UD1

Tarea: Selección de arquitectura y herramientas de programación

Tarea 1: Lenguajes Interpretados Vs Lenguajes Compilados

Realice un **resumen** explicando cada una de ellas y mencione las **ventajas y desventajas** de estos 2 tipos/clasificación de los lenguajes de programación. ¿En qué categoría se sitúa JavaScript?.

Muestre algún ejemplo y trate de que se le queden bien claros y asimilados estos 2 conceptos.

NOTA: Debe de encontrar la respuesta en Internet.

Un **lenguaje interpretado** es un tipo de lenguaje de programación para el cual la mayoría de sus implementaciones ejecutan instrucciones directa y libremente, sin compilar previamente un programa en instrucciones de lenguaje de máquina. El intérprete ejecuta el programa directamente, traduciendo cada instrucción en una secuencia de una o más subrutinas, y luego en otro lenguaje (a menudo código de máquina).
La principal ventaja de un lenguaje interpretado es que es independiente de la máquina y del sistema operativo ya que no contiene instrucciones propias de un procesador sino que contiene llamadas a funciones que el intérprete deberá reconocer. Esto hace la vida más fácil al programador.

Como desventaja hay que destacar la velocidad. Es el aspecto más notable y el cual se debe evaluar a fondo al crear software con este tipo de lenguajes. Pues se debe equilibrar la portabilidad con la velocidad que se está sacrificando a no ser que las prestaciones de los equipos informáticos sean bastante altas, en cuyo caso, se podría despreciar este aspecto.

Por otro lado existe el problema de la portabilidad. En la actualidad, casi todos los lenguajes compilados existen para todas las plataformas, no así las máquinas virtuales o frameworks, aunque en el caso de Java se ha hecho un excelente trabajo.

Un **lenguaje compilado** es un lenguaje de programación cuyas implementaciones son normalmente compiladores (traductores que generan código de máquina a partir del código fuente) y no intérpretes.
Entre las ventajas que ofrece el lenguaje compilado está la velocidad. Esto se debe a que cuando es ejecutado ya se encuentra en código de máquina y eso también le permite hacer algunas optimizaciones que no son posibles con un lenguaje interpretado. No se necesitan intérpretes en el lenguaje compilado al ser convertido a lenguaje máquina.
Como desventajas, el ciclo de desarrollo (el tiempo entre el momento en que escribes el código y lo pruebas) es más rápido en un lenguaje interpretado. Eso se debe a que en lenguajes compilados es necesario realizar el proceso de compilación cada vez que cambias el código fuente, aunque con herramientas adicionales se puede automatizar.

Cada poco tiempo van saliendo nuevos lenguajes y nuevos paradigmas.

Entre los lenguajes interpretados podemos mencionar algunos famosos como Ruby, Python, PHP (se interpreta del lado del servidor), JavaScript y otros como Perl, Smalltalk, MATLAB, Mathematica (el que usan en Wolfram Alpha).

Entre los lenguajes compilados encontramos C, C++, C#, Visual Basic, COBOL, Java (bytecode) y muchos más.

Tarea 2: Lenguajes Imperativos Vs Lenguajes Declarativos

Realice un **resumen** explicando cada una de ellas y mencione las **ventajas y desventajas** de estos 2 tipos/clasificación de los lenguajes de programación. ¿En qué categoría se sitúa JavaScript?.

Muestre algún ejemplo y trate de que se le queden bien claros y asimilados estos 2 conceptos.

Nota: Los declarativos no son tan obvios como los imperativos!!!

En la **programación imperativa (en el cual podemos situar a Javascript),** un programa se describe en términos de instrucciones, condiciones y pasos que modifican el estado de un programa al permitir la mutación de variables con el objetivo de llegar a un resultado. Ejemplo:

```
$listaparticipantes = [1 => 'Peter', 2 => 'Hans', 3 => 'Sarah'];
$nombres = [];
foreach ($listaparticipantes as $id => $apellido) {
```

```
        $nombres[] = $apellido;
    }
```

Entre las ventajas podemos destacar su relativa simplicidad y facilidad de implementación de los compiladores e intérpretes, capacidad de reutilizar el mismo código en diferentes lugares en el programa sin copiarlo, pista de flujo del programa fácil de seguir, capacidad modular y requiere menos memoria.

