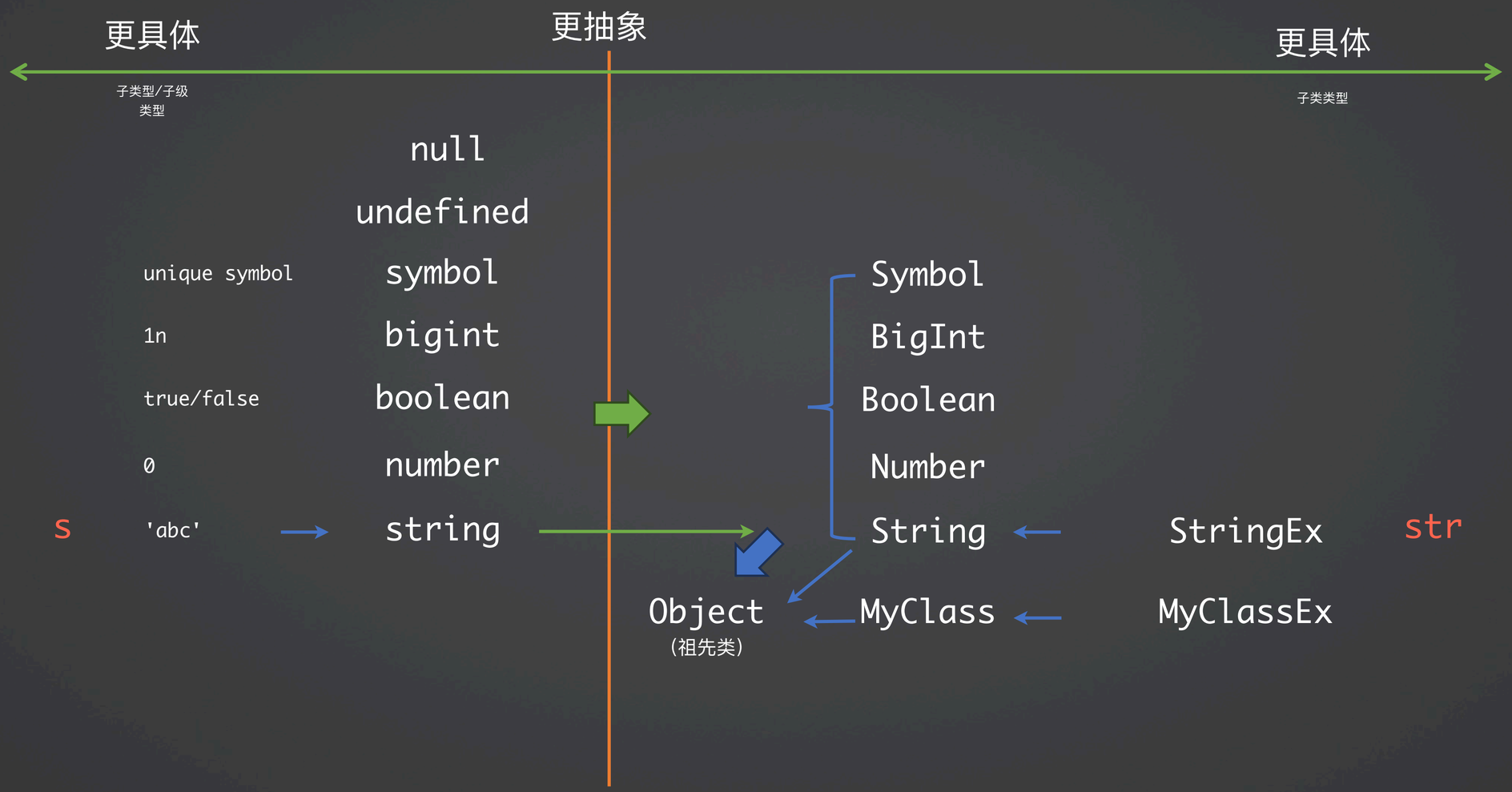
04 | 联合和交叉

周爱民（Aimingoo）

目录

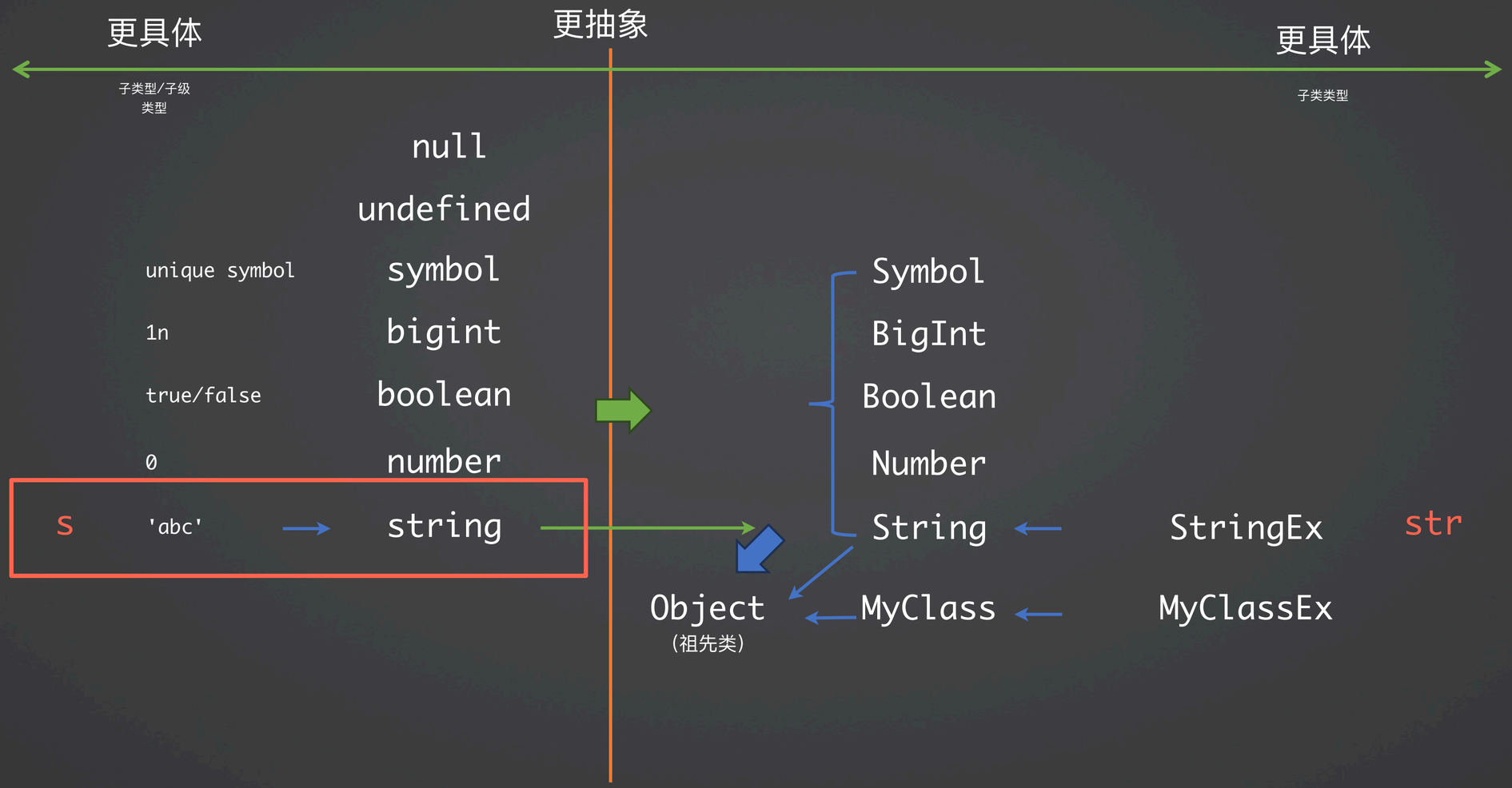
子类型兼容

类型的演化，以及最基础的类型兼容性处理

