y asegura que los productos de apoyo (lectores de pantalla) transmitan la información correcta.



### Ayuda visual para comandos de voz (Solución a problema 2)

- Solución: Añadir un Tooltip al botón "Dictar" o un pequeño botón de información (?) que despliegue la lista de palabras clave aceptadas ("negrita", "flores", "guardar").
- Justificación: Hace visibles las opciones del sistema, reduciendo la necesidad de memorización.

En la imagen se aprecia el tooltip con todas las palabras que puedes decir para cambiar el estado de tu texto.



### Indicador visual de foco (Solución a problema 7)

- Solución: Añadir reglas CSS en `estetica.css` para el pseudo-estado `:focused` en los botones, aplicando un borde azul claro o un cambio de fondo evidente.
- Justificación: Proporciona feedback inmediato al usuario de teclado sobre su posición en la interfaz.

Como se aprecia, al presionar un botón se rodea de azul, como en el ejemplo de la imagen con el botón de dictar.



## **FASE 4: REEVALUACIÓN / EXPLICACIÓN DEL IMPACTO**

Tras la implementación de las mejoras propuestas (habilitación del foco, diálogos de confirmación y etiquetas accesibles), la experiencia de usuario ha mejorado un montón :D

Lo más importante es que ahora sí se puede usar sin ratón, así que cumplimos con la accesibilidad WCAG, que antes era imposible.

También he arreglado el botón de borrar poniéndole una confirmación, así que ya no te cargas todo el trabajo por error.

En resumen, ahora el programa es más seguro y fácil de entender para cualquiera que lo use por primera vez.