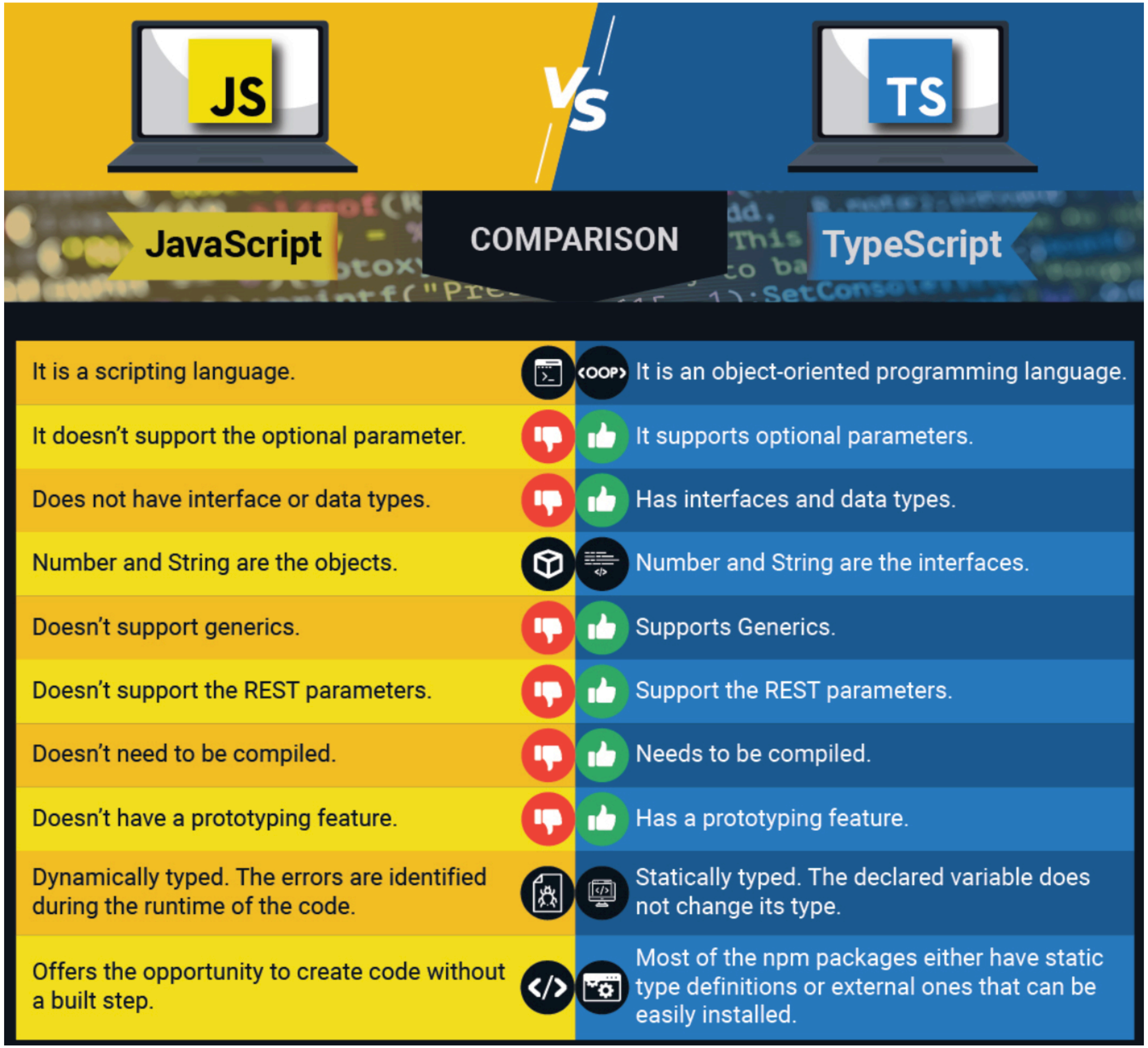
Typescript

CHEATSHEET FOR

TypeScript

Typescript for Javascript programmers

https://devhints.io/typescript



JavaScript	COMPARISON	TypeScript
It is a scripting language.		It is an object-oriented programming language.
It doesn't support the optional parameter.	👎 👍	It supports optional parameters.
Does not have interface or data types.	👎 👍	Has interfaces and data types.
Number and String are the objects.		Number and String are the interfaces.
Doesn't support generics.	👎 👍	Supports Generics.
Doesn't support the REST parameters.	👎 👍	Support the REST parameters.
Doesn't need to be compiled.	👎 👍	Needs to be compiled.
Doesn't have a prototyping feature.	👎 👍	Has a prototyping feature.
Dynamically typed. The errors are identified during the runtime of the code.		Statically typed. The declared variable does not change its type.
Offers the opportunity to create code without a built step.		Most of the npm packages either have static type definitions or external ones that can be easily installed.

