
500
TypeScript
Interview
Questions
Vamsee Puligadda

cause.

**For any suggestions/concerns, please write to us:**
indianwolfpublications@gmail.com

**Q1. What is TypeScript and why do we need it?**

**Ans.** JavaScript is the only client side language universally supported by all browsers. But JavaScript is not the best designed language. It's not a class-based object-oriented language, doesn't support class based inheritance, unreliable dynamic typing and lacks in compile time error checking. And TypeScript addresses all these problems. In other words, TypeScript is an attempt to "fix" JavaScript problems.

TypeScript is a free and open source programming language developed and maintained by Microsoft. It is a strict superset of JavaScript, and adds **optional static typing** and **class-based object-oriented programming** to the language. TypeScript is quite easy to learn and use for developers familiar with C#, Java and all strong typed languages. At the end of day "TypeScript is a language that generates plain JavaScript files."

As stated on Typescript official website, "TypeScript lets you write JavaScript the way you really want to. TypeScript is a typed superset of JavaScript that compiles to plain JavaScript. Any browser. Any host. Any OS. Open Source." Where "**typed**" means that it considers the types of variables, parameters and functions.

**Q2. What are the benefits of TypeScript?**

**Ans.** TypeScript has following benefits.

- It helps in code structuring.
- Use class based object oriented programming.
- Impose coding guidelines.
- Offers type checking.
- Compile time error checking.
- Intellisense.

**Q3. What are different components of TypeScript?**

**Ans.** 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, Sublime, the TypeScript playground and other editor.

**Q4. How can you get TypeScript and install it?**

**Ans.** TypeScript can be installed and managed via npm, the Node.js package manager. To install TypeScript, first ensure the npm is installed properly. And then run following command to install TypeScript globally on your system.

```
1  npm install -g typescript
```

TypeScript is included in Visual Studio 2013 Update 2 and Visual Studio 2015 by default. TypeScript also provides support for other editors like Visual Studio Code, sublime, Emacs and Vim.

**Q5. How do you compile TypeScript files?**

**Ans.** The extension for any TypeScript file is **".ts"**. And any JavaScript file is TypeScript file as it is a superset of JavaScript. So change extension of ".js" to ".ts" file and your TypeScript file is ready. To compile any .ts file into .js use following command.

```
1  tsc <TypeScript File Name>
```

For example, to compile "Helloworld.ts"

```
1  tsc helloworld.ts
```

And the result would be helloworld.js

**Q6. Is it possible to combine multiple .ts files into a single .js file?**

**Ans.** Yes, it possible. While compiling add --outFILE [OutputJSFileName] option.

```
1  tsc --outFile comman.js file1.ts file2.ts file3.ts
```

This will compile all 3 ".ts" file and output into single "comman.js" file. And what will happen if you don't provide a output file name.

```
1  tsc --outFile file1.ts file2.ts file3.ts
```

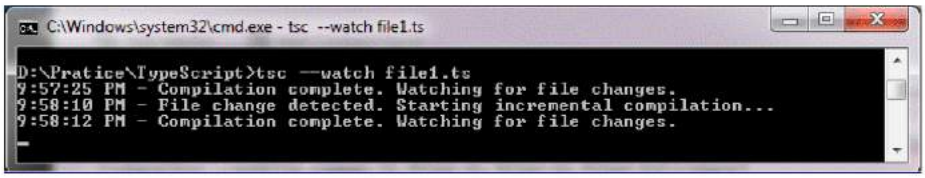
In this case, file2.ts and file3.ts will be compiled and the output will be placed in file1.ts. So now your file1.ts contains JavaScript code.

**Q7. Is it possible to compile .ts automatically with real time changes in .ts file?**

**Ans.** Yes. Using --watch compiler option this can be achieved.

```
1  tsc --watch file1.ts
```

This will first compile file1.ts in file.js and watch for the file changes. As soon as there is any change detected, it will compile it again. But you need to ensure that command prompt must not be closed, used with --watch option.




