

Fase 2 - Diseño: Crear el Guion y la maqueta para un OVI.- Entrega de la actividad

DISEÑOS DE SITIOS WEB

Integrante

Ángel Andrés Ortiz Meza
CC 1065842893

Grupo

301122A_761

Tutor

AVILA PEREZ MARIO LUIS

Universidad nacional abierta y a distancia (UNAD). Escuela de ciencias básicas tecnología e ingeniera
ingeniería de sistemas
Santa Marta 2020

ACTIVIDAD FASE DE PLANEACIÓN Y ANÁLISIS
CURSO DISEÑOS DE SITIOS WEB - COD. 301122
FORMATO GUION SITIO WEB DEL OVI

Diseñado Por: Director Curso

- 1. Objetivos del OVI** (describa mediante el registro de 1 objetivo general y tres específicos para que se construye este OVI)

Objetivo general:

Implementar a partir de los conocimientos adquiridos sobre el uso de herramientas específicas en el ámbito laboral para gestionar proyectos colaborativos de diseño y creación de contenidos web.

Objetivo específico 1:

- **Crear un repositorio y clonar en local y remoto con la ayuda de las herramientas GIT y GITHUB.**



Objetivo específico 2:

- **Identificar e implementar los componentes Estructurales de html y html5 (etiquetas, atributos, valores, HTML, HEAD, BODY, NAV - UL - OL – LI, SECTION, ASIDE, FOOTER, ARTICLE, inline, block, inline-block, top, right, left, bottom).**

Objetivo específico 3:

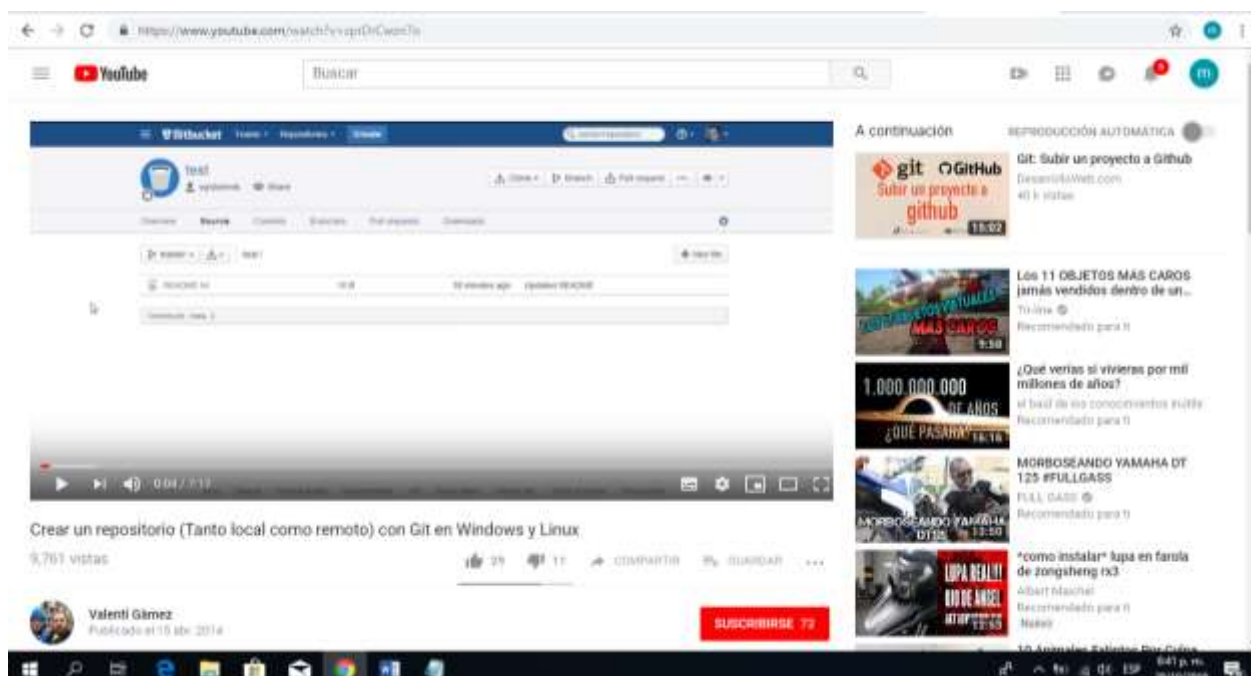
- **Identificar e implementar los componentes Estructurales CSS3 como estilos internos y Externos, Selectores, clases y Propiedades básicas.**

2. **Contenido informativo del OVI por secciones** (Replique el siguiente cuadro de acuerdo al número de secciones que vaya a crear en el OVI)

GIT y GITHUB:
2.1 Objetivo de la sección: (Registre a continuación el objetivo que tiene esta sección)
<ul style="list-style-type: none"> • Aprender a crear un repositorio y clonarlo en local y remoto con la ayuda de las herramientas GIT y GITHUB.
2.2 Recursos de consulta que usara en la sección: (coloque el nombre del material que usara para crear los contenidos de la sección y el enlace de descarga de los mismos sean estos Texto, Imágenes, Audios o Vídeos)
 https://git-scm.com/  GitHub https://github.com/github https://github.com/

