Guia de Examen Modulo 4 Javascript

1. JavaScript (abreviado comúnmente JS) es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos,basado en prototipos, imperativo, débilmente tipado y dinámico.
2. Se utiliza principalmente del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas y JavaScript del lado del servidor (Server-side JavaScript o SSJS).
3. Su uso en aplicaciones externas a la web, por ejemplo en documentos PDF, aplicaciones de escritorio (mayoritariamente widgets) es también significativo.
4. Desde 2012, todos los navegadores modernos soportan completamente ECMAScript 5.1, una versión de JavaScript. Los navegadores más antiguos soportan por lo menos ECMAScript . La sexta edición se liberó en julio de 2015
5. JavaScript se diseñó con una sintaxis similar a C,aunque adopta nombres y convenciones del lenguaje de programación Java. Sin embargo, Java y JavaScript tienen semánticas y propósitos diferentes.
6. Su relación es puramente comercial, tras la compra del creador de Java (Sun Microsystems) de Nestcape Navigator (creador de LiveScript) y el cambio de nombre del lenguaje de programación.
7. Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del Document Object Model (DOM).
8. Javascript es el único lenguaje de programación que entienden de forma nativa los navegadores.
9. Tradicionalmente se venía utilizando en páginas web HTML para realizar operaciones y únicamente en el marco de la aplicación cliente, sin acceso a funciones del servidor.
10. Actualmente es ampliamente utilizado para enviar y recibir información del servidor junto con ayuda de otras tecnologías como AJAX. JavaScript se interpreta en el agente de usuario al mismo tiempo que las sentencias van descargándose junto con el código HTML.
11. Desde el lanzamiento en junio de 1997 del estándar ECMAScript 1, han existido las versiones 2, 3 y 5, que es la más usada actualmente (la 4 se abandonó). En junio de 2015 se cerró y publicó la versión ECMAScript 6.
12. JavaScript fue desarrollado originalmente por Brendan Eich de Netscape con el nombre de Mocha, el cual fue renombrado posteriormente a LiveScript, para finalmente quedar como JavaScript.
13. «JAVASCRIPT» es una marca registrada de Oracle Corporation. Es usada con licencia por los productos creados por Netscape Communications y entidades actuales como la Fundación Mozilla.
14. Las siguientes características son comunes a todas las implementaciones que se ajustan al estándar ECMAScript, a menos que especifique explícitamente en caso contrario.
15. Imperativo y estructurado: JavaScript es compatible con gran parte de la estructura de programación de C (por ejemplo, sentencias if, bucles for, sentencias switch, etc.). Con una salvedad, en parte: en C, el ámbito de las variables alcanza al bloque en el cual fueron definidas; sin embargo JavaScript no es compatible con esto, puesto que el ámbito de las variables es el de la función en la cual fueron declaradas. Esto cambia con la versión de ECMAScript 2015, ya que añade compatibilidad con block scoping por medio de la palabra clave let. Como en C, JavaScript

hace distinción entre expresiones y sentencias. Una diferencia sintáctica con respecto a C es la inserción automática de punto y coma, es decir, en JavaScript los puntos y coma que finalizan una sentencia pueden ser omitidos.