### Q8. Does TypeScript support all object oriented principles?

**Ans.** 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. Read Write Object-Oriented JavaScript with TypeScript.

### Q9. Which object oriented terms are supported by TypeScript?

**Ans.** TypeScript supports following object oriented terms.

- Modules
- Classes
- Interfaces
- Data Types
- Member functions

### Q10. Which are the different data types supported by TypeScript?

**Ans.** TypeScript supports following data types.

- Boolean var bValue: boolean = false;
- Number var age: number = 16;
- String var name: string = "jon";
- Array var list:number[] = [1, 2, 3];
- Enum

```
1  enum Color {Red, Green, Blue};
2  var c: Color = Color.Green;
```

- Any var unknownType: any = 4;
- Void

```
1  function NoReturnType(): void {
2  }
```

### Q11. How TypeScript is optionally statically typed language?

**Ans.** TypeScript is referred as optionally statically typed, which means you can ask the compiler to ignore the type of a variable. Using any data type, we can assign any type of value to the variable. TypeScript will not give any error checking during compilation.

```
1  var unknownType: any = 4;
2  unknownType = "Okay, I am a string";
```

```
3 unknownType = false; // A boolean.
```

**Q12. What are modules in TypeScript?**

**Ans.** Modules are the way to organize code in TypeScript. Modules don't have any features, but can contain classes and interfaces. It is same like namespace in C#.

**Q13. What are classes in TypeScript?**

**Ans.** The concept of classes is very similar to .Net/Java. A Class can have constructor, member variables, properties and methods. TypeScript also allows access modifiers "private" and "public" for member variables and functions.

**Q14. How do you implement inheritance in TypeScript?**

**Ans.** Using extends keyword, we can implement inheritance.

```
class Animal {
  public domestic:boolean;
  constructor(public name: string) { }
}

class Cat extends Animal {
  constructor(name: string, domestic: boolean)
  {
   super(name);
   this.domestic = true;
  }
}

class Tiger extends Animal {
  constructor(name: string, domestic: boolean)
  {
   super(name);
   this.domestic = false;
  }
}
```

**Q15. How to call base class constructor from child class in TypeScript?**

**Ans.** Using super(), we can call base class constructor, as seen in above code.

**Q16. What is the default access modifier for members of a class in TypeScript?**

**Ans.** In TypeScript, each member of class is **public** by default.

**Q17. How can you allow classes defined in a module to accessible outside of the module?**

**Ans.** Classed 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");
```

**Q18. How does TypeScript support optional parameters in function as in JavaScript every parameter is optional for a function?**

**Ans.** In JavaScript, every parameter is considered optional. If no value is supplied, then it is treated as undefined. So while writing functions in TypeScript, we can make a parameter optional using the **"?"**after the parameter name.

```
function Demo(arg1: number, arg2? :number) {
}
```

So, arg1 is always required and arg2 is optional parameter. Remember, **Optional parameters must follow required parameters.** If we want to make arg1 optional, instead of arg2 then we need to change the

order and arg1 must be put after arg2.

```
function Demo(arg2: number, arg1? :number) {
}
```

Similar to optional parameters, default parameters are also supported.

```
function Demo(arg1: number, arg2 = 4) {
}
```

**Q19. Does TypeScript supports function overloading as JavaScript doesn't support function overloading?**

**Ans.** Yes, TypeScript does support function overloading. But the implementation is odd. When you overload in TypeScript you only have one implementation with multiple signatures.

```
class Customer {
  name: string;
  Id: number;

  add(Id: number);
  add(name:string);
  add(value: any) {
  if (value && typeof value == "number") {
    //Do something
  }
  if (value && typeof value == "string") {
    //Do Something
  }
 }
}
```

The first signature has one parameter of type number whereas the second signature has a parameter of type string. The third function contains the actual implementation and has a parameter of type any. The any data type can take any type of data. The implementation then checks for the type of the supplied parameter and execute different piece of code based on supplier parameter type.

**OR** You can also use union type introduced in TypeScript 1.4. Union types let you represent a value which is one of multiple types.