Como desventajas los datos son expuestos a la totalidad del programa, así que no hay seguridad para los datos. Dificultad para relacionarse con los objetos del mundo real. Se da importancia a la operación de datos en lugar de los datos mismos.

En la **programación declarativa,** un programa se describe en términos de proposiciones y afirmaciones que son declaradas para describir el problema sin especificar los pasos para resolverlo. En este tipo de programas, el estado no puede ser modificado ya que todos los tipos de datos son inmutables. Ejemplo:

```
$nombres = array_values($listaparticipantes);
```

Destaca como ventajas la solución de un problema que se puede realizar con un nivel de abstracción considerablemente alto, sin entrar en detalles de implementación irrelevantes, lo que hace a las soluciones más fácil de entender por las personas. La resolución de problemas complejos es resuelta por el intérprete a partir de la declaración de las condiciones dadas.

La principal desventaja de la programación declarativa es que no puede resolver cualquier problema dado, sino que está restringida al subconjunto de problemas para los que el intérprete o compilador fue diseñado. Otra desventaja es la eficiencia: dado que es necesaria una fase de interpretación extra en la cual se deben evaluar todas las consecuencias de todas las declaraciones realizadas, el proceso es relativamente más lento que en la programación imperativa en que los cambios de estado del sistema están dados por instrucciones particulares y no por un conjunto de condiciones arbitrariamente grande.

Entre los ejemplos de programación imperativa además del ya mencionado Javascript tenemos Python, Java, Ruby, C, C++, C#... Por el lado declarativo están Haskell, Lisp, Prolog, SQL, QML...

Tarea 3: Varios códigos fuentes .js !!!

Realice una prueba de cómo cargar código javascript desde más de un archivo .js en una misma página HTML. Busque por internet códigos javascript “sencillos” que hagan alguna operación sencilla. Al menos 2 códigos, y construya una página web simple que

"importe" esos códigos que estarán guardados en archivos diferentes. Lo que se trata es que sepa construir una web simple, que haga uso de código javascript, pero que ese código se importe de diferentes archivos.

Fichero fuente HTML:

```
<!DOCTYPE html>
<html lang="en-US">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>Ejemplo</title>
</head>
<body>
<script src="hora.js">
</script>
<script src="mensaje.js">
</script>
</body>
</html>
```

Fichero fuente "hora.js":

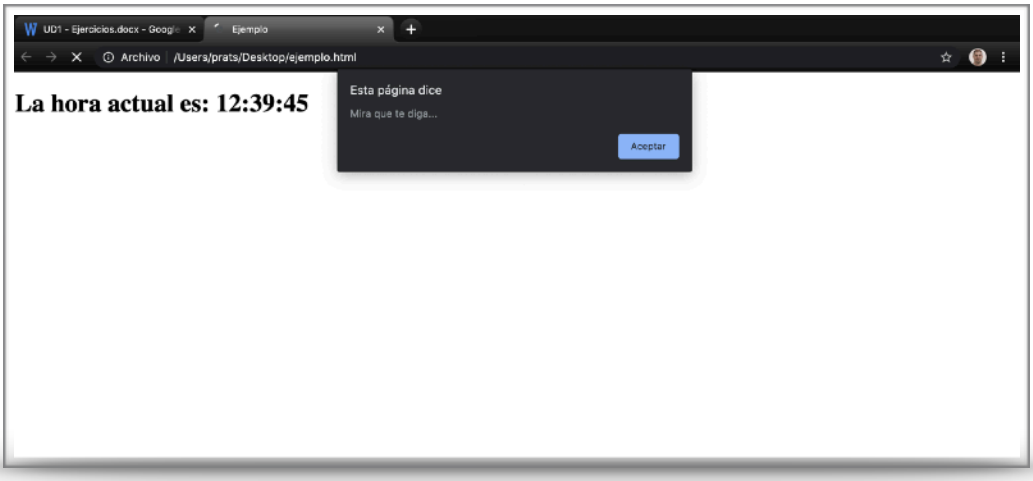
```
let d = new Date();

document.body.innerHTML = "<h1>La hora actual es:  " + d.getHours() + ":" +
d.getMinutes() + ":" + d.getSeconds()
```

Fichero fuente "mensaje.js":

```
alert("Mira que te diga...");
```

Resultado final en navegador:



Tarea 4: ¿Dónde es más recomendable incluir código javascript y por qué?

