

declare

--	--	--

--	--

- □□
- declare variable
- declare function
- declare class
- declare module □ declare namespace
- declare global
- declare enum
- declare module □□□□□□□□
- □□□□



`declare`



declare

```
declare [ ]
```

- `const` `let` `var`
- `type` `interface`
- `class`
- `enum`
- `function`
- `module`
- `namespace`

declare

Diagram illustrating a memory layout with 256 bits. The top row represents a single 256-bit block. The bottom row is split into two 128-bit blocks. The right 128-bit block is labeled "declare".

declare

--	--	--	--	--	--	--	--

declare

declare

declare variable

declare

x

```
x = 123; // 
```

x

declare

```
declare let x:number;  
x = 1;
```

declare any

```
declare let x;  
x = 1;
```

x any

document

```
declare var document;  
document.title = 'Hello';
```

declare document TypeScript
lib.d.ts

TypeScript document any

declare

```
//  
declare let x:number = 1;
```

declare

declare function

declare `function sayHello()`

`sayHello()`

```
declare function sayHello(  
  name:string  
):void;  
  
sayHello('foo');
```

`function` declare `function` `sayHello()`

`function` declare `function` TypeScript
declare `function`

```
// foo  
function sayHello(  
  name:string  
):void;  
  
let foo = 'bar';  
  
function sayHello(name:string) {  
  return 'foo' + name;  
}
```

`function`

declare class

declare `class` `Animal`

```
declare class Animal {  
  constructor(name:string);  
  eat():void;  
  sleep():void;
```

```
}
```

□□□□□□□□□□

```
declare class C {  
    // 属性  
    public static s0():string;  
    private static s1:string;  
  
    // 方法  
    public a:number;  
    private b:number;  
  
    // 构造函数  
    constructor(arg:number);  
  
    // 方法  
    m(x:number, y:number):number;  
  
    // 属性  
    get c():number;  
    set c(value:number);  
  
    // 属性  
    [index:string]:any;  
}
```

□□□□ declare □□□□□ Class □□□□□□□□

declare module□ declare namespace

□□□□□□□□□□□□□□

declare □ module □ namespace □□□□

```
declare namespace AnimalLib {  
    class Animal {  
        constructor(name:string);  
    }  
}
```

```

    eat():void;
    sleep():void;
}

type Animals = 'Fish' | 'Dog';
}

// ❷
declare module AnimalLib {
    class Animal {
        constructor(name:string);
        eat(): void;
        sleep(): void;
    }

    type Animals = 'Fish' | 'Dog';
}

```

❶❶❶❶❶ declare ❶❶❶❶❶ module ❶ namespace ❶❶❶❶❶

declare module ❶ declare namespace ❶❶❶❶❶ export ❶❶❶❶❶

```

declare namespace Foo {
    export var a: boolean;
}

declare module 'io' {
    export function readFile(filename:string):string;
}

```

❶❶❶❶❶ namespace ❶ module ❶❶❶❶❶ export ❶❶❶❶

❶❶❶❶❶❶❶❶❶❶❶❶ myLib ❶❶❶❶❶❶❶❶❶❶ makeGreeting() ❶❶❶ numberOfGreetings ❶

```

let result = myLib.makeGreeting('❶❶');
console.log('❶❶❶❶' + result);

let count = myLib.numberOfGreetings;

```

myLib ❶❶❶❶❶❶❶❶❶❶❶❶

```
declare namespace myLib {
  function makeGreeting(s:string): string;
  let numberOfGreetings: number;
}
```

declare

```
import { Foo as Bar } from 'moduleA';

declare module 'moduleA' {
  interface Foo {
    custom: {
      prop1: string;
    }
  }
}
```

moduleA Foo interface Bar

declare Foo custom Foo Bar

Foo interface

```
// a.ts
export interface A {
  x: number;
}

// b.ts
import { A } from './a';

declare module './a' {
  interface A {
    y: number;
  }
}

const a:A = { x: 0, y: 0 };
```

a.ts A b.ts y declare module './a' {}

a.ts interface

```

1 declare module NAME
    import
    export
    import { A } from './a'

```

3 default export

```
// ❷
declare module "foo";

// ❸
declare module "hot-new-module";
```

```
declare module [ ]
```

declare global

JavaScript `declare global {}`

```
export {};
```

```
declare global {
  interface String {
    toSmallString(): string;
  }
}
```

```
String.prototype.toSmallString = ():string => {
  // 测试
  return '';
};
```

测试
JavaScript
String
toSmallString()
declare global

测试
export {}
declare

测试
window
Window
myAppConfig

```
export {};
```

```
declare global {
  interface Window {
    myAppConfig:object;
  }
}
```

```
const config = window.myAppConfig;
```

declare global 测试

declare enum

declare 测试 enum 测试

```
declare enum E1 {
  A,
```

```

    B,
}

declare enum E2 {
    A = 0,
    B = 1,
}

declare const enum E3 {
    A,
    B,
}

declare const enum E4 {
    A = 0,
    B = 1,
}

```

declare module















```

declare module "url" {
    export interface Url {
        protocol?: string;
        hostname?: string;
        pathname?: string;
    }

    export function parse(
        urlStr: string,
        parseQueryString?,
        slashesDenoteHost?
    ): Url;
}

```

```
declare module "path" {  
  export function normalize(p: string): string;  
  export function join(...paths: any[]): string;  
  export var sep: string;  
}
```

|||||

url |

path |||||

node.d.ts

|||||

|||||

declare module

|||||

```
declare module "lodash" {  
  export * from "../../dependencies/lodash";  
  export default from "../../dependencies/lodash";  
}
```

|||||

declare module "lodash" |||

lodash

|||||

|||||

```
/// <reference path="node.d.ts"/>
```

|||||

declare

|||||

|||||

- [How Does The Declare Keyword Work In TypeScript?](#), Tim Mouskhelichvili

||| #1

| ||| ||| 2023-07-09 12:07:22 UTC

| ||| ||| 2025-09-02 08:24:04 UTC