Crear un repositorio (Tanto local como remoto) con Git en Windows y Linux:

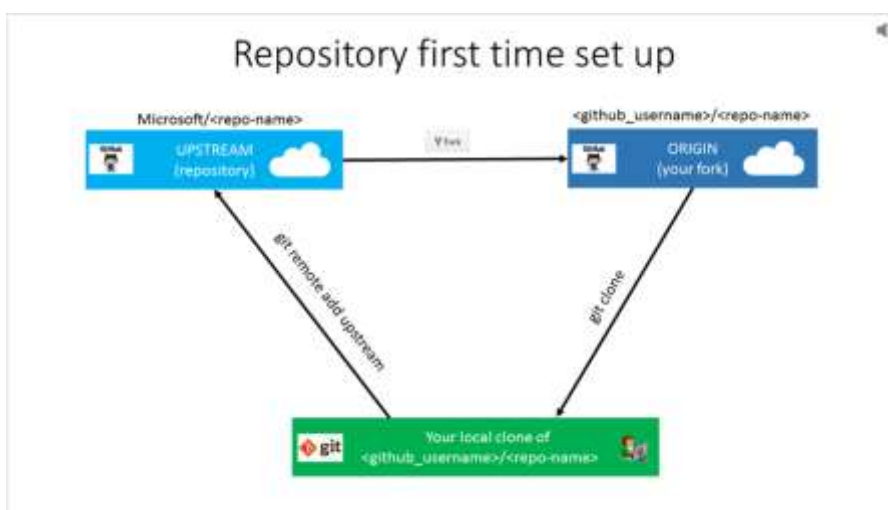
<https://www.youtube.com/watch?v=zprDrCwzn7o>



Configuración local del repositorio de Git para documentación:

<https://docs.microsoft.com/es-es/contribute/get-started-setup-local>

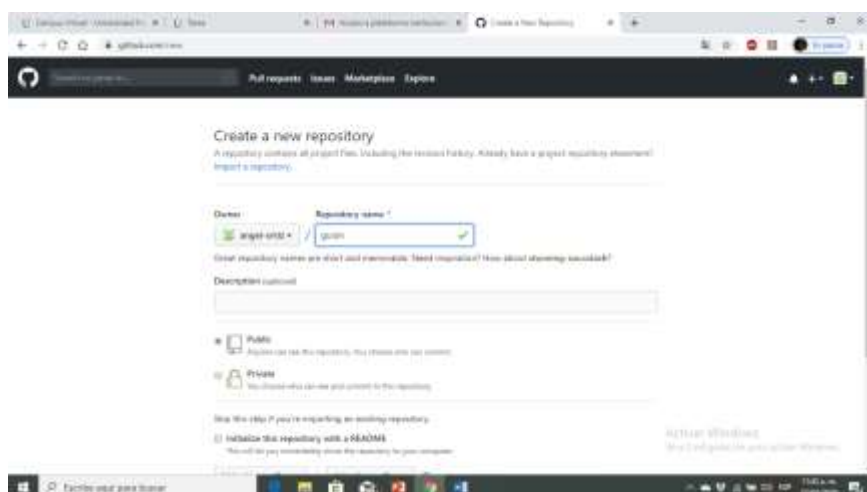
<https://docs.microsoft.com/es-es/contribute/media/git-and-github-initial-setup.png>



Crear el repositorio remoto en GitHub

<https://services.github.com/on-demand/github-desktop/es/crear-repo-remoto>

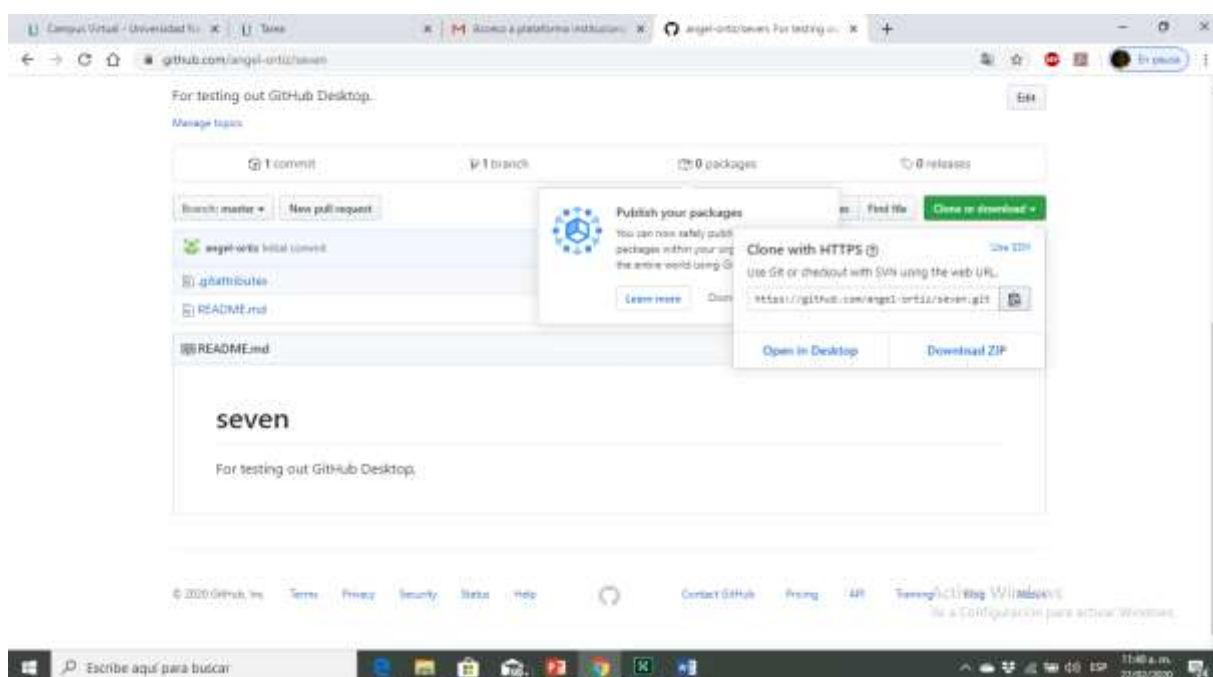
<https://services.github.com/on-demand/images/gifs/github-desktop/create-repo.gif>



