Desarrollo de Aplicaciones Web

Instrucciones generales

1. Objetivos

- Establecer unas pautas comunes para el desarrollo de las prácticas.
- Indicar algunas herramientas que pueden ayudar al desarrollo de las prácticas.

2. Introducción a la asignatura

La asignatura "Desarrollo de Aplicaciones Web" está dividida en dos partes, la programación del lado del cliente y la programación del lado del servidor.

Desde el nacimiento de la Web en el año 1990, se han desarrollado diferentes tecnologías para la programación de las aplicaciones web, tanto del lado del cliente como del servidor. A la hora de elegir una tecnología, un requisito imprescindible es garantizar la compatibilidad con los diferentes navegadores web que existen. Por ello, la mejor opción es emplear siempre las tecnologías que tienen la consideración de "estándar de facto".

Las tecnologías que hoy en día son consideradas el estándar del cliente web son HTML, CSS, JavaScript y DOM. De hecho, estas son las mismas tecnologías que se emplean en el desarrollo web desde mediados de los años noventa. De estas cuatro tecnologías, en esta asignatura se estudiará lo mínimo para poder desarrollar una aplicación web con una funcionalidad básica. El estudio profundo de estas tecnologías está reservado para la asignatura "Programación de Clientes Web".

En la programación del lado del servidor existe más libertad, ya que lo único que se debe cumplir es que el resultado que genere el servidor web sea HTML, CSS, JavaScript y DOM.

En esta asignatura, en la programación del lado del servidor se utiliza XAMPP (pero se puede usar otra plataforma similar, no es problema) para poder trabajar con PHP y MySQL.

3. Sobre el proyecto de la asignatura

En esta asignatura tienes que desarrollar una aplicación web, con su parte cliente y su parte servidor. El proyecto se puede considerar que es un **producto mínimo viable**[1]: "un producto con suficientes características para satisfacer a los clientes iniciales, y proporcionar retroalimentación para el desarrollo futuro". Esto significa que hay muchas cosas que se deberían mejorar para ser considerado un producto real, pero es lo normal en cualquier desarrollo.

El proyecto se ha dividido en prácticas semanales, como si fueran hitos desde el punto de vista de la ingeniería del software. Cada práctica aporta al desarrollo del proyecto, lo que significa que se deben realizar todas las prácticas para considerar que el proyecto ha tenido éxito.

Cada semana, en el enunciado de la práctica se te facilitará un diagrama y una descripción de las páginas que debes implementar. Las páginas del diagrama pueden ser también estados de la página, es decir, cada página no tiene que ser un fichero independiente en la aplicación final, pero en las primeras prácticas en las que la aplicación es estática sí que tienen que ser ficheros independientes para poderse probar, para poder navegar de una página a otra.

Por ejemplo, en algunos casos se pueden implementar varias páginas del diagrama en una sola mediante pestañas. A veces también se puede implementar mediante ventanas emergentes (*pop-ups*), pero es mejor no abusar de las ventanas emergentes porque pueden causar problemas de usabilidad y accesibilidad.

[1] `https://es.wikipedia.org/wiki/Producto_viable_m%C3%ADnimo`

4. Cómo se corrigen las prácticas

Las prácticas de esta asignatura son acumulativas, por lo que no es posible "saltarse" una práctica. Si por cualquier razón, una práctica no es entregada en la fecha establecida, la práctica no será recuperada posteriormente, pero sí que será necesario realizarla porque las prácticas posteriores dependerán de ella.

Además, como las prácticas son acumulativas, tienes que cumplir exactamente lo que se te pida en cada práctica, si te desvías de lo que se te pida es posible que tengas graves problemas en prácticas posteriores.

Como las prácticas son acumulativas, **los errores que se detecten en una corrección se deben corregir para que no vuelvan a aparecer en las siguientes correcciones**, ya que se tendrán en cuenta en todo momento.

