ORT אורט ARGENTINA

Educando para la vida

Programación en Nuevas Tecnologías 2

Javascript, ES6

ES5, ES6, ES2016, ES2017, ES.Next

- ECMAScript vs JavaScript
- ¿Que soportan la mayoría de los entornos?
- Transpilers (Babel, TypeScript, CoffeeScript, etc.)
- ¿Que sintaxis debería usar?

Closures

- Funciones declaradas dentro del ámbito léxico.
- Tienen acceso a variables declaradas en la función padre.
- Es posible gracias al sistema de ámbitos de Javascript.

Funciones de primer nivel

- Funciones tratadas de la misma manera como si fuesen variables.
  - Pueden ser asignadas a variables, valores de un array o incluso valores de un objeto.
  - Puede ser pasado como argumento de otra función
  - Puede ser retornados desde funciones
- Permiten la creación de funciones de alto nivel.
  - Tomando una o más funciones como argumentos, o incluso devolviendo una función.
  - Ejemplos: map(), filter(), reduce()

Asincronicidad

- Javascript es un lenguaje sincronizado, ejecutado en un solo hilo.
- Una función que toma mucho tiempo correr va a causar problemas de performance y eventualmente congelar la página.
- Javascript tiene funciones que actúan de forma asíncrona.

Asincronicidad

- Ejecución en Pilas
- Browser APIs
- Función de encolado: Function queue
- Ciclo de eventos.

Asincronicidad

- Ejemplo de funciones asíncronas
  - setTimeout()
  - XMLHttpRequest(), jQuery.ajax(), fetch()
  - Llamadas a base de datos.

Callbacks

- Flujo de control con llamadas asíncronas.
- Ejecuta la función una vez la llamada asíncrona retorna un valor.

Promesas (Promises)

- Solución al problema de callbacks
- Te permite escribir código que asume que un valor es retornado dentro de una función completada.
- Solo necesita un solo manejador de errores.

Clases

- ES6+ permite la defínción de “clases” dentro de un entorno.
- En realidad son objetos con un prototype definido.
- Permite Herencia de clases (que no es mas que una herencia prototype).
- Utilización de conceptos básicos de Orientación a Objeto en JS/Nodejs.

Objeto "this"

- Hace referencia a un objeto o función en donde está siendo ejecutado
- En un contexto de ejecución global, hace referencia al objeto global/window
- Si la función es llamada como método de un objeto, `this` hace referencia al objeto padre.

Browser y el DOM

- El browser renderiza el HTML de una página web
- El HTML es en lenguaje con estructura de árbol.
- El Browser construye este árbol en memoria antes de renderizar la página.
- Ese árbol es llamado Document Object Model (DOM). Simplemente Document.
- El DOM puede ser modificado usando Javascript.