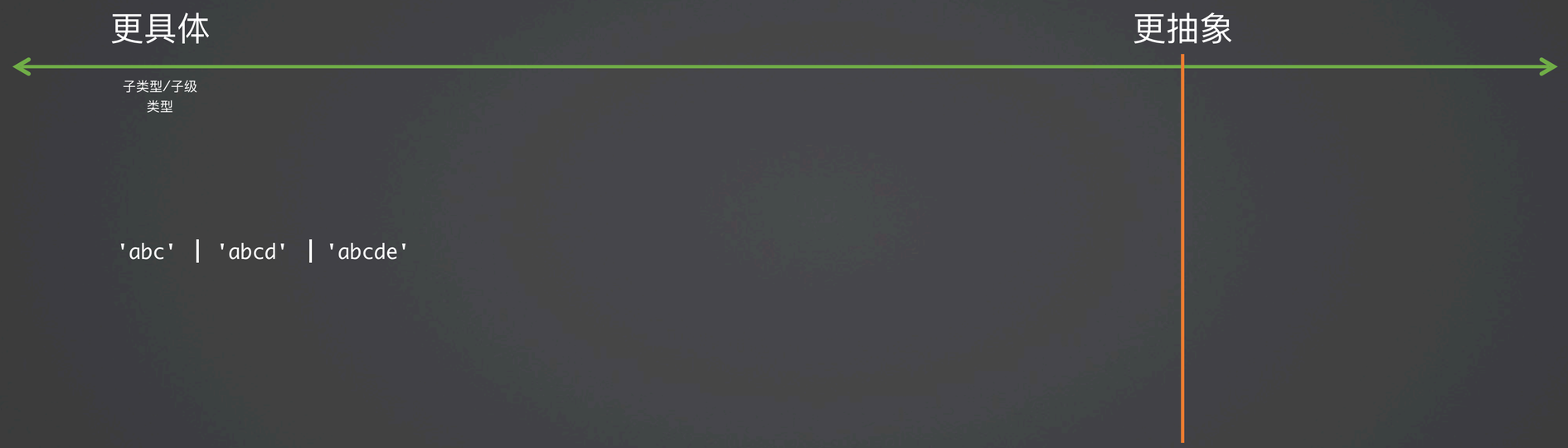


子类型兼容

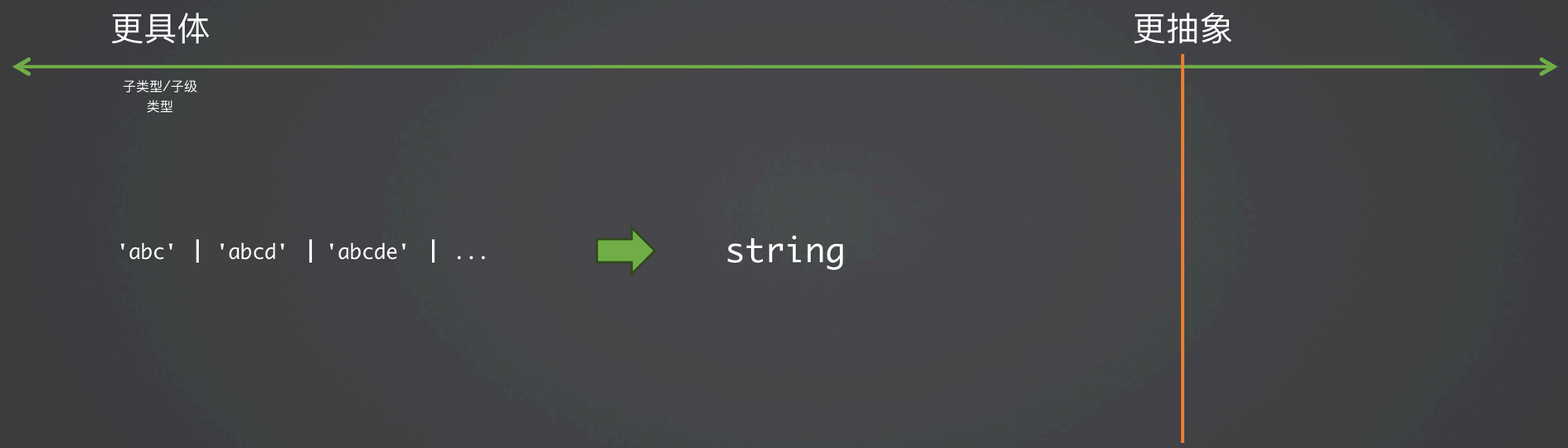
类型的演化，以及最基础的类型兼容性处理



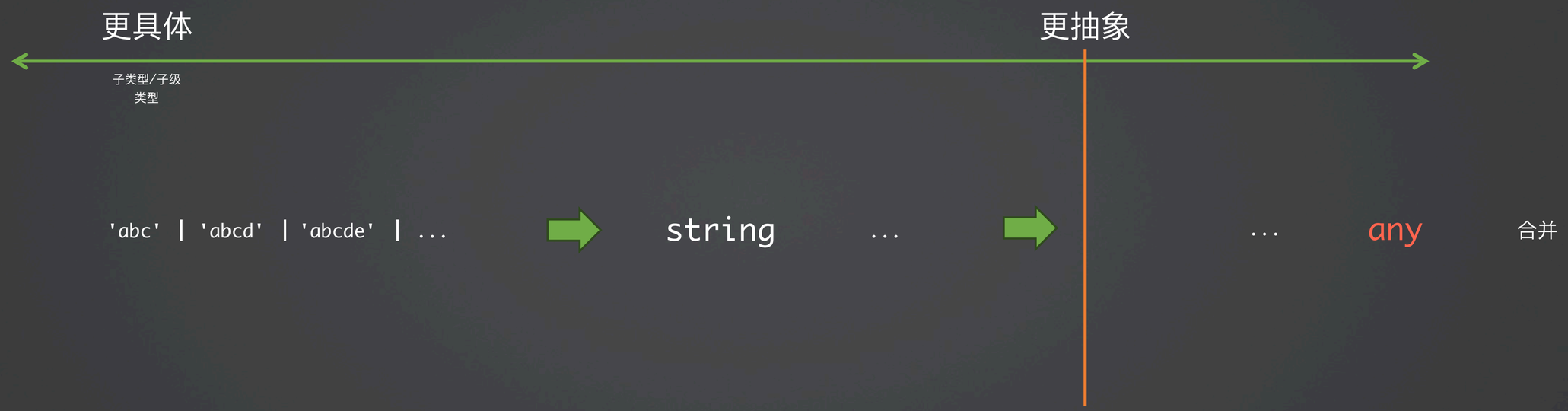