https://odaninfotech.com/why-has-typescript-become-more-popular-than-javascript/

Basic Types

```
any
void

boolean
number
string

null
undefined

bigint
symbol

string[]           /* or Array<string> */
[string, number]  /* tuple */

string | null | undefined   /* union */

never  /* unreachable */
unknown
```

```
enum Color {
  Red,
  Green,
  Blue = 4
};

let c: Color = Color.Green
```

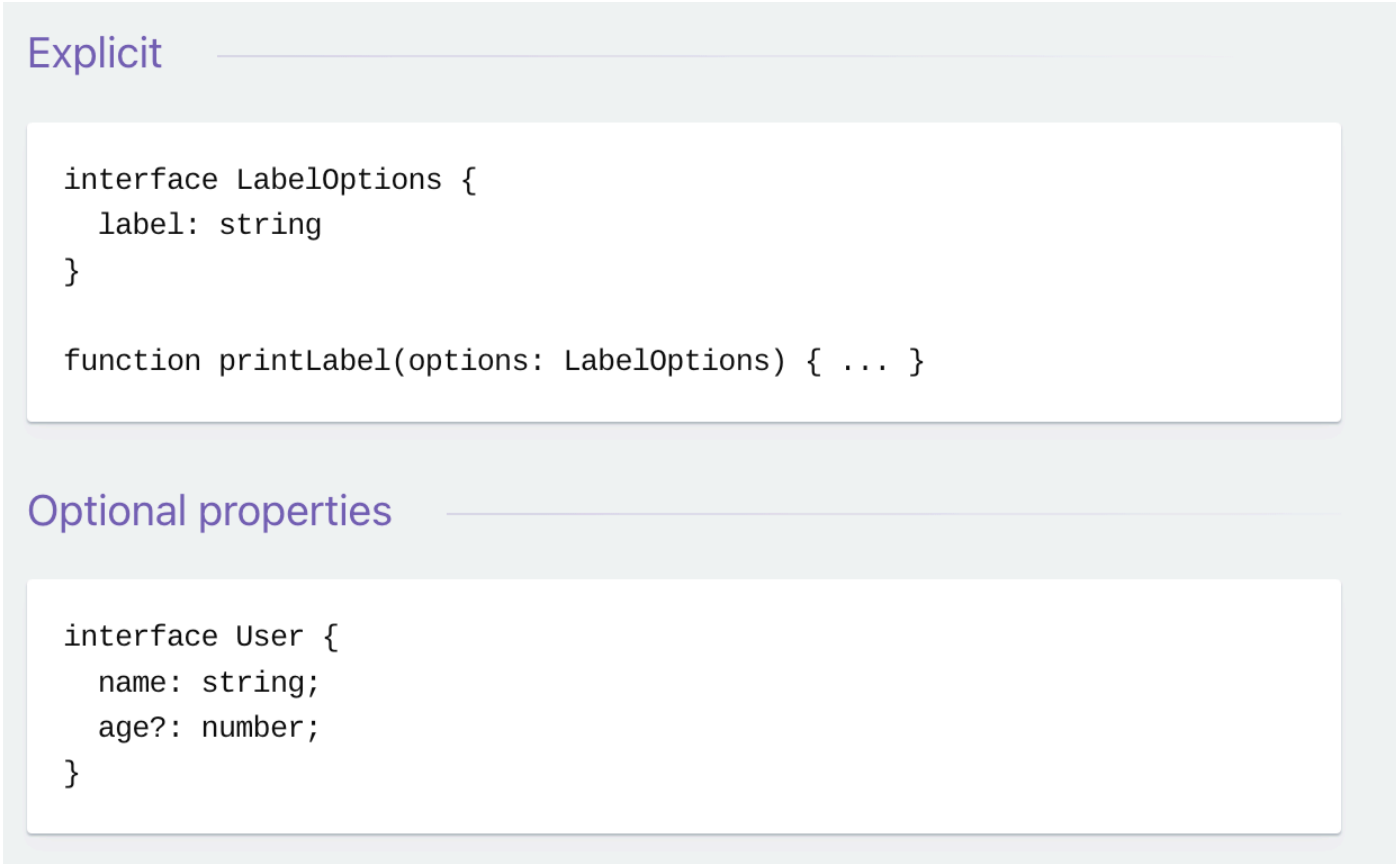
Declarations

```
let isDone: boolean
let isDone: boolean = false
```

```
function add (a: number, b: number): number {
  return a + b
}

// Return type is optional
function add (a: number, b: number) { ... }
```

