CODER HOUSE

Clase 13. Programación Backend

Node.js como herramienta de desarrollo

OBJETIVOS DE LA CLASE

- Comprender el concepto de transpilador.
- Instalación y uso de Babel mediante Node.js.
- Instalación y uso de Typescript en un proyecto Node.js.

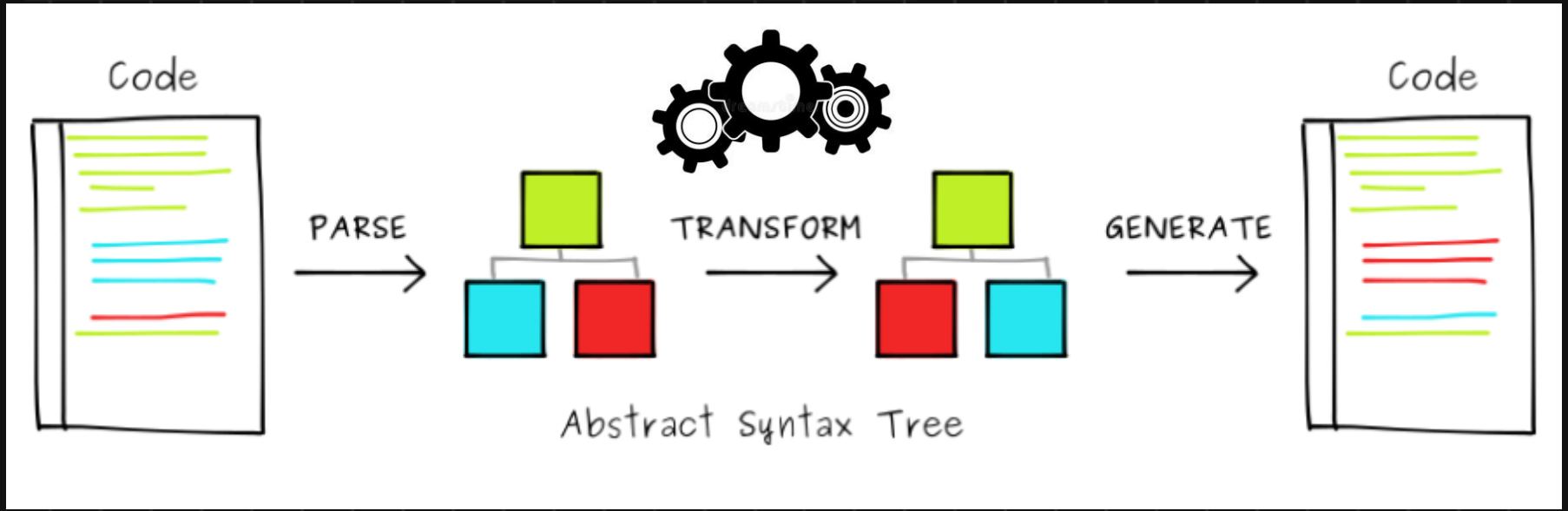
CODER HOUSE

CRONOGRAMA DEL CURSO

Clase 12	Clase 13	Clase 14
Aplicación chat con websocket	**Node.js como herramienta de desarrollo**	**Webpack: Module Bundler**
		1º ENTREGA PROYECTO FINAL

CODER HOUSE

Transpilador



Es un programa que **traduce código** escrito en **un lenguaje de programación a otro lenguaje**. En este **tipo de traductor** el lenguaje fuente es generalmente un lenguaje de **alto nivel** y el objeto un lenguaje de **bajo nivel**, como assembly o código máquina por ejemplo.

¿Qué es un Compilador?

¿Qué es un transpilador?

- Es un **tipo** especial de **compilador** que **traduce de un lenguaje fuente a otro fuente**, también de un nivel de abstracción parecido.
- Se diferencia de los compiladores tradicionales ya que estos últimos reciben como entrada archivos conteniendo código fuente y generan código máquina del más bajo nivel.
- La **transpilación**, que es la acción que realiza el *transpilador*, es un **caso particular** de la **compilación**.