Trate de buscar en foros, discusiones, artículos información que se discuta las ventajas y desventajas de incluir código javascript en el “head” o en el “body”. Trate de razonar y ver los pros y contras, y/o los diversos casos en los que un programador se puede ver en esta disyuntiva.

Tras mucho analizar, la conclusión es: **no se puede adjudicar la mejor práctica como única vía para todos los casos**. Habrá casos en los que no sea necesario realizar uno o varios scripts y en consecuencia habrá otros casos en los que bastará con incluírlo como código embebido dentro del propio html, ya sea en la cabecera o en el cuerpo. En cualquier caso, como norma general sí es cierto es que muchas opiniones coinciden en que lo mejor es incluirlo justo antes de la etiqueta final </body>, ya que de esa forma se evitan problemas de rendimiento y rendering.

¡¡¡ No realizar Tarea 5 porque vamos a realizar una práctica exclusiva sobre depurar código !!!

Tarea 5: ¿Cómo podemos depurar / debugear un código JavaScript haciendo uso de Mozilla Firefox? “Ejercicio Estrella”

Encuentre información y realice un ejercicio práctico para demostrar que haya adquirido esta destreza. Añada alguna captura. Le recomiendo que para hacer las capturas busque un código Javascript que contenga alguna variable, y cambios en ella, de forma que pueda ver y mostrar por capturas como en la pestaña de depuración del Firefox se va modificando el valor. ¿Podemos cambiar el valor de una variable en tiempo de ejecución haciendo uso del depurador?

Sin duda, todo programador debe de aprender a depurar bien su técnica de depuración, válgase de la redundancia.

Tarea 6: Busque información sobre los distintos y más usados frameworks/librerías utilizados hoy día para desarrollar funcionalidad propia de JavaScript.

Hay mucha información sobre estos frameworks en la web. De lo que trata la tarea es de alcanzar una visión global y observar cuales son tendencia en el mundo actual. Lo discutiremos en clase.

Los frameworks Javascript que se encuentran entre el top 3 para desarrollo **frontend**:

· Angular: lanzado en 2010, se ha convertido en un standard gracias a las mejoras incluídas en cada nueva versión que han permitido hacerlo más amigable para el usuario a la vez que más efectivo para el desarrollo web.

· React: Facebook introdujo React en 2013 y ha sido una tendencia en los últimos cuatro años. Aunque los últimos lanzamientos no han incluído ninguna mejora importante, aún se jacta de su mayor base de conocimientos, apoyo y comunidad.

· Ionic: Ionic es un framework para construir y desplegar aplicaciones multiplataforma (Android, iOS, web). Muchos desarrolladores suelen prefieren este framework especialmente por sus componentes intuitivos de UI.

Entre los frameworks Javascript que se encuentran entre el top 3 para desarrollo **backend** están:

· Express: los desarrolladores prefieren este framework para construir APIs y aplicaciones web por robustez y minimalismo. Han habido 7 lanzamientos para mejorar el rendimiento de este framework y, sin duda, mantiene la posición de los frameworks javascript en tendencia incluso para el 2020-21.

· Next.js: se trata de un framework pequeño, en particular para el desarrollo de aplicaciones en React. Next.js permite desarrollar aplicaciones complejas con una mínima codificación. Este año, React y Next.js van de la mano y gracias a la creciente popularidad de React, está impulsando la inclinación de los desarrolladores hacia el uso de Next.js.

· Meteor: se trata de un framework “isomórfico” de código abierto (aplicaciones que se ejecutan tanto en el cliente como en el servidor) basado en Javascript. Meteor ha crecido exponencialmente desde su lanzamiento en 2012.