Interfaces 1



Explicit

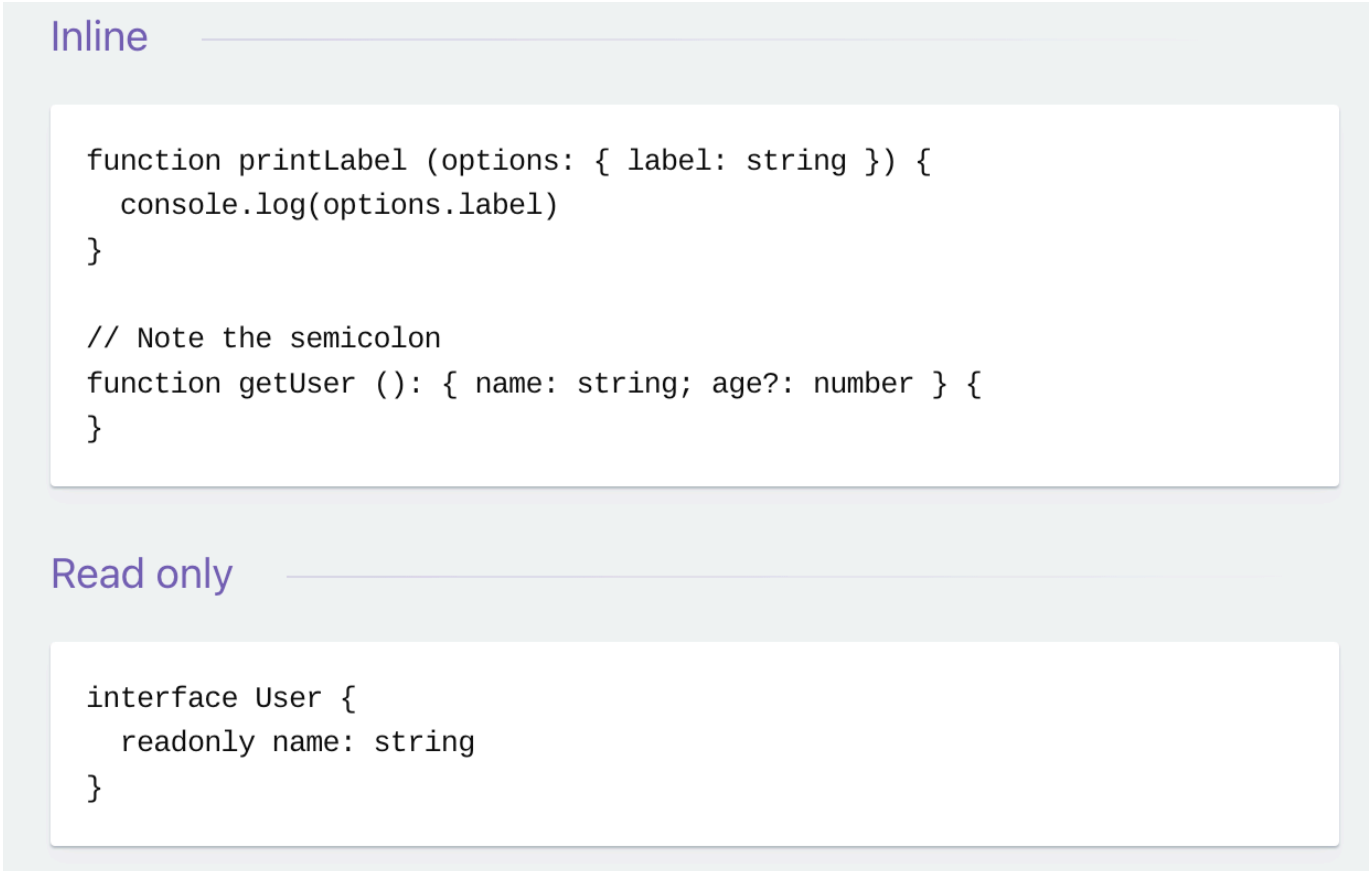
```
interface LabelOptions {
  label: string
}

function printLabel(options: LabelOptions) { ... }
```

