Programación con Java Script II

Reactivos

Pregunta 1

Explica para qué sirve la herramienta npm en JavaScript.

@ Revisa la explicación del tema 1.

Opciones:

a. Para llamados asíncronos.
b. Para realizar llamados a servicios web.
c. Para agregar eventos a objetos en tiempo de programación.
d. Para instalar y actualizar dependencias en proyectos.

Pregunta 2

Explica para qué sirve la instrucción "npm init".

@ Revisa la explicación del tema 1.

Opciones:

a. Crea el archivo packaje.json, el cual contiene información importante del proyecto.
b. Instala nuevos paquetes en ambiente local.
c. Permite conocer la versión del npm.
d. Para actualizar sus paquetes y dependencias a versiones nuevas de lo ya instalado.

Pregunta 3

Resuelve qué utilizarías entre npm o Yarn para la instalación de paquetería en tu entorno de desarrollo cuando los aspectos de seguridad y velocidad son prioridad que cuidar en tu empresa.

@ Revisa la explicación del tema 1.

Opciones:

a. Npm
b. Cualquiera de los 2 es igual de confiable.
c. Yarn
d. La seguridad no es un aspecto de un gestionador de paquetes.

Pregunta 4

Identifica la alternativa de gestionador de paquetes npm para usarse en ambientes de desarrollo JavaScript

@ Revisa la explicación del tema 1.

Opciones:

a. Visual Studio Code
b. Yarn
c. Explorador web
d. No hay alternativa.

Pregunta 5

Detalla una alternativa eficiente para un proyecto de JavaScript que usa varios archivos y que crea un entregable a producción lo más ligero posible.

@ Revisa la explicación del tema 1.

Opciones:

a. Uso de Webpack para crear la versión a producción.
b. Reunir todos los archivos del proyecto en uno solo.
c. No hay opción de simplificar el proyecto a producción.
d. Cambiar de lenguaje de desarrollo a uno más moderno.

Pregunta 6

Resume el tipo de programación común en JavaScript que implica iniciar una tarea de larga duración y seguir respondiendo a otros eventos mientras la tarea anterior se ejecuta.

@ Revisa la explicación del tema 2.

Opciones:

a. Programación síncrona
b. Programación asíncrona
c. Programación funcional
d. Programación dinámica

Pregunta 7

Describe qué es un *callback* en JavaScript.

@ Revisa la explicación del tema 2.

Opciones:

a. Función que se pasa a otra función como un parámetro y luego se invoca dentro de la función externa.
b. Objeto que representa la realización o el fracaso de una operación asíncrona.
c. API que proporciona una interfaz para manipular partes de llamados HTTP.
d. Función sin nombre que ejecuta un código.

Pregunta 8

Elige el objeto que es el resultado de una operación asíncrona y se configura con dos llamadas para informar el éxito o el fallo de esta.

@ Revisa la explicación del tema 2.

Opciones:

a. XMLHttpRequest
b. Callback
c. Promise
d. async/await

Pregunta 9

Identifica la función que requiere la promesa para entregar un resultado.

@ Revisa la explicación del tema 2.

Opciones:

a. Reject
b. Catch
c. Resolve
d. Then

Pregunta 10

Recuerda las palabras reservadas con las que se puede convertir una función síncrona en asíncrona.

@ Revisa la explicación del tema 2.

Opciones:

a. Async/await
b. Try/catch
c. Get/Post
d. Open/Send

Pregunta 11

Menciona el tipo de dato en JavaScript que se puede utilizar para crear propiedades de objetos privadas de forma única.

@ Revisa la explicación del tema 3.

Opciones:

a. Number
b. Symbol
c. String
d. Object

Pregunta 12

Menciona qué es el tipo *set* en JavaScript.

@ Revisa la explicación del tema 3.

Opciones:

a. Es un objeto que representa un conjunto de valores conocidos como claves, donde cada clave tiene otro valor asociado.

b. Es una función sin nombre que ejecuta un bloque de código.
c. Es una colección de valores, donde no están ordenados ni indexados. La principal característica de este tipo es que no permite duplicados.
d. Es un objeto que implementa el protocolo de iteración a través del método next().

