Antes de comenzar a escribir líneas de código entrar en angular o las características del ECMAScript

que yo necesito que ustedes conozcan déjenme darles esta pequeña presentación.

Así que pónganse cómodos y escúcheme por unos momentos JavaScript como lenguajes de programación ha

ido creciendo conforme los años y cada vez es más demandado y cada vez se utiliza más para crear aplicaciones

ya no sólo se utiliza para crear aplicaciones web ahora se crean Bakken se crean aplicaciones móviles

y otro montón de cosas.

El problema es que JavaScript no fue diseñado para crear aplicaciones de mediana y gran escala.

Originalmente sólo se creó para ser validaciones de formularios pero al ver que la necesidad de hacer

manipulaciones en el html antes de enviar la información al servidor fue creciendo de que cada vez se

podía hacer más cosas en JavaScript pero al no ser pensado para crear aplicaciones tan grandes surgieron

frameworks y librerías como Angola Albiach View entre otro montón.

Con el objetivo de ayudarnos a poder crear aplicaciones de mediana y gran escala que nos permitan sacar

el máximo provecho del poder que tienen JavaScript pero como les mencioné originalmente JavaScript fue

diseñado para tener unas cuantas líneas de código tal vez cientos de líneas de código por mucho.

Pero hoy en día es muy fácil superar miles de líneas de código y el problema es que Chabás que al ser

un lenguaje de programación de tipado débil es muy fácil cometer errores especialmente cuando las aplicaciones

empiezan a crecer en complejidad.

Vamos a sorprendernos un pequeño panorama ustedes llegan a una empresa les dan una función y les dice

Ok simplemente llama a la función calcular y eso se encarga de todo ok dicen ustedes perfecto.

Llega el momento de utilizarla y algo no funciona bien.

Puede ser que dicha función no esté bien documentada y nos toca entrar al código y todo lo vemos en

chino.

Bueno si ustedes entienden chino pues otro lenguaje que ustedes no conozcan pero puede ser que nos tomen

mucho tiempo encontrar el problema y bueno al fin del día nos damos cuenta de que lo que pasaba es que

estábamos mandando mal el argumento porque nos pedía digamos un objeto y estábamos mandando números

en su otra cosa.

Pero como ya que acepta prácticamente cualquier tipo de dato por eso nos costó encontrar este problema

ya que carece de muchas características como por ejemplo tipado de variables estricto.

Errores en tiempo de escritura autocompletado dependiendo de la variable métodos estáticos de programación

clases y módulos.

Aunque después del estándar del ECMAScript 6 ya los incorporó pero todavía hacen falta ciertas características

sería excelente que cuando nosotros estemos trabajando en JavaScript podamos tener el poder como Visual

Studio NET Vince Eclipse como estos.

Y es que a la hora de escribir el código les van diciendo Hey esta línea está mal.

Aquí te falta algo falta cerrar esto.

Eso no puede ser válido eso no se puede asignar a esta variable que si yo etcetc pero tener todo ese

poder en JavaScript sería súper conveniente.

Claro tenemos web Storm que es el otro que faltaba y que es un editor de código bastante poderoso que

a todo esto no sé si considerarlo un IDE porque también tiene formas de compilar entre otras cosas es

básicamente todo lo que ustedes necesitan.

Pero el problema es que Wheatstone no es gratuito pero a estas alturas existen otros programas como

Visual Studio Cold Atom entre otros que son muy poderosos para trabajar en JavaScript pero el problema

es que nos faltan todas esas características que ides robusto de desarrollo ya tienen ciertos problemas

que tienen JavaScript puede ser que tengamos errores porque la variable no estaba definida.

Errores porque el objeto no tiene la propiedad esperada.

Errores porque no se tiene idea de cómo trabaja el código de otros compañeros errores porque no se escriben

variables errores porque se escriben variables clases funciones constantes entre otras cosas.

También tenemos errores porque el código colisiona con el de otra persona errores porque el caché del

navegador mantiene los archivos viejos errores porque colocamos una mayúscula o minúscula en el lugar

incorrecto.

Errores porque simplemente no sabemos cómo funciona el código de los demás errores porque el desarrollo

no me dijo que eso no se podía hacer y lo peor de todo es que nos damos cuenta hasta que el programa

está corriendo.

Ahora todos estos problemas que les mencioné se pueden resolver con tags script con tags ustedes pueden

tener la experiencia de programar en un lenguaje al que ya estás acostumbrado como se llama Cèsar visual

basic entre otros que son lenguajes de programación de tipado estricto por lo cual el IDE de desarrollo

o editor de código les puede dar a ustedes toda la información de los errores de escritura junto con

sus sugerencias y cómo se llaman los métodos porque sabe qué cosas son permitidas y qué cosas no..

El problema es que Skype no corre directamente en el navegador web por lo cual hay que compilar o traducirlo

o transitarlo cualquiera de sus términos es básicamente lo mismo pero es tomar nuestro código tal script

y generarlo como un archivo de JavaScript ahora talves que es básicamente un super set de JavaScript.

A qué me refiero con eso.

Bueno tenemos el estándar de JavaScript como la mayoría lo conoce luego el estándar evolucionó y agregó

bastantes características nuevas en el ECMAScript 6.

Luego vino el G-7 el 8 y así sucesivamente pero al ser un super set envuelve todo de tal manera tú puedas

utilizar características nuevas de JavaScript y traducirlo al estándar global de la industria de JavaScript.

Qué sería básicamente el ECMAScript 5.

Pero hay que comprender que se va implementando características inclusive más rápido que los navegadores

web.

Pero todavía hay cosas del nuevo estándar de JavaScript que no son incorporadas en Skype pero la verdad

para hacerlos honestos yo no he tenido ese inconveniente de que necesito algo del ECMAScript nuevo que

acaba de salir y que yo no necesito implementar en Tawfiq usualmente eso no sucede y en lo personal

a mí no me ha pasado.

Veremos la siguiente gráfica la cual explica sobre la evolución de JavaScript en la industria y el estado

actual del estándar ECMAScript.

El problema es que si nos vamos allá afuera éste es el estado de JavaScript actualmente en la web.

Estamos muy cerca de que el estándar del ECMAScript 5 sea mundialmente aceptado en todos los navegadores

lo cual hace que el rango de productividad JavaScript sea entre el estándar de Legna scrip 5 y el estándar

de ECMAScript 2015 que sería el 16.

Pero hay muchas cosas del nuevo estándar de JavaScript que realmente valen la pena aprovechar y sería

Hynde ideal que nosotros estuviéramos en ese rango pero realmente no es así en la realidad estamos algo

como por este lado y esto conlleva a muchos muchos problemas porque puede ser que ustedes quieran trabajar

con características nuevas como leí sin que la waited o hacer peticiones Fetch trabajar con ser visual

entre otro montón de cosas pero eso todavía no es mundialmente aceptado.

Pero la ventaja es que contactes que ustedes pueden utilizar características nuevas de JavaScript y

trabajarlas con confianza porque se va a encargar de pasarlo bien al estándar de JavaScript que nosotros

especificamos y no haber que hacer ninguna modificación a nuestro código para que sea compatible con

versiones muy viejas de JavaScript.

Bueno eso sería toda la introducción que criarlas en esta presentación.

Espero que por lo menos tengamos una idea de por qué es bueno trabajar con Taxqueña pero ya en los próximos

videos empezaremos a tocar código.