联合与交叉



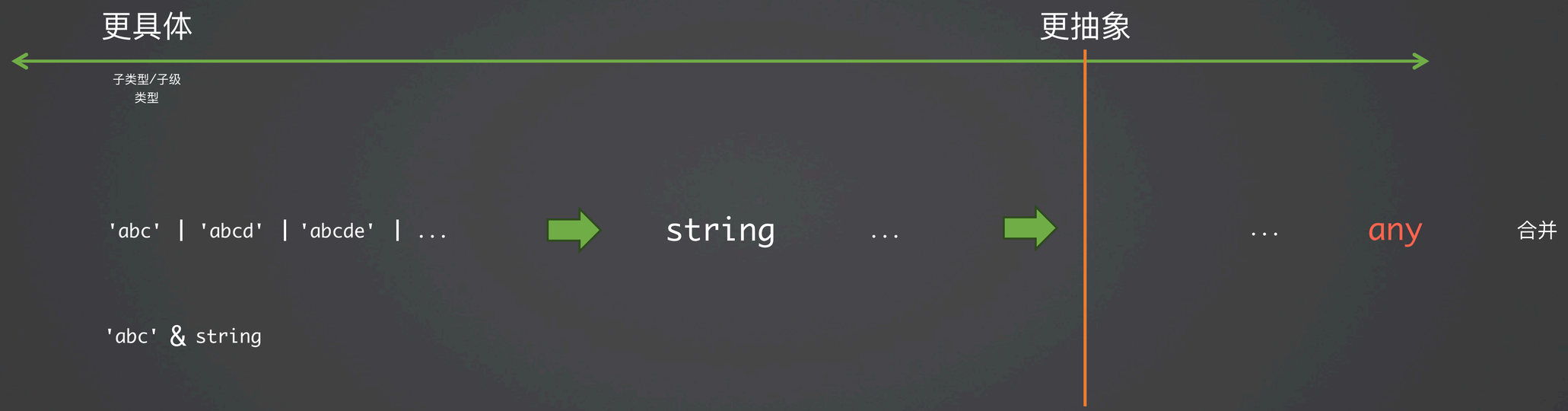
联合与交叉



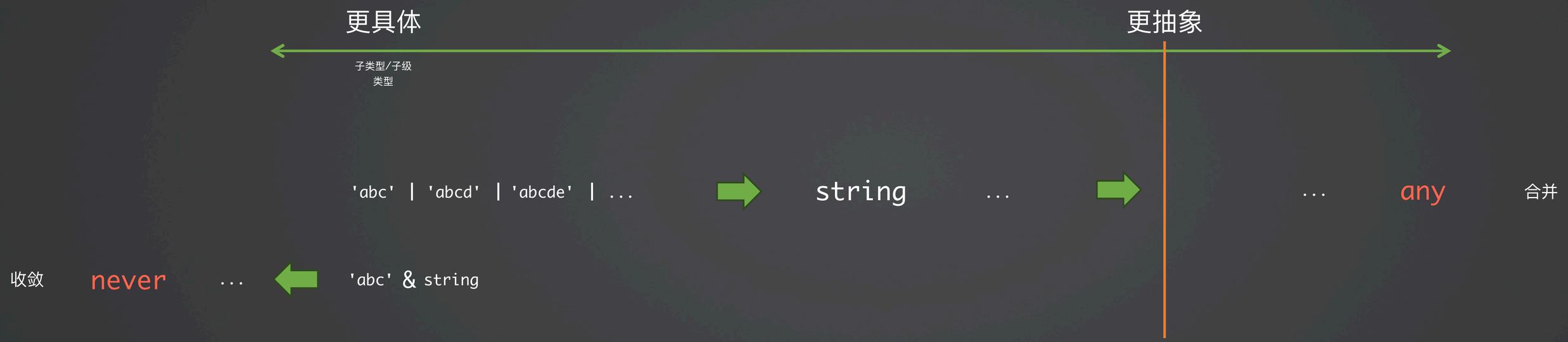
联合与交叉