5. Documentación de las prácticas

Documentar una aplicación es aburrido, pero es algo necesario. En esta asignatura no se exige que las prácticas tengan una documentación independiente en un formato estándar de documentación, pero **sí que se exige que el código esté correctamente organizado y documentado y que exista un fichero con una documentación general que explique la estructura y contenido del código fuente**.

Documentar vuestro trabajo es fundamental, tanto en vuestra etapa de estudiante como luego cuando estéis en la etapa laboral. En los proyectos de software se le da mucha importancia a la documentación. Sin ella, es imposible el mantenimiento de aplicaciones ni el trabajo en equipo. Para que os hagáis una idea, un ingeniero programador emplea cerca del 30 % de su tiempo en documentar los programas y proyectos que realiza.

Si en un futuro cercano tuvieseis que realizar algún cambio en vuestros proyectos, y estos están bien documentados, no tendréis problemas en retomarlos y modificarlos según las nuevas necesidades. Pero si no los habéis documentado y tienen una mínima complejidad, realizar cualquier cambio puede convertirse en algo tan complicado y tedioso que quizá a veces sea más rentable hacerlo de nuevo.

Igualmente, cuando trabajas dentro de un equipo, la documentación permite a tus compañeros entender lo que haces y por qué lo haces, reutilizar código de otros, comprobar funcionamientos correctos y, en general, ahorrar tiempo y trabajo.

Desde el principio es importante que documentéis vuestro trabajo, pues si no, conforme se va incrementando la complejidad de la práctica, se hace muy complicado el seguimiento de esta. En caso de necesitar corrección o ayuda sobre el código entregado, será en algunos casos imposible encontrar los fallos si el código no está correctamente documentado.

No es necesario que realices una documentación exhaustiva de la práctica, pero sí es necesario cumplir como mínimo lo siguiente:

- Incluir un archivo `Leeme.txt` o `Leeme.doc` (`Readme.txt` o `Readme.doc`) en el cual se indiquen todos los archivos que componen el proyecto y una mínima explicación (una o dos líneas como máximo) de lo que hacen o para que sirven. Por ejemplo:

  ```
  index.php: Archivo principal de código desde el cual se genera la página inicial

  head.php: Cabecera genérica para todas las páginas

  compruebaclave.php: Archivo con las funciones para comprobar el usuario y la contraseña
  ```

- En cada archivo de código se debe incluir un comentario que incluya:

  - El nombre del archivo.
  - Una descripción general de lo que hace el archivo.
  - Los nombres de los autores.
  - La fecha incial de creación.
  - Un historial de revisiones o cambios.

  Por ejemplo, en el caso de HTML se podría indicar de la siguiente forma:

```
<!--
Archivo: index.html
En este archivo se define la página principal del sitio web

Creado por: Juan Nadie el 12/09/2020

Historial de cambios:
15/10/2020 Se añade el pie de página con los nombres de los autores
-->
```

En el caso de CSS podría ser:

```
/*
Archivo: estilos.css
En este archivo se definen los estilos básicos de la página web

Creado por: Juan Nadie el 12/09/2020

Historial de cambios:
15/10/2020 Se añade el estilo para el pie de página con los nombres de los autores
*/
```

Y en el caso de PHP podría ser:

```
/**************************
Archivo: compruebaclave.php
En este archivo se define la función compruebaClave(usuario, contraseña),
a la cual se le pasan los parámetros de usuario y contraseña recogidos
desde un formulario, y devuelve si es valido o no

Creado por: Pepito Martinez el 20/10/2020

Historial de cambios:
21/10/2020: Se añade arrays para almacenar usuarios

10/11/2020: Se sustituyen arrays por acceso a la base de datos para comprobar usuarios
*****************************/
```

- Dentro de cada archivo de código, un comentario al principio de cada función, bucle, o cualquier otro punto singular que necesite una explicación. Por ejemplo, en el caso de PHP podría ser:

```
/***********************
Función que comprueba que un usuario y clave sean correctos

Como entrada toma dos cadenas de texto con el usuario y la clave a comprobar

Devuelve 1 si es correcto, o 0 si no lo es

Para la comprobación, guarda en el array $usuarios todos los usuarios permitidos con sus
    correspondientes claves
***********************/

function compruebaClave(usuario, clave){

       // El código de la función

}
```

