[@class类声明注解 — EmmyLua for IntelliJ IDEA 1.3.2 文档](https://emmylua.github.io/zh_CN/annotations/class.html)

**Class：local变量、global变量**

--**@class** 子类[:父类] [**@注释**]

--**@class** MY\_TYPE[:PARENT\_TYPE] [**@comment**]

---@class Car:Transport @这里写注释

**local** cls = class()

**function** **cls**:test()

**end**

**Type：local变量、global变量、Property属性**

---**@type** 类型[|其他类型] [**@注释**]

---**@type** MY\_TYPE[|OTHER\_TYPE] [**@comment**]

---@type Car @单类型，变量

**local** car1 = {}

---@type Car|Ship @多个类型，变量

**local** transport = {}

**local** obj = {}

---@type Car @属性Property

obj.car = getCar()

**Param：函数参数、回调函数参数、for循环参数**

---**@param** 名字 类型[|其他类型] [**@注释**]

---**@param** param\_name MY\_TYPE[|other\_type] [**@comment**]

---@param car Car @函数参数

**local** **function** setCar(car)

...

**end**

---@param car Car @回调函数参数

setCallback(**function**(car)

...

**end**)

---@param car Car @for循环参数

**for** k, car **in** ipairs(list) **do**

**end**

**Return：类方法、普通方法**

---**@return** 类型[|其他类型] [**@注释**]

---**@return** MY\_TYPE[|OTHER\_TYPE] [**@comment**]

---@return Car|Ship @普通方法

**local** **function** create()

...

**End**

---这里不用 @type Car来注解，因为可以从“create”方法知道是Car的类型

**local** car\_or\_ship = create()

---@return Car @类方法

**function** **factory**:create()

...

**end**

**Field：标记类额外的属性（即使这个属性没有出现在代码里）**

---**@field** [public|protected|private] 属性名 类型[|其他类型] [**@注释**]

---**@field** [public|protected|private] field\_name FIELD\_TYPE[|OTHER\_TYPE] [**@comment**]

---@class Car

---@field public name string @给类Car添加了name的属性，会出现在代码补全提示

**local** cls = class()

**Alias：将一些复杂不容易输入的类型注册为一个新的别名**

---**@alias** 新名字 类型

---**@alias** NEW\_NAME TYPE

---下面将一个回调弄了一个别名，不知道代码提示时会是什么样，待测试

---@alias Handler fun(type: string, data: any):void

---@param handler Handler

**function** addHandler(handler)

**end**

**数组：**

---**@type** MY\_TYPE[]

*---@type Car[]*

**local** list = {}

**local** car = list[1]

*-- car. and you'll see completion*

**for** i, car **in** ipairs(list) **do**

*-- car. and you'll see completion*

**end**

**字典：**

---**@type** table<KEY\_TYPE, VALUE\_TYPE>

*---@type table<string, Car>*

**local** dict = {}

**local** car = dict['key']

*-- car. and you'll see completion*

**for** key, car **in** pairs(dict) **do**

*-- car. and you'll see completion*

**end**

**方法：**

---**@type** fun(param:MY\_TYPE):RETURN\_TYPE

*---@type fun(key:string):Car*

**local** carCreatorFn1

**local** car = carCreatorFn1('key')

*-- car. and you see code completion*

*---@type fun():Car[]*

**local** carCreatorFn2

**for** i, car **in** ipairs(carCreatorFn2()) **do**

*-- car. and you see completion*

**end**

**Generic：泛型，有点牵强，暂时不用**

**Vararg：不定参数部分的类型，暂时不用**