Pregunta 13

Selecciona qué tipo de objeto, característico de ES6, permite que las matrices, cadenas, objetos *set* y *map* puedan ser recorridos y su contenido pueda iterar con el bucle *for/of*.

@ Revisa la explicación del tema 3.

Opciones:

a. Symbol
b. addEventListener
c. this
d. Objetos iterables

Pregunta 14

Explica que es un objeto *proxy* en JavaScript.

@ Revisa la explicación del tema 3.

Opciones:

a. Este objeto es una colección de valores de manera única desordenada.
b. Objeto con colección de otros elementos donde cada uno contiene un par de valores: una clave y un valor en sí.
c. Permite asignar nombres de propiedades en tiempo de ejecución con el valor de una función o variable.
d. Este tipo de objeto permite crear un intermediario para otro objeto, el cual puede interceptar y redefinir operaciones fundamentales para dicho objeto.

Pregunta 15

Explica qué es una propiedad dinámica.

@ Revisa la explicación del tema 3.

Opciones:

a. Es la que se establece con *var* en lugar de *let*.
b. Es aquella que permite una función con el signo de = y > sin necesidad de nombre.
c. Propiedad de un objeto cuyo nombre no es una constante en tiempo de compilación, sino que proviene de una variable o es el valor de retorno de una función.
d. Es la que se establece dentro de llaves ({}) como parte de un objeto.

Pregunta 16

Resume la principal diferencia entre las palabras reservadas *let* y *var* en JavaScript.

@ Revisa la explicación del tema 4.

Opciones:

a. No hay diferencia entre ellas.
b. *Let* se usa para declarar variables limitando su alcance (*scope*) al bloque donde se está usando; *var* define una variable sin importar el ámbito del bloque.
c. *Var* es la manera más actual de declarar variables en lugar de *let*.
d. Con *let*, a diferencia de *var*, el alcance no es una limitante en la declaración de variables.

Pregunta 17

Explica qué es la palabra reservada this.

@ Revisa la explicación del tema 4.

Opciones:

a. Hace referencia o representa a un objeto que se llamará dentro de un bloque de código, como es el caso, por mencionar un ejemplo, de una función.
b. Este sólo se usa en lenguajes orientados a objeto, no en JavaScript.
c. *This* se utiliza para retornar un valor como resultado del llamado de una función.
d. *This* sirve para declarar una clase.

Pregunta 18

Identifica qué métodos se utilizarían para las ocasiones en que se quiere forzar a una función a usar un objeto como contexto sin tener una referencia de función de propiedad en el objeto.

@ Revisa la explicación del tema 4.

Opciones:

a. *join()* y *sort()*
b. *indexOf()* y *split()*
c. *call(), apply()* y *bind()*
d. *stringyfy()* y *json()*

Pregunta 19

Reconoce qué significa *scope* en javascript.

@ Revisa la explicación del tema 4.

Opciones:

a. Es el alcance que una variable tiene dentro del código y decide a qué variables se tiene acceso en cada parte del código, normalmente delimitado por llaves ({}).
b. Sintaxis o forma de escritura que nos permite extraer valores de un arreglo o propiedades de un objeto en secciones más pequeñas.
c. Es crear un nuevo objeto basado en una clase particular.
d. Es una técnica que permite a un programa iniciar una tarea de larga duración y seguir respondiendo a otros eventos mientras esa tarea se ejecuta.

Pregunta 20

Selecciona la palabra reservada para crear un objeto nuevo, al cual la referencia *this* se vincula.

@ Revisa la explicación del tema 4.

Opciones:

a. aync
b. function
c. switch
d. new

Pregunta 21

Explica qué significa AJAX.

@ Revisa la explicación del tema 5.

Opciones:

a. Es la herramienta que sirve para la instalación de paquetería.
b. Técnicas de desarrollo web que permiten que las aplicaciones web funcionen de forma asíncrona, procesando cualquier solicitud al servidor en segundo plano.
c. Nombre de una librería o *framework* de JavaScript.
d. Es una técnica para realizar llamado sólo con XML y especialmente de manera sincrónica.

Pregunta 22

Describe qué es el objeto XMLHttpRequest.