16. Tipado dinámico: Como en la mayoría de lenguajes de scripting, el tipo está asociado al valor, no a la variable. Por ejemplo, una variable x en un momento dado puede estar ligada a un número y más adelante, religada a una cadena. JavaScript es compatible con varias formas de comprobar el tipo de un objeto, incluyendo duck typing.25 Una forma de saberlo es por medio de la palabra clave typeof.
17. Objetual : JavaScript está formado casi en su totalidad por objetos. Los objetos en JavaScript son arrays asociativos, mejorados con la inclusión de prototipos (ver más adelante). Los nombres de las propiedades de los objetos son claves de tipo cadena: obj.x = 10 y obj['x'] = 10 son equivalentes, siendo azúcar sintáctico la notación con punto. Las propiedades y sus valores pueden ser creados, cambiados o eliminados en tiempo de ejecución. La mayoría de propiedades de un objeto (y aquellas que son incluidas por la cadena de la herencia prototípica) pueden ser enumeradas por medio de la instrucción de bucle for... in. JavaScript tiene un pequeño número de objetos predefinidos como son Function y Date.
18. Evaluación en tiempo de ejecución: JavaScript incluye la función eval que permite evaluar expresiones expresadas como cadenas en tiempo de ejecución. Por ello se recomienda que eval sea utilizado con precaución y que se opte por utilizar la función JSON.parse() en la medida de lo posible, pues puede resultar mucho más segura.
19. Funciones de primera clase: A las funciones se les suele llamar ciudadanos de primera clase; son objetos en sí mismos. Como tal, poseen propiedades y métodos, como .call() y .bind(). Una función anidada es una función definida dentro de otra. Esta es creada cada vez que la función externa es invocada. Además, cada función creada forma una clausura; es el resultado de evaluar un ámbito conteniendo en una o más variables dependientes de otro ámbito externo, incluyendo constantes, variables locales y argumentos de la función externa llamante. El resultado de la evaluación de dicha clausura forma parte del estado interno de cada objeto función, incluso después de que la función exterior concluya su evaluación.
20. Prototipos: JavaScript usa prototipos en vez de clases para el uso de herencia. Es posible llegar a emular muchas de las características que proporcionan las clases en lenguajes orientados a objetos tradicionales por medio de prototipos en JavaScript.
21. Las funciones también se comportan como constructores. Prefijar una llamada a la función con la palabra clave new crear una nueva instancia de un prototipo, que heredan propiedades y métodos del constructor (incluidas las propiedades del prototipo de Object). ECMAScript 5 ofrece el método Object.create, permitiendo la creación explícita de una instancia sin tener que heredar automáticamente del prototipo de Object (en entornos antiguos puede aparecer el prototipo del objeto creado como null).La propiedad prototype del constructor determina el objeto usado para el prototipo interno de los nuevos objetos creados. Se pueden añadir nuevos métodos modificando el prototipo del objeto usado como constructor. Constructores predefinidos en JavaScript, como Array u Object, también tienen prototipos que pueden ser modificados. Aunque esto sea posible se considera una mala práctica modificar el prototipo de Object ya que la mayoría de los objetos en Javascript heredan los métodos y propiedades del objeto prototype, objetos los cuales pueden esperar que estos no hayan sido modificados.

22. JavaScript normalmente depende del entorno en el que se ejecute (por ejemplo, en un navegador web) para ofrecer objetos y métodos por los que los scripts pueden interactuar con el "mundo exterior". De hecho, depende del entorno para ser capaz de proporcionar la capacidad de incluir o importar scripts (por ejemplo, en HTML por medio del tag <script>). (Esto no es una característica del lenguaje, pero es común en la mayoría de las implementaciones de JavaScript.)
23. Funciones variádicas: Un número indefinido de parámetros pueden ser pasados a la función. La función puede acceder a ellos a través de los parámetros o también a través del objeto local arguments. Las funciones variádicas también pueden ser creadas usando el método .apply().
24. Funciones como métodos: A diferencia de muchos lenguajes orientados a objetos, no hay distinción entre la definición de función y la definición de método. Más bien, la distinción se produce durante la llamada a la función; una función puede ser llamada como un método. Cuando una función es llamada como un método de un objeto, la palabra clave this, que es una variable local a la función, representa al objeto que invocó dicha función.
25. Arrays y la definición literal de objetos: Al igual que muchos lenguajes de script, arrays y objetos (arrays asociativos en otros idiomas) pueden ser creados con una sintaxis abreviada. De hecho, estos literales forman la base del formato de datos JSON.
26. Expresiones regulares: JavaScript también es compatible con expresiones regulares de una manera similar a Perl, que proporcionan una sintaxis concisa y poderosa para la manipulación de texto que es más sofisticado que las funciones incorporadas a los objetos de tipo string.
27. Extensiones específicas del fabricante: JavaScript se encuentra oficialmente bajo la organización de Mozilla Foundation, y periódicamente se añaden nuevas características del lenguaje. Sin embargo, sólo algunos motores JavaScript son compatibles con estas características:
28. Las propiedades get y set (también compatibles con WebKit, Opera,ActionScript y Rhino).

