*Q1*:

Explain generics in TypeScript

Answer

Generics are able to create a component or function to work over a variety of types rather than a single one.

/\*\* A class definition with a generic parameter \*/

class Queue<T> {

private data = [];

push = (item: T) => this.data.push(item);

pop = (): T => this.data.shift();

}

const queue = new Queue<number>();

queue.push(0);

queue.push("1"); // ERROR : cannot push a string. Only numbers allowed

*Having Tech or Coding Interview?* Check 👉 [39 TypeScript Interview Questions](https://www.fullstack.cafe/interview-questions/typescript)

*Source:* [basarat.gitbooks.io](https://basarat.gitbooks.io/typescript/docs/types/generics.html)

*Q2*:

What is TypeScript and why would I use it in place of JavaScript?

Answer

**TypeScript** is a superset of JavaScript which primarily provides optional static typing, classes and interfaces. One of the big benefits is to enable IDEs to provide a richer environment for spotting common errors as *you type the code*. For a large JavaScript project, adopting TypeScript might result in more robust software, while still being deployable where a regular JavaScript application would run.

In details:

* TypeScript supports new ECMAScript standards and compiles them to (older) ECMAScript targets of your choosing. This means that you can use features of ES2015 and beyond, like modules, lambda functions, classes, the spread operator, destructuring, today.
* JavaScript code is valid TypeScript code; TypeScript is a superset of JavaScript.
* TypeScript adds type support to JavaScript. The type system of TypeScript is relatively rich and includes: interfaces, enums, hybrid types, generics, union and intersection types, access modifiers and much more. TypeScript makes typing a bit easier and a lot less explicit by the usage of type inference.
* The development experience with TypeScript is a great improvement over JavaScript. The IDE is informed in real-time by the TypeScript compiler on its rich type information.
* With strict null checks enabled (--strictNullChecks compiler flag) the TypeScript compiler will not allow undefined to be assigned to a variable unless you explicitly declare it to be of nullable type.
* To use TypeScript you need a build process to compile to JavaScript code. The TypeScript compiler can inline source map information in the generated .js files or create separate .map files. This makes it possible for you to set breakpoints and inspect variables during runtime directly on your TypeScript code.
* TypeScript is open source (Apache 2 licensed, see github) and backed by Microsoft. *Anders Hejlsberg*, the lead architect of C# is spearheading the project.

*Having Tech or Coding Interview?* Check 👉 [39 TypeScript Interview Questions](https://www.fullstack.cafe/interview-questions/typescript)

*Source:* [stackoverflow.com](https://stackoverflow.com/questions/12694530/what-is-typescript-and-why-would-i-use-it-in-place-of-javascript)

*Q3*:

Could we use TypeScript on backend and how?

Answer

Typescript doesn’t only work for browser or frontend code, you can also choose to write your backend applications. For example you could choose Node.js and have some additional type safety and the other abstraction that the language brings.

1. Install the default Typescript compiler

npm i -g typescript

1. The TypeScript compiler takes options in the shape of a tsconfig.json file that determines where to put built files and in general is pretty similar to a babel or webpack config.

{

"compilerOptions": {

"target": "es5",

"module": "commonjs",

"declaration": true,

"outDir": "build"

}

}

1. Compile ts files

tsc

1. Run

node build/index.js

*Having Tech or Coding Interview?* Check 👉 [39 TypeScript Interview Questions](https://www.fullstack.cafe/interview-questions/typescript)

*Source:* [jonathanmh.com](https://jonathanmh.com/typescript-node-js-tutorial-backend-beginner/)

*Q4*:

Does TypeScript support all object oriented principles?

**Junior**

[](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[TypeScript](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[39](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)

Answer

The answer is **YES**. There are 4 main principles to Object Oriented Programming:

* Encapsulation,
* Inheritance,
* Abstraction, and
* Polymorphism.

TypeScript can implement all four of them with its smaller and cleaner syntax.