- Dentro del código, un comentario en cualquier otro punto singular que necesite una explicación. Por ejemplo, en un bucle, la definición de una variable o constante. Por ejemplo, en el caso de PHP podría ser:

```
$a = array("pepe" => "clave1", "jose" => "clave2"); //define un array donde se guardan los
    usuarios y claves
```

```
while($c < sizeof($a)) {
       // Recorre el array y muestra resultados que coinciden

       // El código de la función

}
```

Todo lo anterior es suficiente para documentar la práctica de esta asignatura. No obstante, si queréis ampliar información sobre la forma de documentar un proyecto informático, y aunque no existe una norma ISO o UNE que lo defina, hace ya varios años que el Consejo General de Colegios Profesionales de Ingeniería Informática publicó "CCII-N2016-02 Norma Tecnica para la realizacion de la Documentacion de Proyectos en Ingenieria Informatica - V1.0"[2] que puede ayudar en gran medida a estandarizar la documentación de proyectos informáticos. En la noticia[3] sobre su publicación se explica:

> Norma Técnica sobre la documentación de los proyectos de ingeniería informática, y por lo tanto un elemento esencial para la concepción, dirección y gestión de proyectos de ingeniería informática. Se trata de facilitar el desempeño de los profesionales y la labor de las empresas, y organizaciones en general, en el ámbito de la ingeniería informática, aportando estándares de trabajo y normalización de procedimientos en la confección de sus proyectos, ya sea para su licitación, implementación, subcontratación, etc.
>
> Esta norma subsana una importante carencia en el ámbito de la dirección y gestión de proyectos de ingeniería informática, facilitando y promoviendo así la cultura de la normalización en el sector informático español al proporcionar un mecanismo sencillo y eficiente para la especificación documental de los proyectos de ingeniería informática. Adicionalmente su utilización facilita actuaciones y servicios de auditoría de proyectos y es fundamental para implantar la normalización en los proyectos de Ingeniería en Informática y permitir los servicios de "Revisión de integridad documental" y de "Visado" de proyectos, previstas en el Real Decreto 1000/2010, de 5 de agosto, sobre visado colegial obligatorio.

De todas formas, deciros que en el mundo de la empresa, hoy por hoy, cada cual tiene su metodología de documentación, no hay nada estandarizado. Y aunque la mayoría de las metodologías son muy parecidas en general, siempre tendréis que adaptaros a la específica que os pidan en cada empresa. Pero también es cierto que, en todas, la base, a grandes rasgos, suele incluir los puntos que se han indicado en este documento.

6. Usabilidad y accesibilidad

La usabilidad y la accesibilidad del software es un tema muy importante que normalmente los desarrolladores no tienen en cuenta. Un desarrollador de software normalmente no tiene problemas para usar sus propios desarrollos porque, al fin y al cabo, son sus desarrollos y los conoce a la perfección. Pero ¿otras personas los podrán usar?

En el plan de estudio existe otra asignatura centrada en la usabilidad y la accesibilidad, pero ello no es excusa para que cualquier desarrollo deba ser usable y accesible. Por ejemplo, cualquier aplicación debe:

- Mostrar la información de forma ordenada para que el usuario la pueda localizar y usar fácilmente.
- No cambiar el comportamiento normal de los elementos de la interfaz de usuario.
- Proporcionar una respuesta (*feedback*) a cualquier acción que realice el usuario.
- Mostrar mensajes de error claros y significativos, que ayuden al usuario a entender lo que está pasando en todo momento.