@ Revisa la explicación del tema 5.

Opciones:

a. Etiqueta que permite crear código JavaScript en una página web.
b. Es el modelo que sirve de interfaz de programación para los documentos HTML y XML.
c. Es un objeto nativo del navegador web para realizar solicitudes HTTP desde código de JavaScript, principalmente asíncronas.
d. El objeto XMLHttpRequest no existe en JavaScript.

Pregunta 23

Describe qué es el API *fetch*.

@ Revisa la explicación del tema 5.

Opciones:

a. API cuya interfaz permite realizar peticiones HTTP a servidores web de una manera eficaz, con un consumo de recursos mínimos y más potente.
b. API que permite un método incorporado para temporizar la ejecución de una determinada función.
c. API que permite llamados asíncronos, pero únicamente para servicios web *restful*.
d. Es una API exactamente igual al objeto XMLHttpRequest.

Pregunta 24

Miembro de la API *fetch* que permite atrapar errores durante su ejecución.

@ Revisa la explicación del tema 5.

Opciones:

a. Then
b. Catch
c. Stringify
d. Foreach

Pregunta 25

Describe la principal diferencia en el funcionamiento entre el objeto XMLHttpRequest y el API *fetch*.

@ Revisa la explicación del tema 5.

Opciones:

a. No existe diferencia en su funcionamiento.
b. La única diferencia es su sintaxis.
c. Son elementos totalmente distintos en los resultados que proporcionan, así que no se pueden comparar.
d. XMLHttpRequest es un tipo basado en un *callback*, mientras que *fetch* está basado en una promesa (*promise*) e integra conceptos avanzados de HTTP.

Pregunta 26

Explica qué es una API.

@ Revisa la explicación del tema 6.

Opciones:

a. Es un lenguaje de programación para el desarrollo de nuevas funcionalidades a detalle.
b. Es un llamado a un código intermediario que traduce peticiones para integrar nuevas funcionalidades a un *software* existente.
c. Es el nombre a una funcionalidad interna parte del *software* desarrollado.
d. API es una manera de hacer llamados asíncronos.

Pregunta 27

Selecciona el ejemplo de formato json:

@ Revisa la explicación del tema 6.

Opciones:

a. 01100001 01100010 01100011

b. <Catalogo>
<Libro id="101">
<Autor>Autor1</Author>
<Titulo>Desarrollo Software</Title>
</Libro>
<Libro id="102">
<Autor>Autor2</Author>
<Titulo>Programacion Avanzada</Title>
</Libro>
</Catalogo>

c. Dato1,Valor1, Dato2, Valor2

d. {
"llave1": "valor1",
"llave2": "valor2",
"llave3": "valor3",
"llave4": 4,
}

Pregunta 28

Explica para que sirve la función JSON.parse().

@ Revisa la explicación del tema 6.

Opciones:

a. Permite tomar una cadena de caracteres que tenga la estructura JSON para convertirla en un objeto.
b. Convierte un objeto JSON JavaScript a un *string* JSON.
c. Realiza un llamado asíncrono a un recurso web.
d. Realiza llamados HTTP *get*..

Pregunta 29

Describe qué es un servicio REST en el entorno del desarrollo web.

@ Revisa la explicación del tema 6.

Opciones:

a. Cualquier servicio web que dé un resultado.
b. Son servicios web para ser llamados asíncronamente.
c. Es una interfaz para conectar sistemas basados en el protocolo HTTP.
d. Son servicios para ser consumidos exclusivamente con JavaScript mediante el objeto XMLHttpRequest o el API *fetch*.

Pregunta 30

Selecciona la opción que contenga sólo verbos HTTP:

@ Revisa la explicación del tema 6.

Opciones:

a. Get, fetch, post, ajax
b. Get, post, put, delete
c. Resolve, reject, then, catch
d. Get, then, put, delete

Pregunta 31

Elige la acción que se realiza mediante herramientas que permiten reescribir código en una estructura que pueda ser reconocida por diferentes plataformas web.

@ Revisa la explicación del tema 7.

Opciones:

a. Depuración
b. Codificación
c. Instalación de paquetería
d. Transpilación

Pregunta 32

Explica qué es en programación la herramienta Babel.