*Having Tech or Coding Interview?* Check 👉 [39 TypeScript Interview Questions](https://www.fullstack.cafe/interview-questions/typescript)

*Source:* [jonathanmh.com](https://jonathanmh.com/typescript-node-js-tutorial-backend-beginner/)

*Q5*:

How could you check null and undefined in TypeScript?

**Junior**

[](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[TypeScript](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[39](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)

Answer

Just use:

if (value) {

}

It will evaluate to true if value is not:

* null
* undefined
* NaN
* empty string ''
* 0
* false

TypesScript includes JavaScript rules.

*Q6*:

How to implement class constants in TypeScript?

**Junior**

[](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[TypeScript](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[39](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)

Answer

In TypeScript, the const keyword cannot be used to declare class properties. Doing so causes the compiler to an error with "A class member cannot have the 'const' keyword." TypeScript 2.0 has the readonly modifier:

class MyClass {

readonly myReadonlyProperty = 1;

myMethod() {

console.log(this.myReadonlyProperty);

}

}

new MyClass().myReadonlyProperty = 5; // error, readonly

*Having Tech or Coding Interview?* Check 👉 [39 TypeScript Interview Questions](https://www.fullstack.cafe/interview-questions/typescript)

*Source:* [stackoverflow.com](https://stackoverflow.com/questions/37265275/how-to-implement-class-constants-in-typescript)

*Q7*:

What is a TypeScript Map file?

Answer

.map files are source map files that let tools map between the emitted JavaScript code and the TypeScript source files that created it. Many debuggers (e.g. Visual Studio or Chrome's dev tools) can consume these files so you can debug the TypeScript file instead of the JavaScript file.

*Having Tech or Coding Interview?* Check 👉 [39 TypeScript Interview Questions](https://www.fullstack.cafe/interview-questions/typescript)

*Source:* [stackoverflow.com](https://stackoverflow.com/questions/17493738/what-is-a-typescript-map-file)

*Q8*:

What is getters/setters in TypeScript?

Answer

TypeScript supports **getters/setters** as a way of intercepting accesses to a member of an object. This gives you a way of having finer-grained control over how a member is accessed on each object.

class foo {

private \_bar:boolean = false;

get bar():boolean {

return this.\_bar;

}

set bar(theBar:boolean) {

this.\_bar = theBar;

}

}

var myBar = myFoo.bar; // correct (get)

myFoo.bar = true; // correct (set)

*Having Tech or Coding Interview?* Check 👉 [39 TypeScript Interview Questions](https://www.fullstack.cafe/interview-questions/typescript)

*Source:* [typescriptlang.org](http://www.typescriptlang.org/docs/handbook/classes.html)

*Q9*:

Are strongly-typed functions as parameters possible in TypeScript?

**Mid**

[](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[TypeScript](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[39](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)

Problem

Consider the code:

class Foo {

save(callback: Function) : void {

//Do the save

var result : number = 42; //We get a number from the save operation

//Can I at compile-time ensure the callback accepts a single parameter of type number somehow?

callback(result);

}

}

var foo = new Foo();

var callback = (result: string) : void => {

alert(result);

}

foo.save(callback);

Can you make the result parameter in save a type-safe function? Rewrite the code to demonstrate.

Answer

In TypeScript you can declare your **callback type** like:

type NumberCallback = (n: number) => any;

class Foo {

// Equivalent

save(callback: NumberCallback): void {

console.log(1)

callback(42);

}

}

var numCallback: NumberCallback = (result: number) : void => {

console.log("numCallback: ", result.toString());

}

var foo = new Foo();

foo.save(numCallback)

*Having Tech or Coding Interview?* Check 👉 [39 TypeScript Interview Questions](https://www.fullstack.cafe/interview-questions/typescript)

*Source:* [stackoverflow.com](https://stackoverflow.com/questions/12694530/what-is-typescript-and-why-would-i-use-it-in-place-of-javascript)

*Q10*:

Does TypeScript supports function overloading?