Optional properties

```
interface User {
  name: string;
  age?: number;
}
```

Interfaces 2



Inline

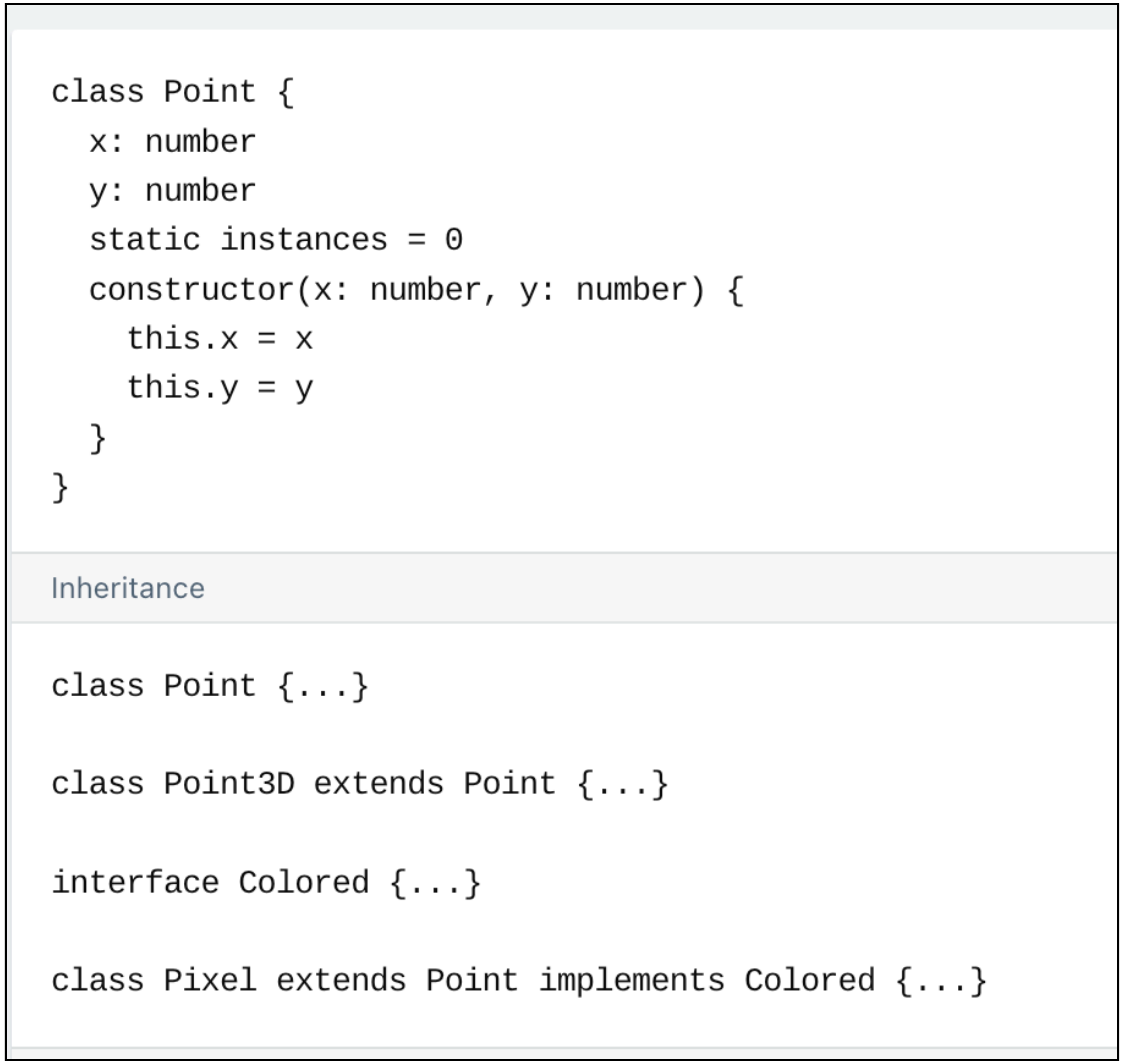
```
function printLabel (options: { label: string }) {
  console.log(options.label)
}

// Note the semicolon
function getUser (): { name: string; age?: number } {
}
```