```
add(a:string|number) {
 //do something
}
```

Using union type, you can typically remove the need for an overload.

**Q20. Is it possible to debug any TypeScript file?**

**Ans.** Yes, it is possible. To debug it, you need .js source map file. If you are new to source map, read more here. So compile the .ts file with the --sourcemap flag to generate a source map file.

```
1 tsc -sourcemap file1.ts
```

This will create file1.js and file1.js.map. And last line of file1.js would be reference of source map file.

```
1 //# sourceMappingURL=file1.js.map
```

**Q21. What is TypeScript Definition Manager and why do you need it?**

**Ans.** TypeScript Definition Manager (TSD) is a package manager to search and install TypeScript definition files directly from the community driven DefinitelyTyped repository. Let's see with an example.

Suppose, you want to use some jQuery code in your .ts file.

```
1 $(document).ready(function() { //Your jQuery code});
```

And now when you try to compile it using tsc, you will get compile time error **Cannot find name "$".**That's because TypeScript can't understand what does "$" means. So somehow we need to inform TypeScript compiler that it belongs to jQuery. That's where TSD comes into play. You can download jQuery Type Definition file and include it in your .ts file. First, install TSD.

```
1 npm install tsd -g
```

In your typescript directory, create a new typescript project by running:

```
1 tsd init
```

Then install the definition file for jquery.

```
1 tsd query jquery --action install
```

This will download and create a new directory containing your jquery definition file. The definition file ends with ".d.ts". So now include it by updating your typescript file to point to the jquery definition.

```
1 /// <reference path="typings/jquery/jquery.d.ts" />
2 $(document).ready(function() { //To Do
3 });
```

Now try to compile again and this time js will be generated without any error.

So TSD will help you to get type definition file for

required framework. If you wish to use angular, then download angular type definition file. You can find the complete list here.

**Q22. What is TypeScript Declare Keyword?**

**Ans.** It's quite possible that JavaScript libraries/frameworks don't have TypeScript definition files and yet you want to use them without any errors. Te solution is to use the declare keyword.
The declarekeyword is used for **ambient declarations** where you want to define a variable that may not have originated from a TypeScript file.

```
1  declare var unKnownLibrary;
```

TypeScript runtime will assign unKnownLibrary variable any type. You won't get Intellisense in design time but you will be able to use the library in your code.

**Q23. How to generate TypeScript definition file from any .ts file?**

**Ans.** You can generate TypeScript definition file from any .ts file via tsc compiler. Generating a TypeScript definition will make your TypeScript file reusable.

```
1  tsc --declaration file1.ts
```

**Q24. What is tsconfig.json file?**

**Ans.** The presence of a tsconfig.json file in a directory indicates that the directory is the root of a TypeScript project. The tsconfig.json file specifies the root files and the compiler options required to compile the project. And using this file we can streamline building TypeScript project. Below is a sample tsconfig.json file.

```
{
 "compilerOptions": {
  "removeComments": true,
  "sourceMap": true
 },
 "files": [
  "main.ts",
  "othermodule.ts"
 ]
}
```

Within files section, define all the .ts files in the project. And When invoke tsc without any other arguments with the above file in the current directory, it will compile all the files with the given compiler option settings.

**Q25. What are disadvantages of TypeScript?**

**Ans.** Well, TypeScript is great but there are some

disadvantage as well.

- TypeScript is just another way to write JavaScript. It doesn't fix the problems of JavaScript. It just creates an illusion that it does.
- One more tool to learn.
- TypeScript has dependency on type definition files, if you wish to use any existing JavaScript libraries.
- Quality of type definition files is a concern as how can you be sure the definitions are correct?
- Frequent releases of new versions JavaScript library is also a pain area. Because if their type definition files are not updated then you can't use them instantly.
- In order to run the application in the browser, a compile step is required to transform TypeScript into JavaScript.
- Web developers are using JavaScript from decades and TypeScript doesn't bring anything new.
- To use any third party library, definition file is you need. And not all the third party library have definition file available.

These are also called the primitive types in TypeScript:

- **Number** type: it is used to represent number type values and represents double precision floating point values.
  - `var variable_name: number;`
- **String** type: it represents a sequence of characters stored as Unicode UTF-16 code. It is the same as JavaScript primitive type.
  - `var variable_name: string;`
- **Boolean** type: in Typescript, it is used to represent a logical value. When we use the Boolean type, we get output only in true or false. It is also the same as JavaScript primitive type.
  - `var variable_name: bool;`
- **Null** type: it represents a null literal and it is not possible to directly reference the null type value itself.
  - `var variable_name:number = null;`
- **Undefined** type: it is the type of undefined literal. This type of built-in type is the sub-type of all the types.
  - `var variable_name:number = undefined;`

### *Q32:* What are Modules in Typescript?

Modules in Typescript helps in organizing the code. There are 2 types of Modules — Internal and External

- **Internal Modules** are now replaceable by using Typescript's namespace.
- **External Modules** used to specify and load

dependencies between multiple external js files. If there is only one js file used, then external modules are not relevant.

### *Q33:* What is Typescript and why one should use it?

TypeScript is a free and open-source programming language developed and maintained by Microsoft. It is a strict syntactical superset of JavaScript, and adds optional static typing and class-based object-oriented programming to the language.

### *Q34:* Explain generics in TypeScript

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
```