En el artículo "Basic usability, basically missed"[4] se proporcionan algunos consejos básicos de usabilidad:

[2] https://ccii.es/servicios/area-de-descargas/download/6-documentacion-de-proyectos/4-ccii-n2015-02-norma-tecnica-para-la-realizacion-de-la-documentacion-de-proyectos-en-ingenieria-informatica-v1-0

[3] https://www.ccii.es/noticias/313-ccii-n2016-02-norma-tecnica-para-la-realizacion-de-la-documentacion-de-proyectos-en-ir

[4] https://uxdesign.cc/basic-usability-basically-missed-f05fd69d2c13

- Navigation Inconsistency.
- Structure Inconsistency.
- Returning Home.
- Signing in/Retrieving quotes.
- Letting the User Know Where They Are.
- Removal of the Hamburger Menu.
- Using Blue for Non-link Text.
- Repetitive Links.

Conviene que consultes estos consejos para no caer en los errores típicos que tienen los sitios y aplicaciones web.

7. Sobre las aplicaciones de una sola página

Según Google, una aplicación de una sola página[5] (*single-page application*, SPA) es:

> Una aplicación de una sola página es una aplicación web o un sitio web que carga todos los recursos necesarios para navegar por el sitio en la primera carga de la página. A medida que el usuario hace clic en los enlaces e interactúa con la página, el contenido posterior se carga de forma dinámica. Con frecuencia, la aplicación actualiza la URL en la barra de dirección para emular la navegación de página tradicional, pero no se vuelve a hacer otra solicitud de página completa.

En ese pequeño párrafo de Google ya aparece uno de los problemas de las SPA, se pierden los URL y para tener ciertas funcionalidades como los marcadores (*bookmarks*), el volver atrás o el posicionamiento en los buscadores es necesario que "la aplicación actualiza la URL en la barra de dirección para emular la navegación de página tradicional".

Antes de usar SPA te tendrías que hacer la siguiente pregunta: ¿realmente necesito SPA? En el artículo "You probably don't need a single-page application"[6] dan varios consejos para responder a esa pregunta. También es muy interesante el artículo "The disadvantages of single page applications"[7]. Y otro problema muy importante son los problemas de accesibilidad que presentan las SPA, tal como se explica en "One-page-applications are not accessible"[8].

Por todo ello, **en esta asignatura no se requiere que la práctica se desarrolle como una SPA, es más, se desaconseja totalmente**.

8. Sobre el scroll infinito

El scroll infinito es otra técnica de desarrollo web moderna que "mola mucho", pero que en la mayoría de las situaciones causa problemas graves. Los artículos "Infinite Scroll: Let?s Get To The Bottom Of This"[9] y "Infinite Scrolling Is Not for Every Website"[10] explican los problemas que presenta el scroll infinito.

Es más, se cree que el scroll infinito causa adicción entre los usuarios y hay incluso un movimiento para pedir su prohibición: "Social media apps are 'deliberately' addictive to users"[11] y "US could ban 'addictive' autoplay videos and infinite scrolling online"[12].

Por ello, **en esta asignatura tampoco se debe hacer uso del scroll infinito**. En su lugar debes emplear paginación cuando se tenga que mostrar mucha información.

[5] https://developers.google.com/analytics/devguides/collection/analyticsjs/single-page-applications?hl=es-419

[6] https://journal.plausible.io/you-probably-dont-need-a-single-page-app

[7] https://adamsilver.io/articles/the-disadvantages-of-single-page-applications/

[8] http://www.craigabbott.co.uk/one-page-applications-are-not-accessible

[9] https://www.smashingmagazine.com/2013/05/infinite-scrolling-lets-get-to-the-bottom-of-this/

[10] https://www.nngroup.com/articles/infinite-scrolling/

[11] https://www.bbc.com/news/technology-44640959

[12] https://www.theguardian.com/media/2019/jul/31/us-could-ban-addictive-autoplay-videos-and-infinite-scrolling-online

9. Sobre los frameworks

Según la Wikipedia[13], un framework es "un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar".

