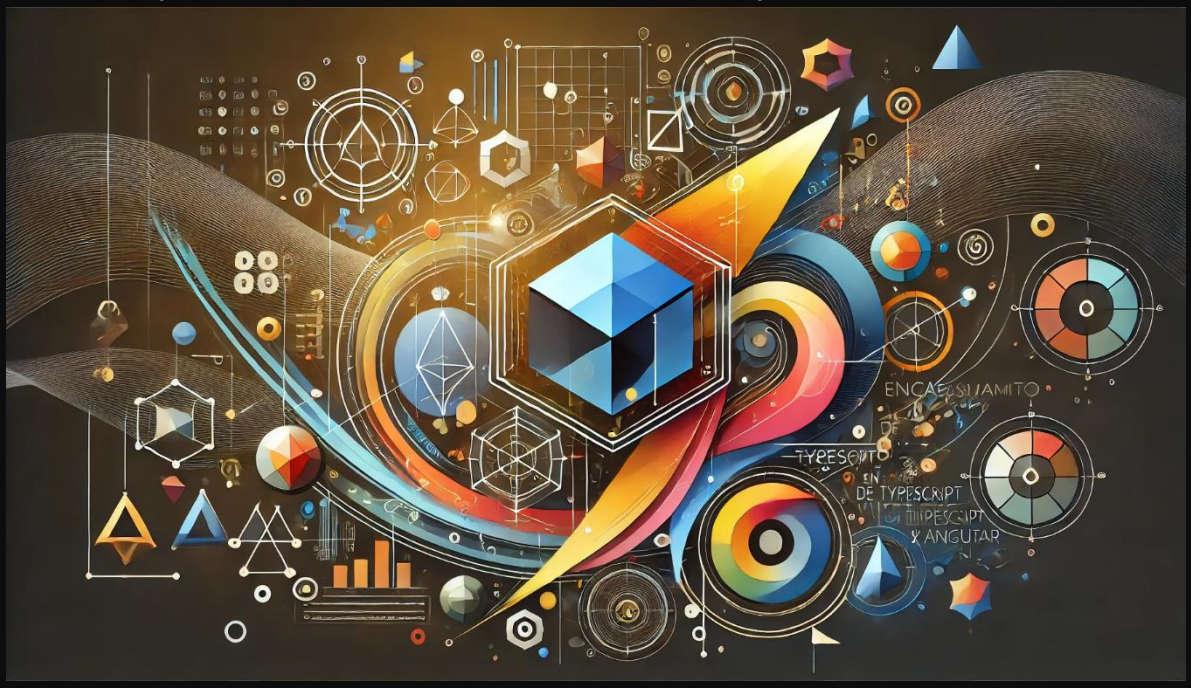
Encapsulamiento de Propiedades



Concepto de Encapsulamiento de Propiedades en TypeScript y Angular

En TypeScript y JavaScript, hay una diferencia significativa entre un método que usa la palabra clave `get` (con espacio) para definir un **accesor** (getter) y un método regular que actúa como parte del encapsulamiento.

1. Métodos `get` (Accessor)

- **TypeScript** permite definir métodos con la palabra clave `get`, lo que crea un **getter**.
- Estos métodos se acceden como si fueran propiedades en lugar de métodos. No necesitan paréntesis cuando se usan en plantillas o al llamarse en código.
- Los getters son útiles para encapsular lógica de acceso sin que el llamador sepa que está llamando a una función.

```
export class HijoComponent {
  private titulo = 'Titulo Componente Hijo';

  // Getter
  get mostrarTitulo() {
    return this.titulo;
  }
}
```

2. Métodos de Encapsulamiento

- Un método regular de encapsulamiento no usa espacio después de la palabra clave `get`.
- Para acceder a este método en la plantilla o en el código, necesitas usar paréntesis `()` porque es un método.
- Es útil para operaciones que necesitan recibir parámetros o realizar cálculos o acciones más complejas.

```
export class HijoComponent {
  private titulo = 'Titulo Componente Hijo';

  // Método regular
  getTitulo() {
    return this.titulo;
  }
}
```

Uso en la Plantilla Angular

Cuando usas estos métodos en la plantilla de Angular, se acceden de manera diferente:

Uso del Getter (`get`) de TypeScript:

```
<p>{{ mostrarTitulo }} método get como si fuera propiedad</p>
```

- Aquí, `mostrarTitulo` se usa como si fuera una propiedad, sin paréntesis.

Uso del Método Regular get de Encapsulamiento:

```
<p>{{ getTitulo() }} método get de encapsulamiento</p>
```

- Aquí, `getTitulo()` se llama como un método, por lo que requiere paréntesis.

Ejemplo Completo

Supongamos que tienes la siguiente clase `HijoComponent`:

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-hijo',
  standalone: true,
  imports: [],
  templateUrl: './hijo.component.html',
  styleUrl: './hijo.component.css'
})
```

```
export class HijoComponent {
  private titulo = 'Titulo Componente Hijo';

  // Se usa en la plantilla como si fuera una propiedad mas
  get mostrarTitulo(){
    return this.titulo;
  }

  // Se usa en la plantilla como un método (uso de paréntesis)
  getTitulo(){
    return this.titulo;
  }

}
```

Esta es la plantilla del componente hijo:

```
<p>{{ mostrarTitulo }} método get como si fuera propiedad</p>
<p>{{ getTitulo() }} método get de encapsulamiento</p>
```

Estos son los cambios en la plantilla del componente padre:

```
<p>Componente Padre</p>
<app-hijo/>
```

Explicación

1. `{{ mostrarTitulo }}`: Aquí estás accediendo al método `get mostrarTitulo` como si fuera una propiedad. No necesitas paréntesis porque el getter está diseñado para funcionar como una propiedad.
2. `{{ getTitulo() }}`: Aquí estás llamando al método `getTitulo()` de forma explícita como un método de encapsulamiento, por lo que necesitas usar paréntesis.

Diferencias Clave:

- **Sintaxis**: Los getters (`get`) de TypeScript no requieren paréntesis cuando se accede a ellos en la plantilla, mientras que los métodos regulares sí.
- **Propósito**: Los getters regulares de Encapsulamiento se usan cuando quieres que una función se comporte como una propiedad (sin necesidad de paréntesis), mientras que los métodos regulares son más versátiles y se usan para encapsular lógica que puede requerir parámetros o acciones más complejas.

Esto es algo que pertenece a TypeScript, aunque JavaScript también permite definir getters con `get`.

Saludos!

Ing. Ubaldo Acosta

Fundador de GlobalMentoring.com.mx