TYPESCRIPT INTERVIEW QUESTIONS & ANSWERS

SWIPE UP

Q1: What is TypeScript and why one should use it?

Answer

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.

Q2: What are Modules in Typescript?

Answer

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.

TS

Q3: List the built-in types in Typescript

Answer

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;
```

TS

SWIPE UP

Q3: List the built-in types in Typescript

Answer

These are also called the primitive types in TypeScript:

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: string;
```

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;
```

TS

SWIPE UP

Q3: List the built-in types in Typescript

Answer

These are also called the primitive types in TypeScript:

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;
```

SWIPE UP

TS

Q4: 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
```

TS

SWIPE UP

Q5: 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.

SWIPE UP

TS

Q5: What is TypeScript and why would I use it in place of JavaScript?

Answer

- 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.

TS

SWIPE UP

Q5: What is TypeScript and why would I use it in place of JavaScript?

Answer

- 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.

TS

SWIPE UP

Q6: Do we need to compile TypeScript files and why?

Answer

Yes we do. Typescript is just a language Extension browsers can't interpret it. Converting from TypeScript to JavaScript is called compilin

Q7: What are the benefits of TypeScript?

Answer

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.

TS

SWIPE UP

Q8: What is TypeScript and why do we need it?

Answer

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.

TS

SWIPE UP

Q9: How to call base class constructor from child class in TypeScript?

Answer

We can call base class constructor using super().

Q10: What is Interface in TypeScript?

Answer

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.

TS

SWIPE UP

Q11: What is the difference between Classes and Interfaces in Typescript?

Answer

We use classes as object factories. A class defines a blueprint of what an object should look like and act like and then implements that blueprint by initialising class properties and defining methods. Classes are present throughout all the phases of our code.

Unlike classes, an interface is a virtual structure that only exists within the context of TypeScript. The TypeScript compiler uses interfaces solely for type-checking purposes. Once code is transpiled to its target language, it will be stripped from interfaces.

A class may define a factory or a singleton by providing initialisation to its properties and implementation to its methods, an interface is simply a structural contract that defines what the properties of an object should have as a name and as a type.

TS

SWIPE UP

Q12: What are the difference beetween Typescript and JavaScript?

Answer

- 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.

TS

SWIPE UP

Q13: What is Decorators in TypeScript?

Answer

A Decorator is a special kind of declaration that can be attached to a class declaration, method, accessor, property, or parameter. Decorators are functions that take their target as the argument. With decorators we can run arbitrary code around the target execution or even entirely replace the target with a new definition.

There are 4 things we can decorate in ECMAScript2016 (and Typescript): constructors, methods, properties and parameters.

Q14: 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.

SWIPE UP

TS

Q15: 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)
```

TS

SWIPE UP

Q16: Which object oriented terms are supported by TypeScript?

Answer

TypeScript supports following object oriented terms:

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

Q17: When to use interfaces and when to use classes in TypeScript?

Answer

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.

SWIPE UP

TS

Q18: Does TypeScript support all object oriented principles?

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.

TS

SWIPE UP

Q19: How could you check null and undefined in TypeScript?

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.

TS

SWIPE UP

Q20: How to implement class constants in TypeScript?

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
```

TS

SWIPE UP

Q21: 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
```

2. 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"
  }
}
```

TS

SWIPE UP