**Mid**

[](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[TypeScript](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[39](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)

Answer

Yes, TypeScript does support function overloading but the implementation is a bit different if we compare it to OO languages. We are creating just one function and a number of declarations so that TypeScript doesn't give compile errors. When this code is compiled to JavaScript, the concrete function alone will be visible. As a JavaScript function can be called by passing multiple arguments, it just works.

class Foo {

myMethod(a: string);

myMethod(a: number);

myMethod(a: number, b: string);

myMethod(a: any, b?: string) {

alert(a.toString());

}

}

*Having Tech or Coding Interview?* Check 👉 [39 TypeScript Interview Questions](https://www.fullstack.cafe/interview-questions/typescript)

*Source:* [typescriptlang.org](https://www.typescriptlang.org/docs/handbook/declaration-files/do-s-and-don-ts.html)

*Q11*:

Explain how and why we could use property decorators in TS?

**Mid**

[](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[TypeScript](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[39](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)

Answer

Decorators can be used to modify the behavior of a class or become even more powerful when integrated into a framework. For instance, if your framework has methods with restricted access requirements (just for admin), it would be easy to write an @admin method decorator to deny access to non-administrative users, or an @owner decorator to only allow the owner of an object the ability to modify it.

class CRUD {

get() { }

post() { }

@admin

delete() { }

@owner

put() { }

}

*Having Tech or Coding Interview?* Check 👉 [39 TypeScript Interview Questions](https://www.fullstack.cafe/interview-questions/typescript)

*Source:* [www.sitepen.com](https://www.sitepen.com/blog/2015/10/20/typescript-decorators/)

*Q12*:

How can you allow classes defined in a module to accessible outside of the module?

**Mid**

[](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[TypeScript](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[39](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)

Answer

Classes define in a module are available within the module. Outside the module you can’t access them.

module Vehicle {

class Car {

constructor (

public make: string,

public model: string) { }

}

var audiCar = new Car("Audi", "Q7");

}

// This won't work

var fordCar = Vehicle.Car("Ford", "Figo");

As per above code, fordCar variable will give us compile time error. To make classes accessible outside module use export keyword for classes.

module Vehicle {

export class Car {

constructor (

public make: string,

public model: string) { }

}

var audiCar = new Car("Audi", "Q7");

}

// This works now

var fordCar = Vehicle.Car("Ford", "Figo");

*Having Tech or Coding Interview?* Check 👉 [39 TypeScript Interview Questions](https://www.fullstack.cafe/interview-questions/typescript)

*Source:* [http://www.talkingdotnet.com](http://www.talkingdotnet.com/typescript-interview-questions/)

*Q13*:

Is that TypeScript code valid? Explain why.

**Mid**

[](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[TypeScript](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[39](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)

Problem

Consider:

class Point {

x: number;

y: number;

}

interface Point3d extends Point {

z: number;

}

let point3d: Point3d = {x: 1, y: 2, z: 3};

Answer

Yes, the code is valid. A class declaration creates two things: a *type* representing instances of the class and a *constructor function*. Because classes create types, you can use them in the same places you would be able to use interfaces.

*Having Tech or Coding Interview?* Check 👉 [39 TypeScript Interview Questions](https://www.fullstack.cafe/interview-questions/typescript)

*Source:* [typescriptlang.org](http://www.typescriptlang.org/docs/handbook/classes.html)

*Q14*:

What are different components of TypeScript?

**Mid**

[](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[TypeScript](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)**[39](https://www.fullstack.cafe/interview-questions/typescript" \o "TypeScript Interview Questions)

Answer

There are mainly 3 components of TypeScript .

1. **Language** – The most important part for developers is the new language. The language consist of new syntax, keywords and allows you to write TypeScript.
2. **Compiler** – The TypeScript compiler is open source, cross-platform and open specification, and is written in TypeScript. Compiler will compile your TypeScript into JavaScript. And it will also emit error, if any. It can also help in concating different files to single output file and in generating source maps.
3. **Language Service** – TypeScript language service which powers the interactive TypeScript experience in Visual Studio, [VS Code](http://www.talkingdotnet.com/what-is-visual-studio-code-and-difference-between-visual-studio-2015/), Sublime, the TypeScript playground and other editor.