Read only

```
interface User {
  readonly name: string
}
```

Classes



```
class Point {
  x: number
  y: number
  static instances = 0
  constructor(x: number, y: number) {
    this.x = x
    this.y = y
  }
}
```

Inheritance

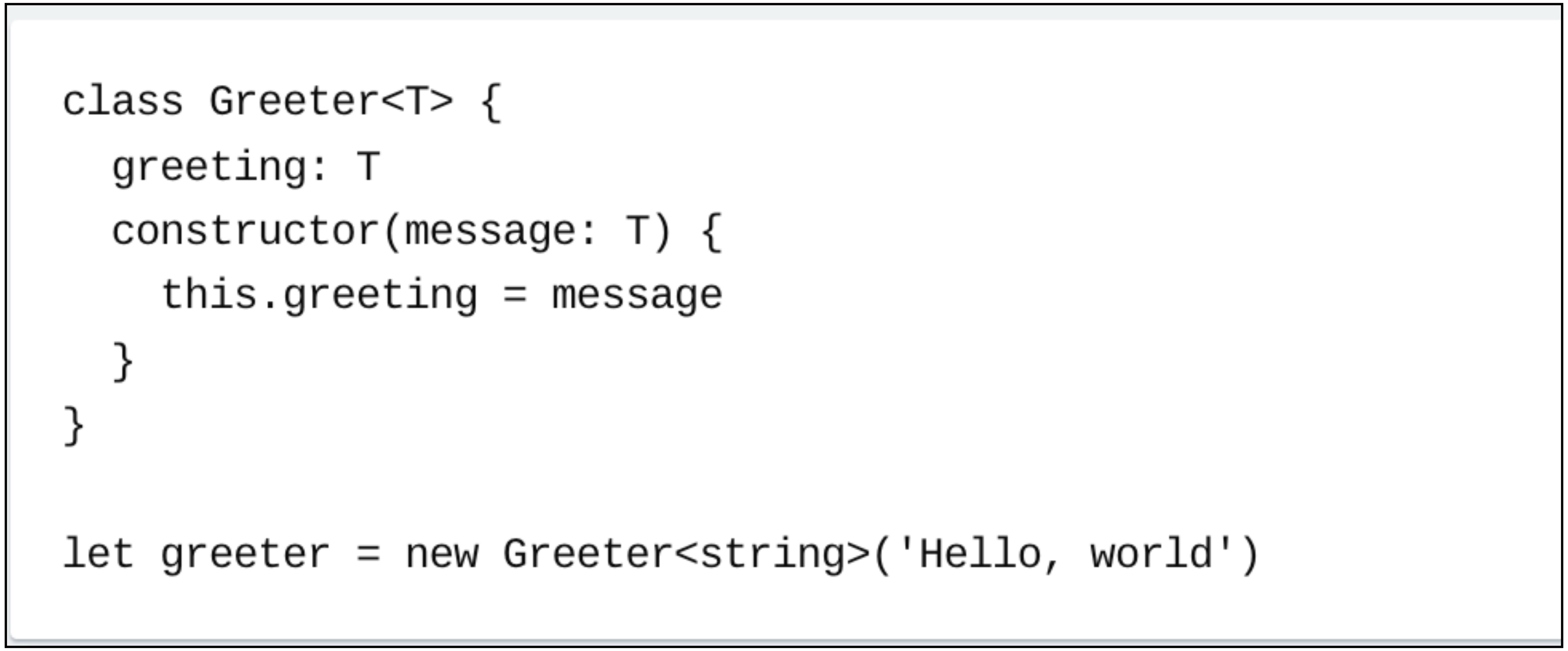
```
class Point {...}

class Point3D extends Point {...}

interface Colored {...}

class Pixel extends Point implements Colored {...}
```

Generics



```
class Greeter<T> {
  greeting: T
  constructor(message: T) {
    this.greeting = message
  }
}

let greeter = new Greeter<string>('Hello, world')
```

Type Assertions

```
let len: number = (input as string).length
let len: number = (<string> input).length
```

Function Types

```
interface User { ... }

function getUser(callback: (user: User) => any) { callback({...}) }

getUser(function (user: User) { ... })
```

Typescript

CHEATSHEET FOR

TypeScript

Typescript for Javascript programmers

https://devhints.io/typescript