📖 Memoria

> Documentación del proceso de trabajo. Comentar si se ha tenido problemas y cómo se ha buscado la solución. Si el resultado final no es totalmente el deseado por el aspirante, mencionar qué habría cambiado o qué opciones de mejora se han detectado.

📋 Desarrollo

▼ Video:

- Reproducción en pantalla completa: Uso de etiqueta básica <video> HTML5 y valores css estándar.
- Reproducción controlada mediante scroll: Uso de la librería Greensock y su plugin ScrollTrigger. Se ha optado por el uso de gsap.fromTo() para controlar el inicio y fin de la reproducción modificando el valor de currentTime del propio componente de video.

▼ Textos:

- Añadir maquetación con los textos en distintas secciones: Uso de etiqueta básica <section> para crear tres secciones correspondientes a los tres fragmentos proporcionados. Para la renderización de los textos, hago uso de la etiqueta <object> pasándole su origen mediante *data*. He encontrado, que en la nomenclatura de los svgs entregados, aparece el caracter “#”, el cual, he tenido que cambiar por guion medio (-) para solventar esta pequeña incidencia. Los svgs, también podrían haber sido renderizados haciendo uso de la etiqueta <img>.
- Animar desplazamiento de los textos: asignar simples dimensiones en su css.
- Animar opacidad de los textos: Uso de la librería Greensock y su plugin ScrollTrigger. Se ha optado por el uso de gsap.fromTo() para controlar el inicio y fin de la animación modificando el valor de opacity del propio texto.

También se hace uso de gsap.to(), cuando solo se necesita modificar el valor de la opacidad en una dirección.

▼ Símbolo de desplazamiento inferior:

- Añadir símbolo de desplazamiento para volver al inicio: Uso de la etiqueta <img> para el renderizado del símbolo, haciendo uso del content-type image/svg+xml.
- Añadir animación de parpadeo: Uso de la librería Greensock y su plugin ScrollTrigger. Se ha optado por el uso de gsap.to() el parpadeo, modificando el valor de opacity del propio texto. También se hace uso del método .yoyo() para generar una animación en ambos sentidos y del método .repeat(), con valor *-1*, para que se repita indefinidamente.
- Añadir animación de rotación 180° al llegar al final de la página: Uso de la librería Greensock y su plugin ScrollTrigger. Se ha optado por el uso de gsap.to() para controlar la rotación de la imagen.
- Añadir funcionalidad de desplazamiento al inicio de la página al hacer click: Se ha creado una funcion *scrollToTop()*, que hace uso del método window.scrollTo() y del parámetro *behavior* con valor *smooth*, para que el movimiento sea fluido.

▼ Maquetación responsive:

- Selección del origen del video en función de las distintas resoluciones: Inicialmente, planteaba hacer uso de @media-queries para modificar el valor de src del propio componente <video> HTML. Esto ha sido deprecado en la mayoría de navegadores, y no parece la implementación adecuada a largo plazo, ya que existen, para mejorar estos comportamientos, estándares responsive de video, como DASH o HLS. Para mantener simple la solución, se ha adoptado el uso de window.innerWidth para establecer al inicio, la fuente del video.
- Tamaños relativos de textos (svgs): Uso de unidades relativas en css, como *rem, %, vw* o *vh*.
- Adaptación de animaciones: Ajuste de los triggers de inicio y fin de cada animación, con el uso de las unidades relativas propias de ScrollTrigger, el plugin de gsap usado, tales como *top*, *bottom* o *%*, descritas en su documentación.

▼ Testing:

- Mobile: Pruebas en dispositivo físico, conectándolo mediante IP privada, en red local, sirviendo la web desde el mac usado para el desarrollo. También se han usado las propias funcionalidades del Inspector de Chrome, eligiendo distintos dispositivos móviles y validando su comportamiento.
- Desktop: Pruebas en el propio equipo, haciendo uso de los distintos navegadores instalados, como Chrome y Safari. También he realizado pruebas variando dinámicamente el viewport y en distintas pantallas físicas.

💡 Puntos de mejora

▼ Svgs:

- El tamaño y disposición del texto se podría mejorar en la renderización, generando los svgs de nuevo ampliando el número de lineas. Con esto, conseguiríamos ver más cantidad de texto en el mismo espacio horizontal.

▼ Animaciones:

- Aunque el uso de gsap.fromTo() y gsap.to() han resuelto la problemática planteada, me gustaría analizar el problema desde el punto de vista del uso de gsap.timeline(), ya que parece ser buena solución para animaciones especificas y acotadas como esta, donde hay muchos elementos involucrados en el mismo instante temporal.