# Interfaces de TypeScript

Podemos hacer dos tipos de uso de las interfaces: implementadas por una clase y como un nuevo tipo de objeto.

Veamos las diferencias

## Uso de interfaz implementadas en clases

Se suele definir a las interfaces **como un contrato en el que se especifican las cosas que deben contener una clase para que pueda implementar una interfaz** o cumplir el contrato declarado por esa interfaz

Las interfaces nos van a permitir mejorar la consistencia de nuestro código mediante la validación de tipos. **Es decir, vamos a poder declarar objetos del tipo definido en la interfaz y en caso que dichos objetos no cumplan con lo definido en la interfaz el transpilador nos va advertir de que hay una inconsistencia**.

Las interfaces nos definen “*un esquema*” que incluyen ciertos campos y que las clases que implementen la interfaz deben cumplir

## Uso de la interfaz para definir un nuevo tipo de datos

Podemos definir variables que sean del tipo de una interface, lo cual nos va a obligar que ese objeto cumpla con el contrato definido en la interfaz (nos los va advertir el transpilador y el editor si es que trabaja con TypeScript)

## Crear una clase mediante Angular-CLI

Mediante este comando automáticamente nos va crear un nuevo archivo TypeScript que va a tener la declaración de la clase



**Notas:**

* Por defecto no crea el archivo .spec
* Podemos especificar el path donde queremos que cree el archivo escribiéndolo antes del nombre
* Se suele convenir comenzar el nombre de la interface con una ‘I’ o poner al final del nombre “*-interface”* *como* para indicar que se trata de una interfaz (es una buena práctica, no es obligatorio)

Podemos usar el signo de pregunta para definir que no se obligatorio

La declaración de la interfaz tiene una pinta así (es exactamente igual a la de la clase, pero en lugar de utilizar la palabra reservada *class* utiliza *interface*)

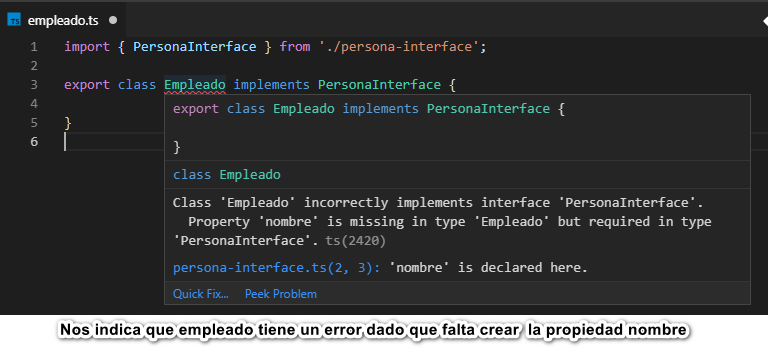


## Ejemplo de funcionamiento

Vamos a declarar una propiedad dentro de la interface



Luego vamos a crear una clase que implemente la interfaz que creamos, para hacerlo primero debemos importar la interface mediante *import* y luego utilizamos *implements* para implementarla



## Declaración de miembros

### Propiedades

Las propiedades se declaran igual que las propiedades dentro de una clase, solo que **no pueden tener valores asignados, es decir solo se definen los nombres y tipos de las propiedades**

Podemos declarar propiedades opcionales mediante ‘?’ luego del nombre del atributo



### Métodos

**En el caso de los métodos solo vamos a definir la firma**, es decir: nombre del método, tipo devuelto y cantidad y tipo de los parámetros recibidos