BABEL

Ejemplos de transpiladores

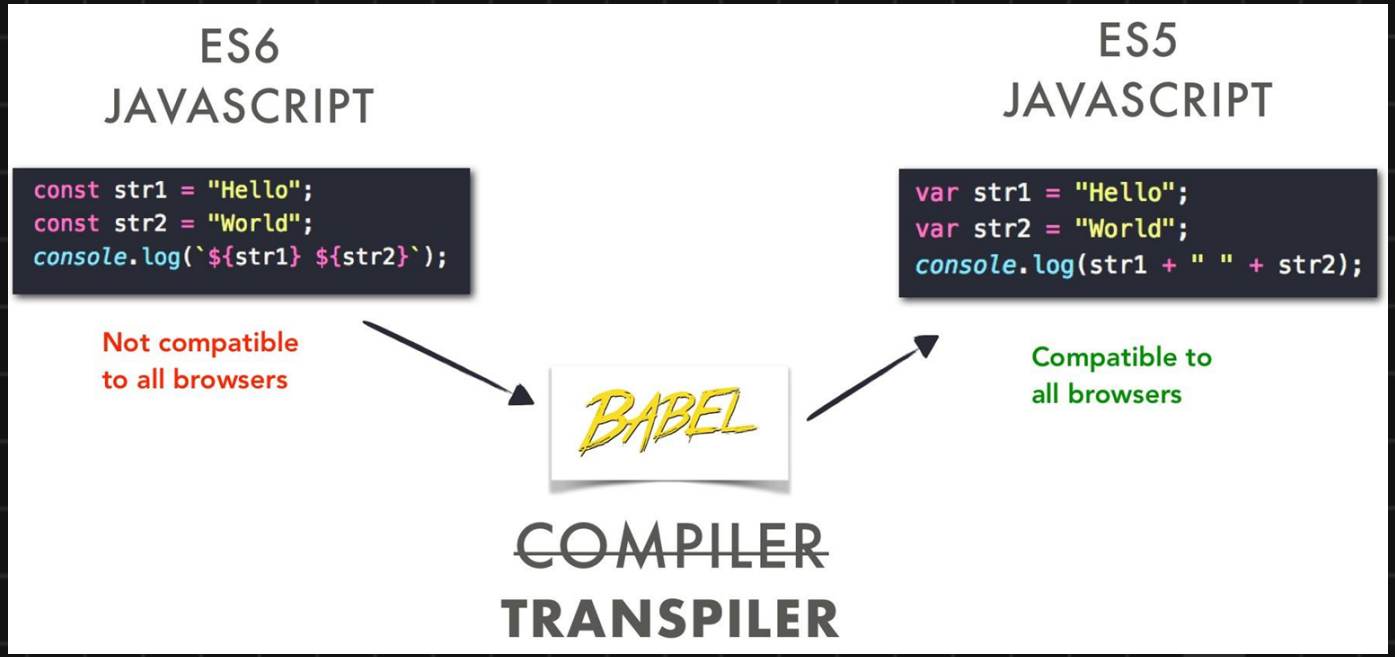
CODER HOUSE

- Los transpiladores y los compiladores traducen código desde un origen hacia un destino.
- La diferencia radica en la relación entre los lenguajes origen y destino de la traducción.
- El transpilador traduce código entre dos lenguajes que están al mismo nivel de abstracción, mientras que el compilador lo hace entre lenguajes de diferente nivel de abstracción

Diferencias entre transpiladores y compiladores

¿Alguna pregunta hasta ahora?