@ Revisa la explicación del tema 7.

Opciones:

a. Herramienta “transpiladora” que funciona como traductor. Se considera una “herramienta en cadena” que permite trasladar sintaxis de código fuente de las versiones de JavaScript más recientes para que se utilicen en ambientes web de versiones anteriores.
b. Herramienta para mostrar código en un explorador web, de forma que, al modificarlo, refleja sus resultados en el explorador de manera automática.
c. Entorno de desarrollo integral para creación de *software*.
d. Herramienta para el uso de verbos HTTP.

Pregunta 33

Explica para qué sirve un *polyfill*.

@ Revisa la explicación del tema 7.

Opciones:

a. Es crear pruebas que validen porciones particulares de código.
b. Se utiliza para describir un extracto de código JavaScript que permite que los navegadores antiguos acepten una funcionalidad moderna de manera nativa.
c. Es la manera de llamar en paralelo una tarea para que otras sigan operando.
d. Es la forma de escribir variables o nombres de funciones donde todas sus letras son minúsculas, con excepción de la primera letra de cada palabra que forma dicho nombre.

Pregunta 34

Selecciona la opción donde podemos verificar si una función de JavaScript es soportada por un explorador web específico.

@ Revisa la explicación del tema 7.

Opciones:

a. babeljs.io
b. nodejs.org
c. caniuse.com
d. javascript.com

Pregunta 35

Decide qué alternativa es la mejor si tienes que dar una presentación de tu aplicativo y descubres que donde lo demostrarás (un equipo de cómputo en oficinas de tu cliente) no tienen un explorador web actualizado para cierta función avanzada en JavaScript dentro de tu código.

@ Revisa la explicación del tema 7.

Opciones:

a. Utilizaría herramientas de transpilación y *polyfills* para la conversión y manejo de código nuevo en exploradores no actualizados.
b. Suspendería la demostración hasta desarrollar manualmente el código para que se adapte al explorador web del cliente.
c. Expresaría que el cliente tiene que actualizar sus exploradores web de su compañía.
d. Ejecutaría mi código sabiendo que fallará explicando que la razón del fallo es usar un explorador web antiguo.

Pregunta 36

Selecciona el tipo de prueba que frecuentemente se realiza para comprobar secciones o bloques de código JavaScript.

@ Revisa la explicación del tema 8.

Opciones:

a. Pruebas de integración

b. Pruebas unitarias
c. Pruebas UAT
d. Pruebas beta.

Pregunta 37

Describe qué es Jest.

@ Revisa la explicación del tema 8.

Opciones:

a. Traductor de código moderno para navegadores web no actualizados
b. Es la instrucción para crear un nuevo objeto promesa.
c. Es un *framework* de JavaScript de código abierto diseñado para simplificar y agilizar el proceso de escritura y ejecución de pruebas.
d. Permite crear un llamado asíncrono y obtener su resultado con las funciones *then()*.

Pregunta 38

Explica, para el uso de Jest en pruebas unitarias en Javascript, ¿qué es un *matcher*?

@ Revisa la explicación del tema 8.

Opciones:

a. Permite hacer una pregunta de un “sí” lógico.
b. Se usa en conjunto con una o varias sentencias *case* para evaluar un resultado y determinar la acción que procede.
c. Es una instrucción necesaria para instalar el paquete de Jest.l.
d. Funciones para comparar los resultados del código con los valores esperados.

Pregunta 39

Selecciona el *matcher* para valor tipo *string*.

@ Revisa la explicación del tema 8.

Opciones:

a. toBeGreaterThanOrEqual(valor)
b. toBeLessThanOrEqual(valor)
c. toBeLessThan(valor)
d. not.toMatch(valor)

Pregunta 40

¿Qué produce el parámetro *coverage* en la siguiente instrucción en el package.json?

```
"scripts": {
"test": "jest --coverage"
},
```

@ Revisa la explicación del tema 8.

Opciones:

a. No pasa nada.
b. Se realiza la prueba rápidamente.
c. Se genera evidencia en formato html justo después de ejecutar la prueba.
d. Genera evidencia de la prueba en formato XML