Clonar el repositorio con GitHub Desktop

<https://services.github.com/on-demand/github-desktop/es/clonar-repositorio-github-desktop>

<https://services.github.com/on-demand/images/gifs/github-desktop/clone-repository-locally.gif>



GitHub. Repositorios

<http://www.mclibre.org/consultar/informatica/lecciones/github-repositorios.html>

Introducción a Git y Github

<https://desarrolloweb.com/articulos/introduccion-git-github.html>

2.3 Redacte un borrador del contenido de lectura en formato de texto que tendrá la sección: (Sea este la presentación de la sección, el contenido o ambos; redacte un borrador del texto que publicara como contenido en la sección coloque un subtítulo para identificar si corresponde a la presentación de la sección o el contenido de lectura de la sección)

Una introducción a los sistemas de control de versiones, Descripción general de lo que son Git y Github.

Los sistemas de control de versiones son programas que tienen como objetivo controlar los cambios en el desarrollo de cualquier tipo de software, permitiendo conocer el estado actual de un proyecto, los cambios que se le han realizado a cualquiera de sus piezas, las personas que intervinieron en ellos, etc.

El control de versiones es una de las tareas fundamentales para la administración de un proyecto de desarrollo de software en general. Surge de la necesidad de mantener y llevar control del código que vamos programando, conservando sus distintos estados. Es absolutamente necesario para el trabajo en equipo, pero resulta útil incluso a desarrolladores independientes.

Sobre Git

Git es un sistema de control de versiones distribuido. Relativamente nuevo que nos ofrece las mejores características en la actualidad, pero sin perder la sencillez. Git es multiplataforma, por lo que puedes usarlo y crear repositorios locales en todos los sistemas operativos más comunes, Windows, Linux o Mac. Existen multitud de GUIs (Graphical User Interface o Interfaz de Usuario Gráfica) para trabajar con Git a golpe de ratón, no obstante, para el aprendizaje se recomienda usarlo con línea de comandos, de modo que puedas dominar el sistema desde su base, en lugar de estar aprendiendo a usar un programa determinado.

Para usar Git debes instalarlo en tu sistema. Hay unas instrucciones distintas dependiendo de tu sistema operativo, pero en realidad es muy sencillo. La página oficial de descargas está en git-scm.com.

Sobre GitHub

Github `github.com` es un servicio para alojamiento de repositorios de software gestionados por el sistema de control de versiones Git. Por tanto, Git es algo más general que nos sirve para controlar el estado de un desarrollo a lo largo del tiempo, mientras que Github es algo más particular: un sitio web que usa Git para ofrecer a la comunidad de desarrolladores repositorios de software. En definitiva, Github es un sitio web pensado para hacer posible el compartir el código de una manera más fácil y al mismo tiempo darle popularidad a la herramienta de control de versiones en sí, que es Git.

Configuración local del repositorio de Git para documentación

Si es la primera vez que usa GitHub, vea el vídeo siguiente para obtener información general de carácter conceptual sobre el proceso de bifurcación y clonación:

Fundamentos de Git

Si sólo puedes leer un capítulo para empezar a trabajar con Git, es éste. Este capítulo cubre todos los comandos básicos que necesitas para hacer la gran mayoría de las cosas a las que vas a dedicar tu tiempo en Git. Al final del capítulo, deberías ser capaz de configurar e inicializar un repositorio, comenzar y detener el seguimiento de archivos, y preparar (stage) y confirmar (commit) cambios. También te enseñaremos a configurar Git para que ignore ciertos archivos y patrones, cómo deshacer errores rápida y fácilmente, cómo navegar por la historia de tu proyecto y ver cambios entre confirmaciones, y cómo enviar (push) y recibir (pull) de repositorios remotos.