Babel



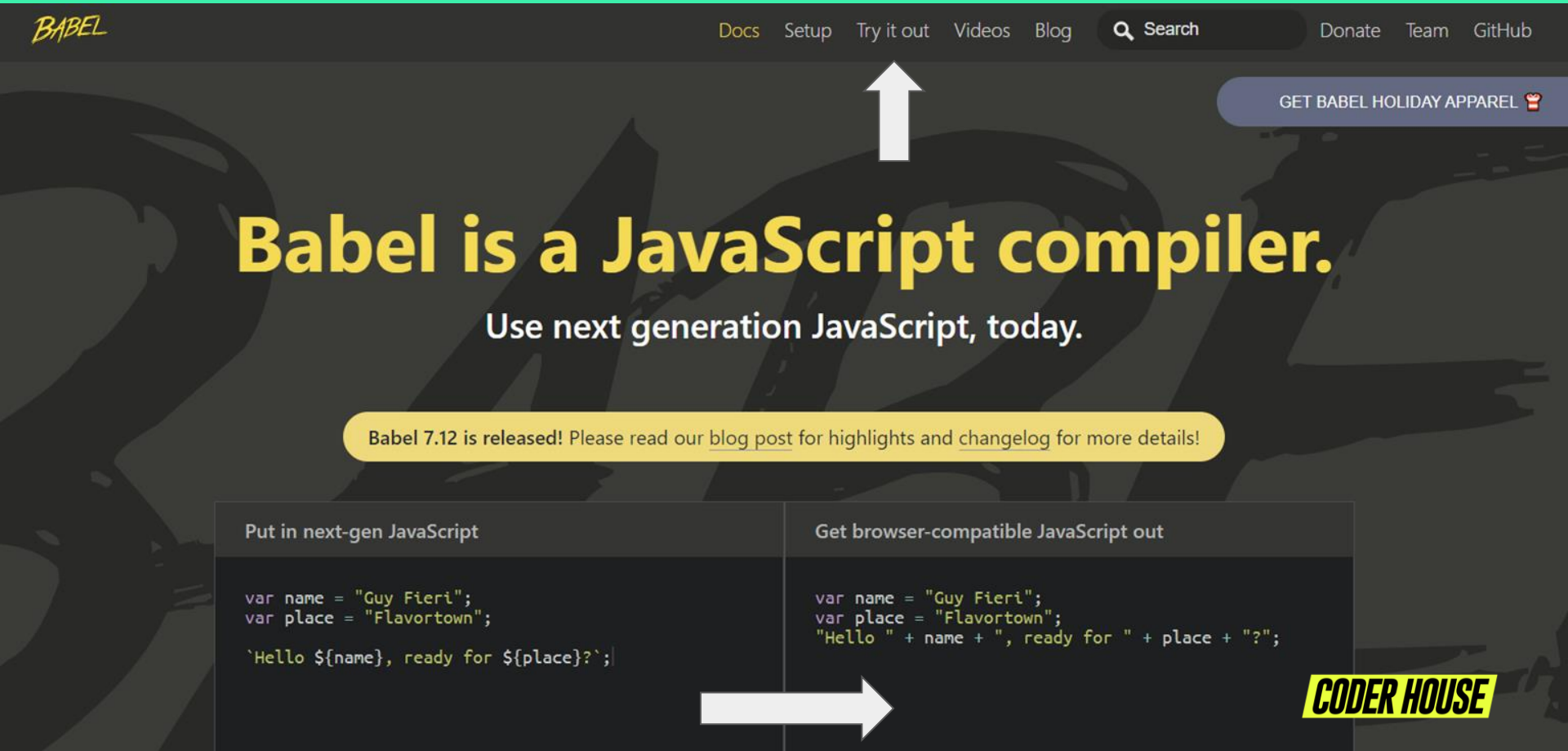
¿Qué es Babel?

- Babel es un transpilador que nos permite transformar nuestro código JS de última generación (o con funcionalidades extras) a JS que cualquier navegador o versión de Node.js entienda.
- Babel funciona mediante plugins con los cuales le indicamos cuál es la transformación que vamos a efectuar.
- Con el plugin babel-plugin-transform-es2015-arrow-functions podemos decirle que transforme las arrow functions de ECMAScript 2015 a funciones normales.

