**FORO TYPESCRIPT**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Promise**

Una promesa q puede devolver un número o un string

The generic type of the Promise should correspond to the non-error return-type of the function. The error is implicitly of type any and is not specified in the Promise generic type.

function test(arg: string): Promise<number> {

return new Promise<number>((resolve, reject) => {

if (arg === "a") {

resolve(1);

} else {

reject("1");

}

});

}

/////////////////////////////////////////////

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*

**Push**

var myArray: string[];

myArray = ["Bob", "Fred"];

myArray.push('Barney');

///////////////////////////////////// /

interface WordDataForQuestion {

question : {

id : number;

vocab : string;

comment : string;

};

possibleAnswers : {

id : number;

vocab : string;

comment : string;

}[];

}

nextWordData.possibleAnswers.push(

{

id : vocabs[index].id,

vocab : vocabs[index].voc\_t,

comment : vocabs[index].com\_t

}

);

//////////////////////////////////////////

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*

**Union types**

type A = 1 | 2

type B = 2 | 3

type I = A & B

type U = A | B

let a: A

let b: B

let i: I

let u: U

a = 1

a = 2

a = 3 // <- error

b = 1 // <- error

b = 2

b = 3

i = 1 // <- error

i = 2

i = 3 // <- error

u = 1

u = 2

u = 3

/////////////////////////////////////////// /

class **MyClass** {

name: string;

duration: number | undefined | 'auto';

}

var c = new **MyClass**();

// ...

if (c.duration === undefined) {

// ...

} else if (c.duration === 'auto') {

// ...

} else {

// In this branch, the type of c.duration will be number.

// ...

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*

**Unknown type**

Is much less permissive than any

The unknown type is only assignable to the any type and the unknown type itself