### **¿Qué es una Función Anónima Autoinvocada?**

Una función anónima autoinvocada (IIFE, por sus siglas en inglés) es una función que se define y se ejecuta inmediatamente en el mismo lugar donde se declara. No tiene un nombre asociado y, al ser autoinvocada, su código se ejecuta una sola vez.

**Sintaxis básica:**

TypeScript

(function() { // Código a ejecutar })();

Use code [with caution.](file:///C:/faq#coding)

### **¿Cómo funciona?**

1. **Definición:** Se define una función anónima envolviendo su código entre paréntesis.
2. **Invocación:** Se añaden otros paréntesis inmediatamente después de la definición, lo que provoca que la función se ejecute de forma instantánea.

### **¿Por qué usar IIFEs en TypeScript?**

* **Encapsulación:** Crea un ámbito local para las variables declaradas dentro de la función, evitando conflictos de nombres con variables globales.
* **Ejecución inmediata:** Garantiza que el código se ejecute una sola vez, sin necesidad de una llamada explícita.
* **Módulos simples:** Antes de que los módulos fueran una característica estándar de JavaScript, las IIFEs se utilizaban para crear módulos simples.

### **Ejemplos Prácticos**

#### **1. Creando un ámbito local**

TypeScript

(function() { let mensaje = "Hola desde una IIFE"; console.log(mensaje); })(); // Intentar acceder a mensaje fuera de la IIFE resultará en un error

Use code [with caution.](file:///C:/faq#coding)

#### **2. Inicializando variables globales**

TypeScript

let contador = 0; (function() { contador = 10; })(); console.log(contador); // Imprimirá 10

Use code [with caution.](file:///C:/faq#coding)

#### **3. Simulando un módulo**

TypeScript

const miModulo = (function() { let variablePrivada = "Soy privada"; function funcionPrivada() { console.log("Función privada"); } return { variablePublica: "Soy pública", funcionPublica: function() { console.log("Función pública"); funcionPrivada(); } }; })(); miModulo.funcionPublica(); // Llama a la función públicaconsole.log(miModulo.variablePublica); // Acceder a la variable pública// No podemos acceder a variablePrivada ni a funcionPrivada desde fuera

Use code [with caution.](file:///C:/faq#coding)

#### **4. Evitando conflictos de nombres**

TypeScript

let nombre = "Juan"; (function(nombre) { console.log("Hola, " + nombre); // Imprimirá "Hola, Pedro" })("Pedro"); console.log(nombre); // Imprimirá "Juan" (el valor original)

Use code [with caution.](file:///C:/faq#coding)

### **IIFEs y TypeScript**

En TypeScript, las IIFEs se utilizan de la misma manera que en JavaScript. Sin embargo, TypeScript agrega el beneficio de la tipado estático, lo que permite escribir código más seguro y legible.

### **¿Cuándo usar IIFEs?**

* **Encapsulación de código:** Cuando deseas crear un ámbito local para evitar conflictos de nombres.
* **Inicialización inmediata:** Cuando necesitas ejecutar código al cargar la página.
* **Creación de módulos simples:** Aunque los módulos nativos de JavaScript son preferibles en la mayoría de los casos.
* **Evitar la contaminación del espacio de nombres global:** Cuando deseas mantener tus variables y funciones aisladas.

**En resumen**, las funciones anónimas autoinvocadas son una herramienta poderosa en TypeScript para organizar tu código y evitar problemas comunes. Al comprender cómo funcionan y cuándo usarlas, puedes escribir código más limpio y eficiente.