Obteniendo un repositorio Git

Obteniendo un repositorio Git

Puedes obtener un proyecto Git de dos maneras. La primera toma un proyecto o directorio existente y lo importa en Git. La segunda clona un repositorio Git existente desde otro servidor.

Inicializando un repositorio en un directorio existente

Si estás empezando el seguimiento en Git de un proyecto existente, necesitas ir al directorio del proyecto y escribir:

```
$ git init
```

Esto crea un nuevo subdirectorio llamado `.git` que contiene todos los archivos necesarios del repositorio —un esqueleto de un repositorio Git. Todavía no hay nada en tu proyecto que esté bajo seguimiento.

Si deseas empezar a controlar versiones de archivos existentes (a diferencia de un directorio vacío), probablemente deberías comenzar el seguimiento de esos archivos y hacer una confirmación inicial. Puedes conseguirlo con unos pocos comandos `git add` para especificar qué archivos quieres controlar, seguidos de un `commit` para confirmar los cambios:

```
$ git add *.c
```

```
$ git add README
```

```
$ git commit -m 'versión inicial del proyecto'
```

Veremos lo que hacen estos comandos dentro de un minuto. En este momento, tienes un repositorio Git con archivos bajo seguimiento, y una confirmación inicial.

Clonando un repositorio existente

Si deseas obtener una copia de un repositorio Git existente —por ejemplo, un proyecto en el que te gustaría contribuir— el comando que necesitas es `Git clone`. Si estás familiarizado con otros sistemas de control de versiones como Subversión, verás que el comando es `clone` y no `checkout`. Es una distinción importante, ya que Git recibe una copia de casi todos los datos que tiene el servidor. Cada versión de cada archivo de la historia del proyecto es descargada cuando ejecutas `Git clone`. De hecho, si el disco de tu servidor se corrompe, puedes usar cualquiera de los clones en cualquiera de los clientes para devolver al servidor al estado en el que estaba cuando fue clonado (puede que pierdas algunos *hooks* del lado del servidor y demás, pero toda la información versionada estaría ahí —véase el Capítulo 4 para más detalles—). Puedes clonar un repositorio con `Git clone [url]`. Por ejemplo, si quieres clonar la librería Ruby llamada Grit, harías algo así:

```
$ git clone git://github.com/schacon/grit.git
```

Esto crea un directorio llamado "Grit", inicializa un directorio. Git en su interior, descarga toda la información de ese repositorio, y saca una copia de trabajo de la última versión. Si te metes en el nuevo directorio `Grit`, verás que están los archivos del proyecto, listos para ser utilizados. Si quieres clonar el repositorio a un directorio con otro nombre que no sea `grit`, puedes especificarlo con la siguiente opción de línea de comandos:

```
$ git clone git://github.com/schacon/grit.git mygrit
```

Ese comando hace lo mismo que el anterior, pero el directorio de destino se llamará `mygrit`.

Git te permite usar distintos protocolos de transferencia. El ejemplo anterior usa el protocolo `git://`, pero también te puedes encontrar con `http(s)://` o `usuario@servidor:/ruta.git`, que utiliza el protocolo de transferencia SSH.


```
# (use "git add <file>..." to include in what will be committed)
```

```
# README
```

```
nothing added to commit but untracked files present (use "git add" to track)
```

Puedes ver que tu nuevo archivo README aparece bajo la cabecera “Archivos sin seguimiento” (“Untracked files”) de la salida del comando. Sin seguimiento significa básicamente que Git ve un archivo que no estaba en la instantánea anterior; Git no empezará a incluirlo en las confirmaciones de tus instantáneas hasta que se lo indiques explícitamente. Lo hace para que no incluyas accidentalmente archivos binarios generados u otros archivos que no tenías intención de incluir. Sí que quieres incluir el README, así que vamos a iniciar el seguimiento del archivo.

Crear el repositorio remoto en GitHub

- En GitHub.com, crea un nuevo repositorio.
- Nombra tu repositorio TU-USUARIO.GITHUB.IO

El nombre de tu repositorio será parte del enlace a tu sitio web. Si utilizas la convención para el nombre descrita, tu sitio web se servirá en esa URL.

- El nombre de tu repositorio será parte del enlace a tu sitio web. Si utilizas la convención para el nombre descrita, tu sitio web se servirá en esa URL.
- Introduce una descripción para tu repositorio.
- Escoge visibilidad Public.

Recomendamos que crees un repositorio público. Los repositorios públicos son gratis. Incluso si seleccionas un repositorio privado, tu sitio web publicado será público.

- Recomendamos que crees un repositorio público. Los repositorios públicos son gratis. Incluso si seleccionas un repositorio privado, tu sitio web publicado será público.
- Selecciona Initialize this repository with a README.
- Haz clic en Create repository.

Clonar el repositorio con GitHub Desktop

- Entra en tu cuenta en GitHub.com y GitHub Desktop.
- En la parte derecha de la pantalla, haz clic en Clone or download.
- Haz clic en Open in Desktop. Esto abrirá GitHub Desktop.
- Selecciona dónde te gustaría guardarlo localmente en el campo Local Path.
- Haz clic en Clone.