Vamos al IDE... 💻

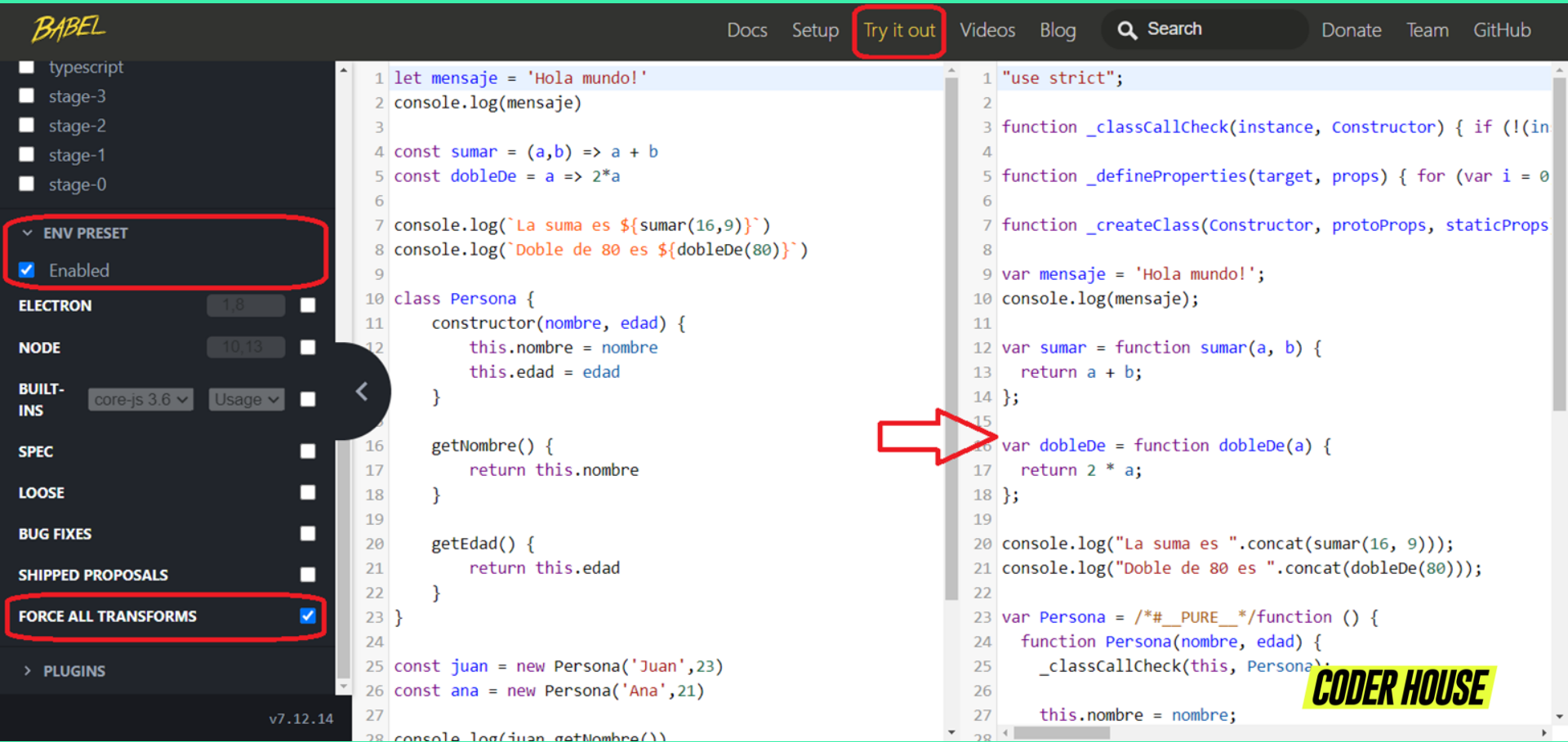
¿Alguna pregunta hasta ahora?