En el desarrollo web existen frameworks para casi todas las tecnologías que se emplean. Hasta HTML tiene sus propios frameworks, o más bien, plantillas para comenzar a desarrollar una página web. Dos de los más famosos son HTML5 Boilerplate[14] e Initializr[15]. Para las otras tecnologías que se emplean en esta asignatura (CSS, JS y PHP) existen cientos de frameworks.

Un framework es una gran ayuda para el desarrollo de grandes aplicaciones. Cuando un framework se usa correctamente, guía el desarrollo, reduce las posibilidades de error y mejora la productividad. Sin embargo, cuando se está aprendiendo una nueva tecnología o paradigma de desarrollo, un framework dificulta el aprendizaje porque oculta muchos detalles que es importante conocer. Por eso, **en esta asignatura no se puede usar un framework**.

10. Consejos generales

- Toda la aplicación debe estar en un mismo idioma. Puedes hacerla en español, valenciano o inglés, pero no mezcles diferentes idiomas.
- Todas las páginas deben tener una presentación consistente. Para ello, deben tener unos elementos comunes que se repitan en todas las páginas (cabecera y pie de página, barra de navegación). En el pie de página incluye los nombres de los autores de la práctica, un aviso de copyright con el año y alguna información más.
- Almacena cada práctica en un directorio independiente; al final de la asignatura debes tener un directorio por cada práctica semanal. Cuando comiences una nueva práctica, copia y renombra el directorio de la práctica anterior.
- El fichero de la página principal de un sitio web suele tener el nombre `index.html`. Sólo como curiosidad, los sevidores web suelen tener configurada la siguiente lista de páginas por defecto para la página principal[16]: `index.htm`, `index.html`, `index.php`, `index.php3`, `index.php5`, `index.php4`, `index.shtml`, `default.htm`, `default.html`, `index.py`, `default.shtml`, `index.pl`, `index.cgi` y `home.html`.
- Se recomienda usar el juego de caracteres UTF-8.
- Ten cuidado con los nombres de los ficheros, utiliza únicamente letras del alfabeto inglés y números, no uses espacios en blanco y emplea únicamente minúsculas.
- Ten cuidado con la caché del navegador, consulta una explicación sobre los problemas y soluciones en el artículo "Ayuda:Cómo limpiar la caché"[17]. En Google Chrome, cuando se muestran las herramientas para inspeccionar el código de una página, se puede pulsar con el botón derecho sobre el icono de recargar y aparece un menú para forzar la recarga y el vaciado de la caché, tal como se puede ver en la Figura 1. Recuerda esto durante toda la asignatura.

[13] `https://es.wikipedia.org/wiki/Framework`
[14] `https://html5boilerplate.com/`
[15] `http://www.initializr.com/`
[16] El orden de configuración es importante porque el servidor web va a intentar encontrar los archivos en el orden indicado.
[17] `https://es.wikipedia.org/wiki/Ayuda:C%C3%B3mo_limpiar_la_cach%C3%A9`

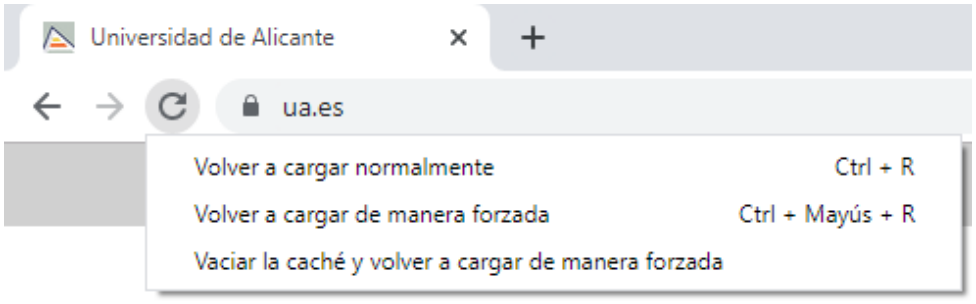

Figura 1: Opciones para volver a cargar una página web en Google Chrome