Nombre de la sección que se creara en el OVI:

3.1 Objetivo de la sección: (Registre a continuación el objetivo que tiene esta sección)

- estructura básica de html y html5

3.2 Recursos de consulta que usara en la sección: (coloque el nombre del material que usara para crear los contenidos de la sección y el enlace de descarga de los mismos sean estos Texto, Imágenes, Audios o Vídeos)

Estructura básica de una página web

<https://www.hazunaweb.com/curso-de-html/estructura-basica-una-pagina-web/>

HTML5 y su estructura básica

<https://rolandocaldas.com/php/html5-estructura-basica>

Lista de etiquetas HTML

<http://www.mclibre.org/consultar/htmlcss/html/html-etiquetas.html>

3.3 Redacte un borrador del contenido de lectura en formato de texto que tendrá la sección: (Sea este la presentación de la sección, el contenido o ambos; redacte un borrador del texto que publicara como contenido en la sección coloque un subtítulo para identificar si corresponde a la presentación de la sección o el contenido de lectura de la sección)

estructura básica de html

HTML (HyperText Markup Language) es el primer lenguaje que una persona debe conocer si desea comenzar a realizar páginas web. HTML no es un lenguaje de programación, sino un lenguaje descriptivo, una serie de etiquetas que el navegador interpretará de una u otra forma para mostrar distintos contenidos por pantalla. Por ejemplo, si creamos un documento de texto que se llame ejemplo.html y que contenga el siguiente texto:

Una estructura HTML se empieza con la etiqueta `<html>` y acaba con `</html>`. Todo lo que esté en medio será la página web. Dentro de `<html></html>` se encuentran 2 partes diferenciadas. La primera `<head></head>` es la cabecera de la página. Aquí irán cierta información que no es directamente el contenido de la página. Aquí se pone el título de la página, los metadatos, estilos, código javascript (todo esto se estudiará en capítulos venideros). La primera que se suele estudiar es `<title></title>`, que indica el título de la página (lo que el navegador pone en la parte superior izquierda).

La segunda parte es `<body></body>`. Aquí va propiamente el contenido de la página: fotos, párrafos, formularios, etc. Por ejemplo, siguiendo con el ejemplo de la página anterior, el siguiente código, podemos cambiar el título de la página.

```
<html>
<head>
<title>Esto es el título de la página.</title>
</head>
<body>
Hola mundo!<br>
<b>Esto es negrita.</b><br>
<i>Y esto itálica.</i><br>
</body>
</html>
```

HTML5, lo básico

HTML5 es la versión 5 del lenguaje de marcado HTML. Un documento escrito en HTML tiene una estructura básica como la siguiente:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="utf-8" />
    <title>Hola Mundo!</title>
  </head>
  <body>
    <h1>Hola Mundo!</h1>
  </body>
</html>
```

<!DOCTYPE html> Lo primero que nos encontramos es la declaración del tipo de documento que se está mostrando. El DOCTYPE variará según el tipo de documento realizado, en nuestro ejemplo hemos utilizado el doctype del HTML5 que es maravillosamente sencillo.

html Tras declarar el tipo de documento, indicamos que iniciamos nuestro documento HTML. Esta etiqueta se cierra cuando finalizamos el documento. Vemos que lleva un atributo lang, esto sirve para indicar el idioma del documento (en nuestro caso español)

head En un documento HTML tenemos una cabecera dónde colocaremos los metadatos de la página, el código JavaScript y el CSS que utilizará el navegador para renderizar la página.

meta charset Obligatorio en HTML5, informa el juego de caracteres del documento, debería ser siempre utf-8. Como todo metadato debe ir dentro del head

title Es un tipo de metadato especial que nos proporciona el título de la página. Por motivos de posicionamiento (SEO) se recomienda que el meta title sea parecido al H1 del documento y a la URL de la página.

body En su interior tendremos el contenido de la página.