Babel: Web oficial https://babeljs.io/



CODER HOUSE

Babel: Online Transpiler ES6 -> JS5



COLOR ALEATORIO CON BABEL

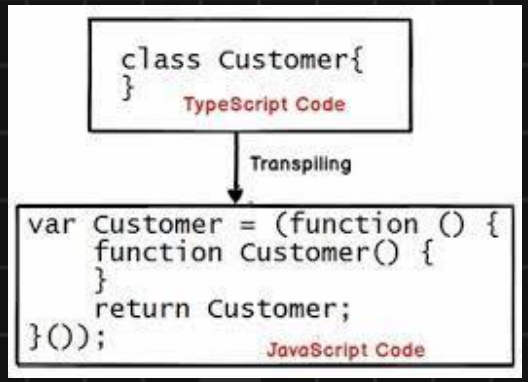
Tiempo: 10 minutos

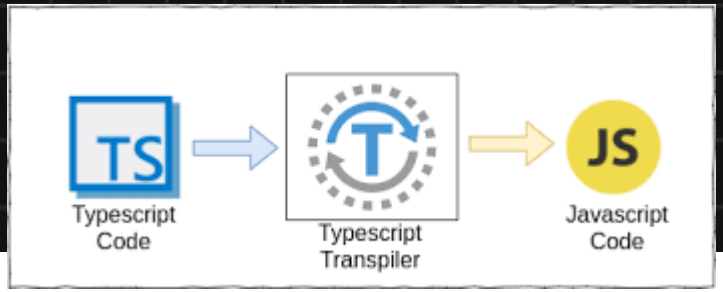
CODER HOUSE

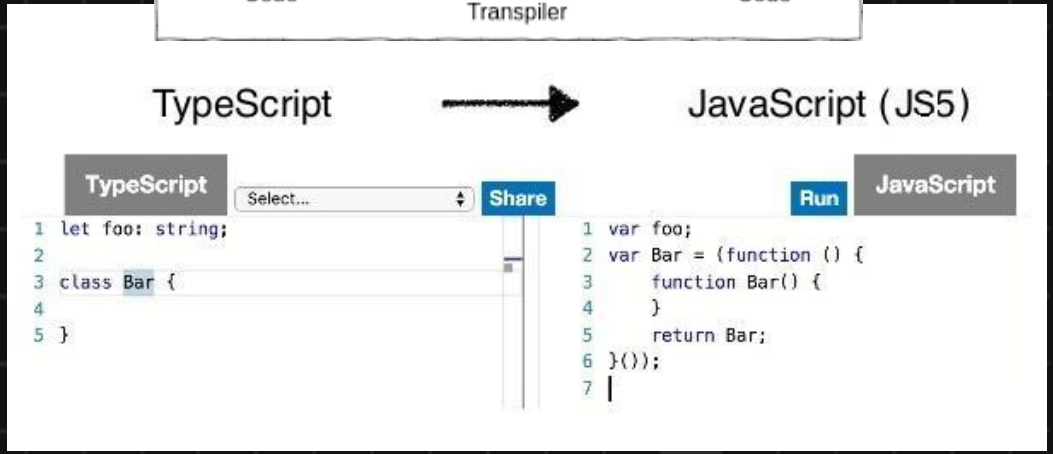
- Realizar un programa que genere un color aleatorio en formato RGB (canal rojo, verde y azul entre 0 y 255) y lo muestre por consola. Este estará implementado en un archivo llamado color.js
- La funcionalidad debe estar implementada dentro de una clase y deberá utilizar sintaxis ES6 (const, let, arrow function y template string).
- Convertir este código ES6 a JS5 con Babel online. Realizar esta conversión en forma automática dentro de un proyecto node.js que utilice Babel CLI

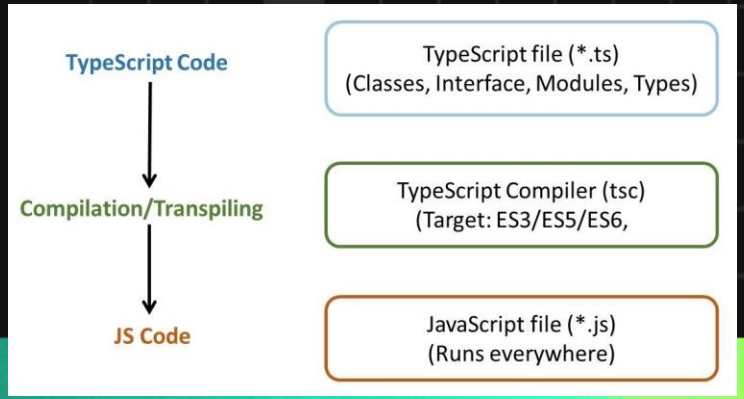
¿Alguna pregunta hasta ahora?

TSC: Typescript compiler

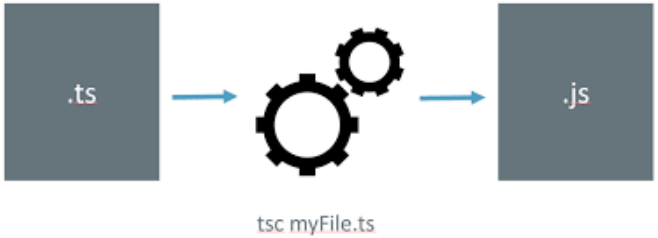








CODER HOUSE



¿Qué es TSC?

- Los archivos de TypeScript se compilan en JavaScript mediante TSC: el **compilador de TypeScript**

- TSC se puede instalar como paquete TypeScript a través de npm

- Para **transpilar** los archivos **Typescript a Javascript** lo hacemos a través de un proyecto en Node.js configurado como se muestra a continuación

Vamos al IDE... 💻

¿Alguna pregunta hasta ahora?