### *Q35:* What is TypeScript and why would I use it in pla

**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.

### *Q36:* What is TypeScript and why do we need it?

JavaScript is the only client side language universally supported by all browsers. But JavaScript is not the best designed language. It's not a class-based object-oriented language, doesn't support class based inheritance, unreliable dynamic typing and lacks in compile time error checking. And TypeScript addresses all these problems. In other words, TypeScript is an attempt to "fix" JavaScript problems.

TypeScript is a free and open source programming language developed and maintained by Microsoft. It is a strict superset of JavaScript, and adds **optional static typing** and **class-based object-oriented programming** to the language. TypeScript is quite easy to learn and use for developers familiar with C#, Java and all strong typed languages. At the end of day "TypeScript is a language that generates plain JavaScript files."

As stated on Typescript official website, "TypeScript lets you write JavaScript the way you really want to. TypeScript is a typed superset of JavaScript that compiles to plain JavaScript. Any browser. Any host. Any OS. Open Source." Where "**typed**" means that it considers the types of variables, parameters and functions.

### *Q37:* What are the benefits of TypeScript?

TypeScript has following benefits.

- It helps in code structuring.
- Use class based object oriented programming.
- Impose coding guidelines.
- Offers type checking.
- Compile time error checking.
- Intellisense.

### *Q38:* Do we need to compile TypeScript files and why?

Yes we do. Typescript is just a language Extension browsers can't interpret it. Converting from TypeScript to JavaScript is called compiling. Compiling doesn't mean binary code is created in this case. For this kind of translation, also the

term transpilation is used instead of compilation.

### *Q39:* How to call base class constructor from child cla

We can call base class constructor using `super()`.

### *Q40:* What are the difference beetween Typescript an

- It is an object oriented programming language (not pure).
- Here it is static typing (We can declare a variable in multiple ways). ex: var num : number.
- It has interfaces.
- It has optional parameter feature.
- It has Rest Parameter feature.
- Supports generics.
- Supports Modules
- Number, string etc. are the interfaces.

### *Q41:* What is Interface in TypeScript?

One of TypeScript's core principles is that type-checking focuses on the *shape* that values have.

An `interface` is a virtual structure that only exists within the context of TypeScript. The TypeScript compiler uses interfaces solely for type-checking purposes.

When you define your interface you're saying that any object (not an instance of a class) given this contract must be an object containing interfaces properties.

### *Q42:* When to use interfaces and when to use classes ir

If you need/wish to create an instance of perhaps a custom object, whilst getting the benefits of type-checking things such as arguments, return types or generics - a class makes sense.

If you're not creating instances - we have interfaces at our disposal, and their benefit comes from not generating any source code, yet allowing us to somewhat "virtually" type-check our code.

## Q41: What is TypeScript and why would I us of JavaScript?

☆

**Answer:**

**TypeScript** is a superset of JavaScript which primar optional static typing, classes and interfaces. One o benefits is to enable IDEs to provide a richer enviror

spotting common errors as *you type the code*. For a la
project, adopting TypeScript might result in more r
software, while still being deployable where a regul
application would run.