HTML5 ha venido a solucionar, entre otras cosas, la falta de un criterio para definir el contenido semántico de una página web agregando una serie de etiquetas destinadas a facilitar la estructura del documento desde el punto de vista de su significado.

section: Esta etiqueta sirve para agrupar elementos relacionados entre sí de forma temática. Los section creados a nivel del body serán aquellos cuyo contenido de significado a la página, o sea, formen el contenido principal de la misma.

article: Esta etiqueta es “la última etiqueta con significado semántico”. Habitualmente se utiliza dentro de un section para separar las unidades de contenido con significado semántico.

header: Creada para incluir información destinada a ayudar en la navegación. Suele incluir un H1 y, de declararse a nivel de body, la etiqueta nav.

nav: Esta etiqueta la utilizaremos para incluir el menú de navegación.

footer: Destinada a incluir la información sobre el elemento que lo contiene (autoría, propiedad, enlaces...)

aside: Su uso indicado es para agrupar el contenido a visualizar en la página, pero que no forma parte del contenido principal de la página.

Estas nuevas etiquetas permiten varios niveles de anidamiento entre sí. Por ejemplo, un section (por definición del estándar) debe tener un header y un footer, además de los article necesarios, pudiendo incluso tener otros section en su interior.

A nivel de body también podemos tener un header y footer, estando dentro del header la etiqueta nav con el menú del sitio web.

Los aside también pueden estar a nivel de body, section e incluso article.

Nombre de la sección que se creará en el OVI:
4.1 Objetivo de la sección: (Registre a continuación el objetivo que tiene esta sección)
<ul style="list-style-type: none"> CSS Y CSS3
4.2 Recursos de consulta que usará en la sección: (coloque el nombre del material que usará para crear los contenidos de la sección y el enlace de descarga de los mismos sean estos Texto, Imágenes, Audios o Vídeos)
<p>INTRODUCCIÓN WEB Y MULTIMEDIA.</p> <p>http://santadecadencia.blogspot.es/1461847625/diferencias-y-caracteristicas-de-css-css3/</p> <p>Definición de CSS - ¿Qué son las hojas de estilo o cascading style sheets?</p> <p>http://www.masadelante.com/faqs/css</p> <p>CSS3</p> <p>https://www.ecured.cu/Crisis_de_Octubre</p>
4.3 Redacte un borrador del contenido de lectura en formato de texto que tendrá la sección: (Sea este la presentación de la sección, el contenido o ambos; redacte un borrador del texto que publicará como contenido en la sección coloque un subtítulo para identificar si corresponde a la presentación de la sección o el contenido de lectura de la sección)

CSS

CSS son las siglas de Cascading Style Sheets - Hojas de Estilo en Cascada - que es un lenguaje que describe la presentación de los documentos estructurados en hojas de estilo para diferentes métodos de interpretación, es decir, describe cómo se va a mostrar un documento en pantalla, por impresora, por voz (cuando la información es pronunciada a través de un dispositivo de lectura) o en dispositivos táctiles basados en Braille.

¿Para qué sirve?

CSS es una especificación desarrollada por el W3C (World Wide Web Consortium) para permitir la separación de los contenidos de los documentos escritos en HTML, XML, XHTML, SVG, o XUL de la presentación del documento con las hojas de estilo, incluyendo elementos tales como los colores, fondos, márgenes, bordes, tipos de letra..., modificando así la apariencia de una página web de una forma más sencilla, permitiendo a los desarrolladores controlar el estilo y formato de sus documentos.

¿Cómo funciona?

El lenguaje CSS se basa en una serie de reglas que rigen el estilo de los elementos en los documentos estructurados, y que forman la sintaxis de las hojas de estilo. Cada regla consiste en un selector y una declaración, esta última va entre corchetes y consiste en una propiedad o atributo, y un valor separados por dos puntos.

Selector

Ejemplo:

```
h2 {color: green;}
```

h2 ---> es el selector

{color: green;} ---> es la declaración

color ---> es la propiedad o atributo

green ---> es el valor

Selector

El *Selector* especifica que elementos HTML van a estar afectados por esa declaración, de manera que hace de enlace entre la estructura del documento y la regla estilística en la hoja de estilo.

Declaración

La *Declaración* que va entre corchetes es la información de estilo que indica cómo se va a ver el selector. En caso de que haya más de una declaración se usa punto y coma para separarlas.

Propiedad o Atributo y Valor

Dentro de la declaración, la *Propiedad* o *Atributo* define la interpretación del elemento asignándole un cierto *Valor*, que puede ser color, alineación, tipo de fuente, tamaño..., es decir, especifican qué aspecto del selector se va a cambiar.

Tres tipos de estilos

La información CSS se puede proporcionar por varias fuentes, ya sea adjunto como un documento por separado o incorporado en el documento HTML, y dentro de estas posibilidades destacan tres formas de dar estilo a un documento web:

Hoja de Estilo Externa

La *Hoja de Estilo Externa* se almacena en un archivo diferente al del archivo con el código HTML al cual está vinculado a través del elemento *link*, que debe ir situado en la sección *head*. Es la manera de programar más eficiente, ya que separa completamente las reglas de formato para la página HTML de la estructura básica de la página.

Hoja de Estilo Interna

La *Hoja de Estilo Interna* es incorporada a un documento HTML, a través del elemento *style* dentro de la sección *head*, consiguiendo de esta manera separar la información del estilo del código HTML.

Estilo en Línea

El *Estilo en Línea* sirve para insertar el lenguaje de estilo directamente dentro de la sección *body* con el elemento *style*. Sin embargo, este tipo de estilo no se recomienda pues se debe intentar siempre separar el contenido de la presentación.

Versiones CSS

Existen varias versiones: *CSS1* y *CSS2*, la *CSS3* está todavía en desarrollo por el CSS WG (Cascading Style Sheets Working Group).

Los navegadores actuales implementan bastante bien *CSS1* desde 1999 (tres años después de su lanzamiento) aunque dependiendo de la marca y versión del navegador hay algunas pequeñas diferencias de implementación. El primer navegador en dar soporte completo al *CSS1* ha sido Internet Explorer 5.0 for the Macintosh en 2000, anteriormente el que mejor soportaba *CSS1* había sido Opera, después otros navegadores también lo han ido implementando.

Sin embargo, *CSS2* (lanzado en 1998) sólo está parcialmente implementado en los navegadores más recientes, variando en estos los niveles de implementación.

Ventajas de CSS

La principal ventaja de CSS sobre el lenguaje HTML o similar, es que el estilo se puede guardar completamente por separado del contenido siendo posible, por ejemplo, almacenar todos los estilos de presentación para una web de 10.000 páginas en un sólo archivo de CSS.

CSS permite un mejor control en la presentación de un sitio web que los elementos de HTML, agilizando su actualización.

Aumento de la accesibilidad de los usuarios gracias a que pueden especificar su propia hoja de estilo, permitiéndoles modificar el formato de un sitio web según sus necesidades, de manera que por ejemplo, personas con deficiencias visuales puedan configurar su propia hoja de estilo para aumentar el tamaño del texto.

El ahorro global en el ancho de banda es notable, ya que la hoja de estilo se almacena en cache después de la primera solicitud y se puede volver a usar para cada página del sitio, no se tiene que descargar con cada página web. Por otro lado, quitando todo lenguaje de marcado en la presentación en favor del uso de CSS reduce su tamaño y ancho de banda hasta más del 50%, esto beneficia al dueño del sitio web con menos ancho de banda y costes de almacenamiento, así como a los visitantes para los cuales las páginas se van a cargar más rápido.

Una página puede tener diferentes hojas de estilo para mostrarse en diferentes dispositivos, como pueden ser impresoras, lectores de voz, o móviles.

¿Para qué sirve el css 3?

El CSS sirve para definir la estética de un sitio web en un documento externo y eso mismo permite que modificando ese documento (la hoja CSS) podamos cambiar la estética entera de un sitio web, el mismo sitio web puede variar totalmente de estética cambiando solo la CSS, sin tocar para nada los documentos HTML o jsp o asp que lo componen.

CSS es un lenguaje utilizado para dar estética a un documento HTML (colores, tamaños de las fuentes, tamaños de elemento, con css podemos establecer diferentes reglas que indicarán como debe presentarse un documento. Podemos indicar propiedades como el color, el tamaño de la letra, el tipo de letra, si es negrita, si es itálica, también se puede dar forma a otras cosas que no sean letras, como colores de fondo de una página, tamaños de un elemento (por ejemplo el alto y el ancho de una tabla).

características principales de css3

- 1. Atributo gradiente de colores en borde con CSS y Firefox:** Posibilidad de definir el un gradiente de color en el borde de los elementos con CSS, en un atributo no estándar de Firefox.
- 2. Bordes redondeados en CSS 3:** Las características de CSS 3 incluyen bordes redondeados, a través del atributo border-radius, que define la curvatura que debe tener el borde del elemento.
- 3. Múltiples imágenes de fondo con CSS:** Cómo conseguir que un elemento de la página tenga varias imágenes de fondo a la vez, con CSS básico y con características de CSS 3.
- 4. Colores RGBA en CSS 3:** Veremos qué son los colores RGBA y su notación, que se incluyen en la especificación de Hojas de Estilo en Cascada CSS 3.
- 5. Word-wrap en CSS 3:** Una propiedad de CSS 3 que sirve para romper las palabras que son demasiado largas y no caben enteras por la anchura de una caja.
- 6. Textos multi-columna con CSS 3:** CSS 3 incorpora nuevos atributos para que el navegador se encargue de producir texto multicolumna, es decir, que maquete directamente el texto en varias columnas sin tener que hacer nosotros nada.
- 7. Bordes con imágenes en CSS 3:** El atributo border-image y varios otros de CSS 3 harán posible la utilización de imágenes como bordes de los elementos de la página, sin código HTML especial, simplemente con hojas de estilo.