- Cláusulas catch condicionales.
- Protocolo iterador adoptado de Python.
- Corrutinas también adoptadas de Python.
- Generación de listas y expresiones por comprensión también adoptado de Python.
- Establecer el ámbito a bloque a través de la palabra clave let.
- Desestructuración de arrays y objetos (forma limita de emparejamiento de patrones).
- Expresiones concretas en funciones (function(args) expr).
- ECMAScript para XML (E4X), una extensión que añade compatibilidad nativa XML a ECMAScript.

29. El uso más común de JavaScript es escribir funciones embebidas o incluidas en páginas HTML y que interactúan con el Document Object Model (DOM o Modelo de Objetos del Documento) de la página.
30. Cargar nuevo contenido para la página o enviar datos al servidor a través de AJAX sin necesidad de recargar la página (por ejemplo, una red social puede permitir al usuario enviar actualizaciones de estado sin salir de la página).
31. Animación de los elementos de página, hacerlos desaparecer, cambiar su tamaño, moverlos, etc.
32. Contenido interactivo, por ejemplo, juegos y reproducción de audio y vídeo.
33. Validación de los valores de entrada de un formulario web para asegurarse de que son aceptables antes de ser enviado al servidor.

34. Transmisión de información sobre los hábitos de lectura de los usuarios y las actividades de navegación a varios sitios web. Las páginas Web con frecuencia lo hacen para hacer análisis web, seguimiento de anuncios, la personalización o para otros fines.
35. Debido a que JavaScript se ejecuta en entornos muy variados, una parte importante de las pruebas y la depuración es probar y verificar que el código JavaScript funciona correctamente en múltiples navegadores.
36. Accesibilidad: Suponiendo que el usuario no haya desactivado la ejecución de código JavaScript, en el lado del cliente JavaScript debe ser escrito tanto con el propósito de mejorar las experiencias de los visitantes con discapacidad visual o física, como el de evitar ocultar información a estos visitantes.
37. Seguridad: JavaScript y el DOM permite que existan programadores que hagan un uso inapropiado para introducir scripts que ejecuten código con contenido malicioso sin el consentimiento del usuario y que pueda así comprometer su seguridad.
38. Vulnerabilidades cross-site: Un problema común de seguridad en JavaScript es el cross-site scripting o XSS, una violación de la política de mismo origen. Las vulnerabilidades XSS permiten a un atacante inyectar código JavaScript en páginas web visitadas por el usuario.
39. Herramientas de desarrollo: En JavaScript, disponer de un depurador se convierte en necesario cuando se desarrollan grandes aplicaciones, no triviales. Dado que puede haber diferencias de implementación entre los diferentes navegadores (especialmente en cuanto al DOM), es útil tener acceso a un depurador para cada uno de los navegadores a los cuales nuestra aplicación web irá dirigido.
40. Los depuradores web están disponibles para Internet Explorer, Firefox, Safari, Google Chrome y Opera.
41. Document Object Model o DOM ('Modelo de Objetos del Documento' o 'Modelo en Objetos para la Representación de Documentos') es esencialmente una interfaz de plataforma que proporciona un conjunto estándar de objetos para representar documentos HTML, XHTML y XML.
42. El DOM permite el acceso dinámico a través de la programación para acceder, añadir y cambiar dinámicamente contenido estructurado en documentos con lenguajes como ECMAScript (JavaScript).
43. Un evento desde el punto de vista computacional ocurre cuando cambia alguna situación en la computadora como, por ejemplo, la posición del ratón, la pulsación de alguna tecla, los contenidos de alguna de las memorias, la condición de la pantalla, etc. En la creación de páginas web estos eventos representan la interacción del usuario con la computadora.
44. AJAX, acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones web asíncronas. Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible interactuar con el servidor sin necesidad de recargar la página web, mejorando la interactividad, velocidad y usabilidad en las aplicaciones.