

**Introducción**

Ahora que ya tenemos todos los conceptos y práctica sobre métodos de array entonces haremos una integración de ambas Mesas de Trabajo vistas en esta semana

Para esta Mesa de Cierre vamos a estar modelando una Inmobiliaria que estará representada esta vez por un objeto literal, donde contendrá propiedades y métodos según se especifiquen a continuación

**Preparando estructuras de archivos**

Tomando como base la estructura de archivos, encontraremos los siguientes archivos

● **departamentos.json** que contiene la información necesaria

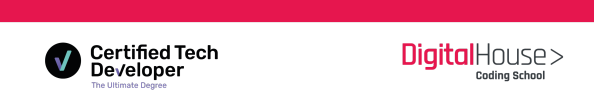
● **lecturaEscritura.js** un módulo con él que nos facilitará leer y escribir en

él archivo departamentos.json

**● app.js,** esta vez será un archivo vacío nuestro trabajo será desarrollar

nuestro código en él mismo

**1**

****

A continuación te planteamos varios desafíos que deberás resolver usando tu ingenio y lo aprendido hasta el momento.

Es probable que no puedas terminar todos estos ejercicios durante el tiempo que tiene la mesa de trabajo, no te preocupes, lo importante es que los termines

**Consigas**

1. Como primer paso necesitarás requerir él módulo lecturaEscritura y

asignarlo a una variable para poder utilizar sus funcionalidades, además

de hacer la lectura del archivo ***departamentos.json***, en detalle:

○ Requerir módulo lecturaEscritura y asignarlo a una variable llamada

por ejemplo archivos, esta contendrá los métodos del módulo

requerido de ahora en más

○ Leer él archivo departamentos.json utilizando la variable archivos

creada anteriormente con él método adecuado y asignar a una

variable llamada *arrayDepartamentos*

○ Comprobar los pasos anteriores imprimiendo la variable

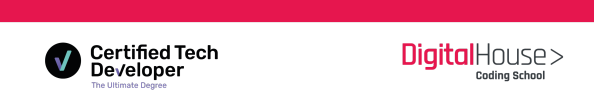
*arrayDepartamentos*. ejemplo : *console.log(arrayDepartamentos)*

2. Crear un objeto literal, que podemos llamar **inmobiliaria**, será nuestra

representación de la inmobiliaria (valga la redundancia) con su datos

(propiedades) y sus funcionalidades (métodos).

**2**

****

A. Agregar una propiedad llamada ***departamentos*** que contenga él

*arrayDepartamentos* obtenido en el punto anterior.

B. Agregar un método **listarDepartamentos** que reciba como parámetro un

array de departamentos y los imprima por consola.

Este método deberá imprimir por cada elemento:

○ El id.

○ El precioAlquiler.

○ disponible (en caso de ser true mostrar “Disponible” caso contrario

“Alquilado”)

○ La cantidad de ambientes

ej: “id: 2, precio $ 7500, está Disponible, 4 ambientes”

C. Agregar un método **buscarPorId** que permita buscar un departamento

en función de su id.

○ Este método recibirá por parámetro un number que representa él

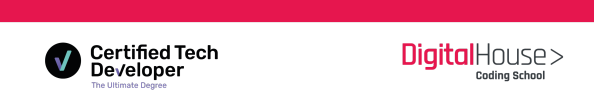
id del departamento a buscar

○ En caso de encontrar un departamento con él id buscado,

devolverá el objeto literal que lo representa.

○ En caso contrario devolverá *undefinded*

**3**

****

*Recordemos que Javascript tiene un método para hacer justamente lo que*

*necesitamos* ��*.*

D. Agregar un método **departamentosNoDisponibles** que permite filtrar

departamentos cuando su propiedad *disponible* sea igual a false, esto

quiere decir que están alquilados.

○ Este método devolverá un array con todos los departamentos que

cumplan la condición mencionada

○ en caso de no encontrar ningún que cumpla con la condición,

devolverá un array vacío

E. Agregar un método **departamentosDisponibles** que permite filtrar

departamentos cuando su propiedad *disponible* sea igual a true.

○ Este método devolverá un array con todos los departamentos que

cumplan la condición mencionada

○ en caso de no encontrar ningún que cumpla con la condición,

devolverá un array vacío

F. Agregar un método **filtrarPorAmbientes** que permite filtrar

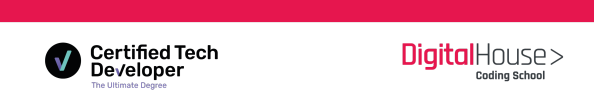
**departamentos**, siempre y cuando su propiedad *ambientes* sea mayor o

igual a una cantidad enviada como argumento.

○ Este método recibirá por parámetro un number que represente la

cantidad de *ambientes* mínimo.

**4**

****

○ Este método devolverá un array con todos los departamentos que

cumplan con la condición mencionada.

○ En caso de no encontrar ningún departamento que cumpla con la

condición, devolverá un array vacío.

G. Agregar un método **filtrarPorPrecio** que permite filtrar **departamentos**,

siempre y cuando su propiedad *precioAlquiler* sea menor o igual a él

precio enviado como argumento.

○ Este método recibirá por parámetro un number que represente él

*precioAlquiler* máximo.

○ Este método devolverá un array con todos los departamentos que

cumplan con la condición mencionada.

○ En caso de no encontrar ningún departamento que cumpla con la

condición, devolverá un array vacío.

○ Este método debe usar **departamentosDisponibles**, para buscar

incluir solamente aquellos departamentos que estén disponibles**.**

H. Agregar un método **rebajarPrecioAlquiler** que modifique él valor de

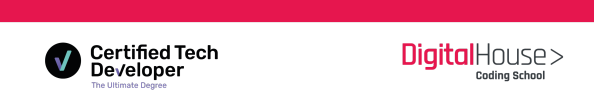
*precioAlquiler* de los departamentos No Alquilados.

○ Este método debe usar **departamentosDisponibles**, para buscar

incluir solamente aquellos departamentos que estén disponibles,

es decir, que no estén Alquilados**.**

**5**

****

○ Este método recibirá por parámetro un number que represente él

*porcentaje* que se desea rebajar a los departamentos no alquilados

por ejemplo un 3%.

○ Este método devolverá un array con todos los departamentos que

sufrieron la modificación del precioAlquiler

○ Este método debe realizar una persistencia de información, para

esto utilizaremos él método *escribirJson* de nuestro objeto

requerido en él primero punto.

archivos.escribirJson('departamentos',this.departamentos)

I. Agregar un método **calcularRecaudación** que calcule él valor que se

depositará en caja tomando en cuenta él precioAlquiler de los

departamentos Alquilados.

○ Este método devolverá un valor que represente la recaudación

total.

○ Este método debe usar **departamentosNoDisponibles**, para

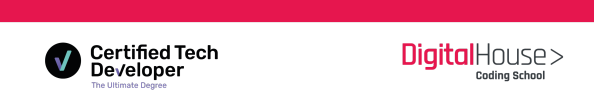
buscar incluir solamente aquellos departamentos que estén no

disponibles, es decir, Alquilados**.**

J. Agregar un método **ordenarPorPrecio** que ordene los departamentos de

menor a mayor según su precio.

**6**

****

○ El método recibirá como parámetro un array de departamentos.

○ Este método devolverá un array con todos los departamentos

ordenados por precio.

*Recordemos que Javascript tiene un método para hacer justamente lo que*

*necesitamos* ��*.*

**7**