- 8. Sombras en CSS 3 con box-shadow:** Crear sombras en CSS3 con el atributo box-shadow. Por fin podremos aplicar sombras a los elementos de la página, sin usar imágenes, Javascript ni nada extra, simplemente con un atributo de CSS 3.
- 9. Resplandor exterior con CSS3:** Cómo realizar un elemento que tenga un resplandor exterior con CSS3 y la propiedad box-shadow.

10. Propiedad background-origin de CSS 3: La propiedad de CSS 3 background-origin permite decidir la posición de la imagen de fondo con respecto al borde, padding o el contenido del elemento.

11. Atributos CSS3 overflow-x y overflow-y: Descripción de los atributos de CSS3 overflow-x y overflow-y, que sirven para definir cómo renderizar un contenido cuando sobrepasa los límites de un contenedor en la horizontal o vertical.

12. Introducción a @font-face de CSS: Fuentes en CSS 3. Sintaxis y principales características de la regla CSS @font-face, que nos permite utilizar cualquier tipografía en una página web.

13. Sombras en el texto con text-shadow de CSS: Cómo aplicar sombras y otros efectos en los textos con CSS y el atributo text-shadow.

Ventajas y desventajas de css3

Se obtiene un mayor control de la presentación del sitio al poder tener todo el código CSS reunido en uno, lo que facilita su modificación.

Al poder elegir el archivo CSS que deseamos mostrar, puede aumentar la accesibilidad ya que podemos asignarle un código CSS concreto a personas con deficiencias visuales, por ejemplo. Esto lo detecta el navegador web.

Conseguimos hacer mucho más legible el código HTML al tener el código CSS aparte (Siempre que no usemos estilos en línea, claro está).

Pueden mostrarse distintas hojas de estilo según el dispositivo que estemos utilizando (versión impresa, versión móvil, leída por un sintetizador de voz...) o dejar que el usuario elija.

Gracias a la técnica CSS Sprites podemos aligerar la carga de nuestro sitio al juntar todas las imágenes en una.

Las novedades de CSS3 nos permiten ahorrarnos tiempo y trabajo al poder seguir varias técnicas (bordes redondeados, sombra en el texto, sombra en las cajas, etc.) sin necesidad de usar un editor gráfico.

Desventajas

Existen limitaciones que CSS 2.x todavía no permite, por ejemplo, la alineación vertical de capas, las sombras, los bordes redondeados...

El uso de las tablas nos permitía crear diseños complejos de forma mucho más sencilla que utilizando CSS, aunque CSS3 está intentando facilitar dicho trabajo.

A veces, dependiendo del navegador (Acid tests), la página que ha sido maquetada con CSS puede verse distinta (Aunque, si hemos seguido los estándares web de forma correcta, el problema es del navegador).

Aplicaciones o usos de css3

- Botones y HTML: Mostramos algunas insuficiencias del HTML en el diseño de formularios y en concreto en el retoque de botones de envío de formularios.
- Botones y CSS: Explicamos las mejoras que pueden obtenerse por el uso de hojas de estilo en la creación de formularios y, más concretamente, los botones de envío.

- Botones e imágenes: Como podemos mejorar los botones con el uso de imágenes en lugar de botones de submit corrientes.
- Botones, CSS y tablas: Vamos a mejorar los botones normales, que al hacer click hacen efecto de pulsación, con imágenes a los lados para mejorar su aspecto.
- Botones, imágenes y eventos: Para que los botones de imagen no ignoren los eventos de teclado, vamos a utilizar una función Javascript.
- Botones, tablas y eventos: Ampliamos el ejemplo anterior para que el texto de los botones se pueda editar fácilmente, sin perder en presentación.

Importancia de su uso en determinados problemas o en ciertos sistemas

El CSS nos permite controlar los estilos de nuestros elementos HTML, en realidad con el HTML definimos la estructura de la web & con el CSS todo lo demás - o sea colores, fuentes, ubicación & todo lo que se va a mostrar de nuestra web.

Formas de aplicar CSS a una página HTML

1. Dentro de la etiqueta HTML – controlando solo la etiqueta – por ejemplo - `p {color:red;text-align:center;}`.
2. Dentro del archivo HTML (dentro de la zona HEAD) – controlando solamente este archivo -
`<style>p {color:red;text-align:center;}</style>`.
3. Creando un archivo exterior .css, controlando todos los archivos web del sitio usando la etiqueta para aplicar los estilos - `<link rel="stylesheet" type="text/css" href="theme.css" />` - esto va dentro del HEAD de nuestra página `p {color:red;text-align:center;}` – los estilos los aplicamos en el archivo externo css.

CSS3 considerado tecnología de punta o moderna

Las nuevas características de CSS nos permiten añadir efectos de animación a la mayoría de elementos HTML, sin necesidad de Javascript o Flash. Por el momento, es compatible con los navegadores WebKit -incluyendo Safari, Safari para iOS y Chrome- y Firefox. Está previsto que Internet Explorer, a partir de su versión 10, también tenga soporte para este tipo de animaciones. Debido a que la tecnología es relativamente nueva, se debe añadir un prefijo con el motor del navegador. En nuestro caso, los ejemplos los realizaremos con los prefijos webkit y moz.

Diferencias entre css y css3

"La diferencia entre css y css3 es que css3 es una actualización que agrega más cantidades de atributos a un lenguaje como es el css."

Tal vez la mayor diferencia entre CSS2 y CSS3 es la separación de los módulos. Mientras que en la versión anterior todo fue una larga especificación definiendo diferentes características, CSS3 está dividido en varios documentos llamados módulos. Cada módulo cuenta con nuevas capacidades, sin afectar la compatibilidad de la versión estable anterior. Al hablar de módulos, podemos nombrar más de cincuenta, sin embargo, cuatro de ellos han sido publicados como recomendaciones formales, y se componen de lo siguiente:

Link Publico de la maquetación

<file:///C:/Users/usuarios/Desktop/Diseno%20Web/index.html>



INICIO NOSOTROS SERVICIOS PRODUCTOS NOTICIAS CONTACTO



DISEÑO WEB

ADAPTADO
PERSONALIZADO
ACCESIBLE
DIFERENTE
AUTOGESTIONABLE

ESPECIALISTAS EN DISEÑO WEB

*Contamos con los mejores servicios para su empresa,
que espera comuníquese con nosotros*



Diseño Web



Tiendas Virtuales



Marketing Digital

Realizado por Angel Andres Ortiz 301122A_761