In details:

- TypeScript supports new ECMAScript standards ar
them to (older) ECMAScript targets of your choosir
that you can use features of ES2015 and beyond, li
lambda functions, classes, the spread operator, des
today.
- JavaScript code is valid TypeScript code; TypeScrip
JavaScript.
- TypeScript adds type support to JavaScript. The ty
TypeScript is relatively rich and includes: interface
hybrid types, generics, union and intersection type
modifiers and much more. TypeScript makes typir
and a lot less explicit by the usage of type inferenc
- The development experience with TypeScript is a g
improvement over JavaScript. The IDE is informed
the TypeScript compiler on its rich type informatic
- With strict null checks enabled ( `--strictNullChecks`
the TypeScript compiler will not allow undefined t
a variable unless you explicitly declare it to be of n
- To use TypeScript you need a build process to com
JavaScript code. The TypeScript compiler can inlin
information in the generated .js files or create sepa
This makes it possible for you to set breakpoints ar
variables during runtime directly on your TypeScr
- TypeScript is open source (Apache 2 licensed, see g
backed by Microsoft. *Anders Hejlsberg*, the lead arch
spearheading the project.

## Q42: Explain generics in TypeScript

☆

Generics are able to create a component or function
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 numb
```

## Q43: Does TypeScript support all object orier principles?

☆☆

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

- Encapsulation,
- Inheritance,
- Abstraction, and
- Polymorphism.

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

## Q44: How could you check null and undefine TypeScript?

☆☆

Just use:

```
if (value) {
}
```

It will evaluate to `true` if `value` is not:

- null
- undefined
- NaN
- empty string `''`
- 0
- false

TypesScript includes JavaScript rules.

## Q45: How to implement class constants in TypeSc

☆☆

In TypeScript, the `const` keyword cannot be used to properties. Doing so causes the compiler to an error member cannot have the 'const' keyword." TypeScri the `readonly` modifier:

```
class MyClass {
  readonly myReadonlyProperty = 1;

  myMethod() {
    console.log(this.myReadonlyProperty);
  }
}

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

## Q46: What is a TypeScript Map file?

☆☆

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

## Q47: What is getters/setters in TypeScript?

☆☆

TypeScript supports **getters/setters** as a way of int
accesses to a member of an object. This gives you a v
finer-grained control over how a member is accesse
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)
```

## Q48: Could we use TypeScript on backend an

☆☆

Typescript doesn't only work for browser or fronter
can also choose to write your backend applications.
you could choose Node.js and have some additional
and the other abstraction that the language brings.

1. Install the default Typescript compiler

```
npm i -g typescript
```

2. The TypeScript compiler takes options in the shap
   tsconfig.json file that determines where to put bui
   general is pretty similar to a babel or webpack con

```
{
 "compilerOptions": {
  "target": "es5",
  "module": "commonjs",
  "declaration": true,
```

```
utDir": "build"
```

. Compile ts files

```
```

. Run

```
e build/index.js
```

## 9: What are different components of TypeScri

☆

re are mainly 3 components of TypeScript .

. **Language** – The most important part for developers is t language. The language consist of new syntax, keywor allows you to write TypeScript.

. **Compiler** – The TypeScript compiler is open source, cro platform and open specification, and is written in Type Compiler will compile your TypeScript into JavaScript. also emit error, if any. It can also help in concating diff to single output file and in generating source maps.

. **Language Service** – TypeScript language service which interactive TypeScript experience in Visual Studio, VS ( Sublime, the TypeScript playground and other editor.

## 0: Is that TypeScript code valid? Explain why.

☆

sider:

```
s Point {
number;
number;

rface Point3d extends Point {
number;

oint3d: Point3d = {x: 1, y: 2, z: 3};
```

**swer:**

, the code is valid. A class declaration creates two thi *pe* representing instances of the class and a *construct ction*. Because classes create types, you can use them ne places you would be able to use interfaces.