COLOR ALEATORIO CON TSC

Tiempo: 10 minutos

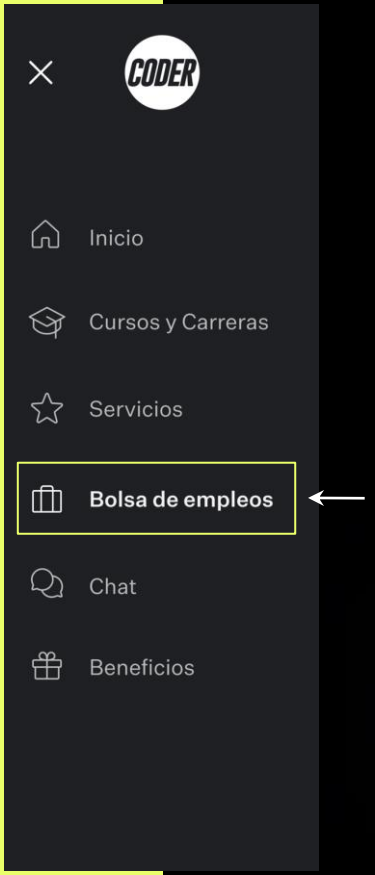
CODER HOUSE

- Realizar un proyecto TypeScript node.js que genere un **color aleatorio** en formato RGB (canal rojo, verde y azul entre 0 y 255) y lo muestre por consola.
- La funcionalidad debe estar implementada dentro de una clase en un archivo color.ts y deberá utilizar sintaxis Typescript tipada.
- El proyecto deberá convertir este código TS a JS5 en forma automática con TSC CLI

¿Alguna pregunta hasta ahora?

BREAK

¡5/10 MINUTOS Y VOLVEMOS!



Nuevo

¡Lanzamos la Bolsa de Empleos!

Un espacio para seguir **potenciando tu carrera** y que tengas más **oportunidades de inserción laboral**.

Podrás encontrar la **Bolsa de Empleos** en el menú izquierdo de la plataforma.

Te invitamos a conocerla y ¡postularte a tu futuro trabajo!

Conócela

CODERHOUSE

Módulos en ES6

CODER HOUSE

Módulos en ES6: Introducción

A partir de ES6 de Node.js admite definir archivos y proyectos como *módulos*. A diferencia de los archivos y proyectos comunes en JavaScript (“commonJs”), los módulos permiten ser importados en forma asincrónica en lugar de sincrónica, lo cual libera el hilo principal y mejora la performance de los programas (entre otras ventajas). Cuando se trata de proyectos, este cambio se puede realizar fácilmente desde el archivo *package.json*, agregando el siguiente par clave-valor: `"type": "module"`.

Módulos en ES6: Sintaxis

Una vez definido el proyecto como módulo, ya no podremos utilizar la función *require* para importar otros archivos, ni *module.exports* para exportar objetos hacia otros archivos. Para esto se utiliza la nueva sintaxis, según las siguientes equivalencias:

```
// MiClase.js
class MiClase { }
module.exports = MiClase

// libreria.js
function f() { }
module.exports = { f }

// main.js
const Clase = require('./MiClase.js')
const { f } = require('./libreria.js')
```

```
// MiClase.js
class MiClase { }
export default MiClase

// libreria.js
function f() { }
export { f }

// main.js
import Clase from './MiClase.js'
import { f } from './libreria.js'
```

CODER HOUSE

Módulos en ES6: Sintaxis

En caso de querer realizar una importación condicional, se puede import como función:

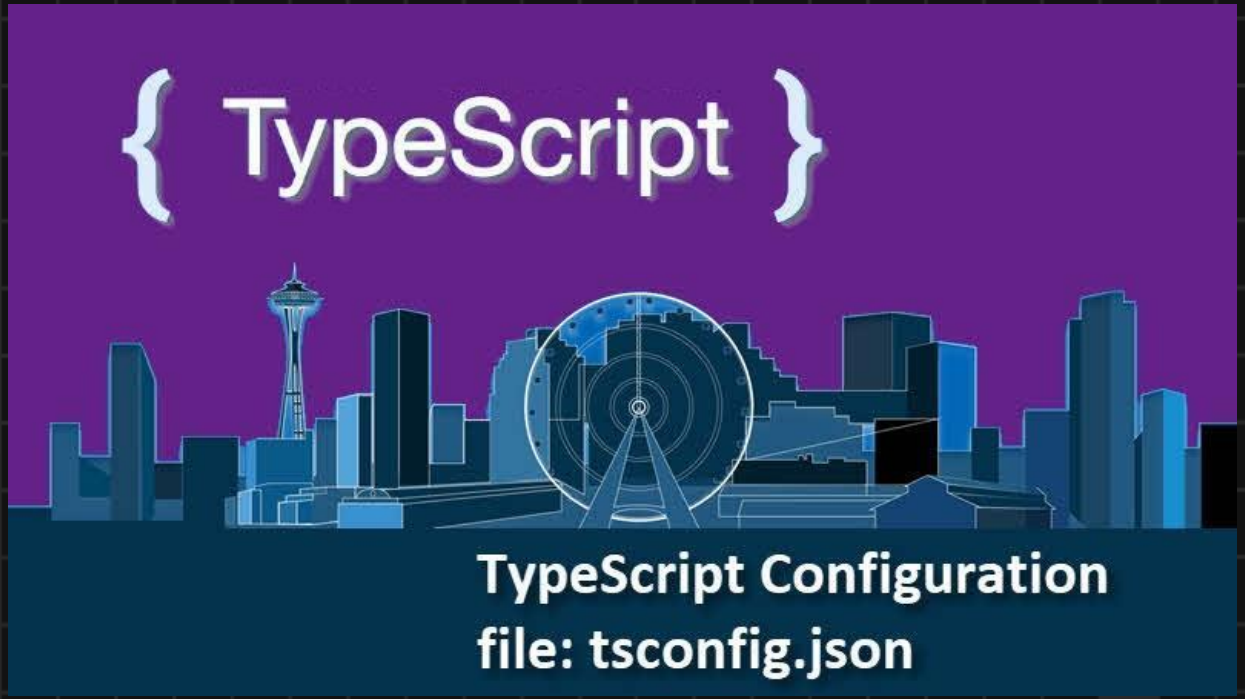
```
if (condicion) {
    const { default: Clase } = await import('./MiClase.js')
    const { f }  = await import('./libreria.js')
}
```

Notese que al ser asincrónica, devuelve una promesa, y admite el uso de *async/await*.

Dentro de los módulos es posible escribir await aún estando fuera de una función async (uso a nivel archivo), causando la espera de la resolución de la promesa como es de esperarse. A esta funcionalidad se la conoce como: ***Top-level Await****.*

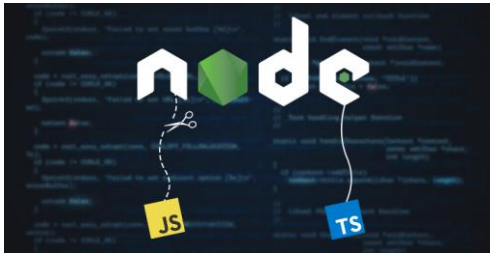
¿Alguna pregunta hasta ahora?

Uso avanzado de TSC



Creando un proyecto Typescript en node.js

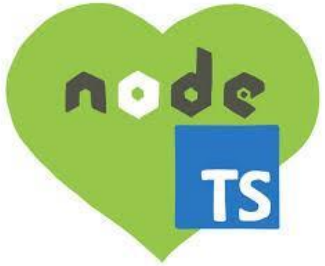
CODER HOUSE



Introducción

- **Node.js** es un entorno de tiempo de ejecución que hace que sea posible **escribir JavaScript en el lado del servidor**. Esto puede ser difícil a medida que la base de código crece debido a la naturaleza del lenguaje JavaScript: dinámico y con escritura débil.

- Los desarrolladores que llegan a JavaScript desde otros lenguajes a menudo se quejan sobre su falta de escritura estática fuerte, pero aquí es donde entra **TypeScript**, para cerrar esta brecha.



Introducción

- TypeScript puede **ayudar** a la hora de crear y gestionar **proyectos** JavaScript a **gran escala**. Puede verse como JavaScript con funciones adicionales como escritura estática fuerte, compilación y programación orientada a objetos.

- TypeScript es técnicamente un **superconjunto de JavaScript**, lo que significa que todo el código JavaScript es código TypeScript válido.

Vamos al IDE... 💻

¿PREGUNTAS? 

¡MUCHAS GRACIAS!

Resumen de lo visto en clase hoy:

-Transpiladores JS

-Babel

-TSC

OPINA Y VALORA ESTA CLASE

#DEMOCRATIZANDOLAEDUCACIÓN

CODER HOUSE