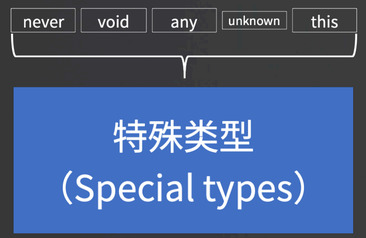
联合与交叉

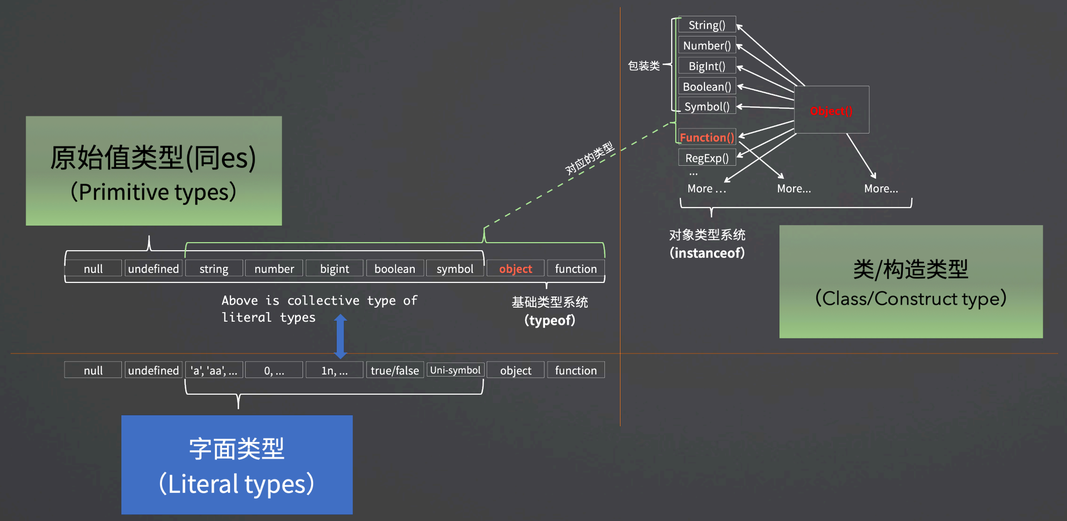


联合与交叉

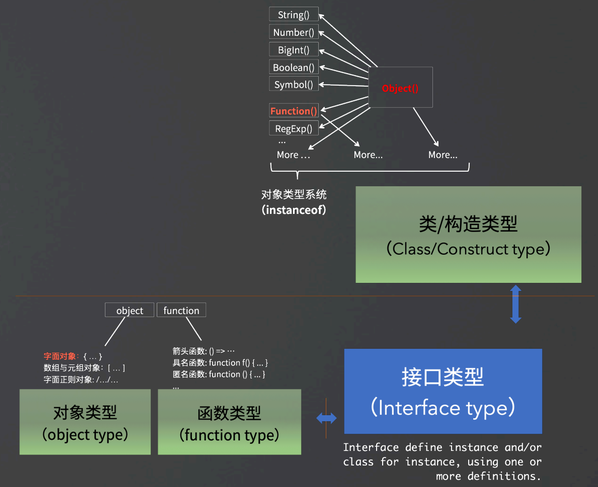

更具体
更抽象
子类型/子级
类型
'abc' | 'abcd' | 'abcde' | ...
string
...
...
any
合并
收敛
never
...
'abc' & string

赋值兼容



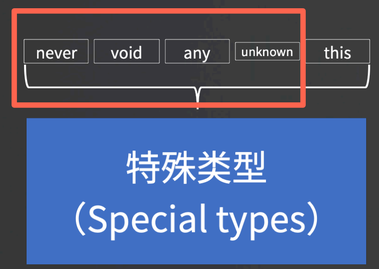


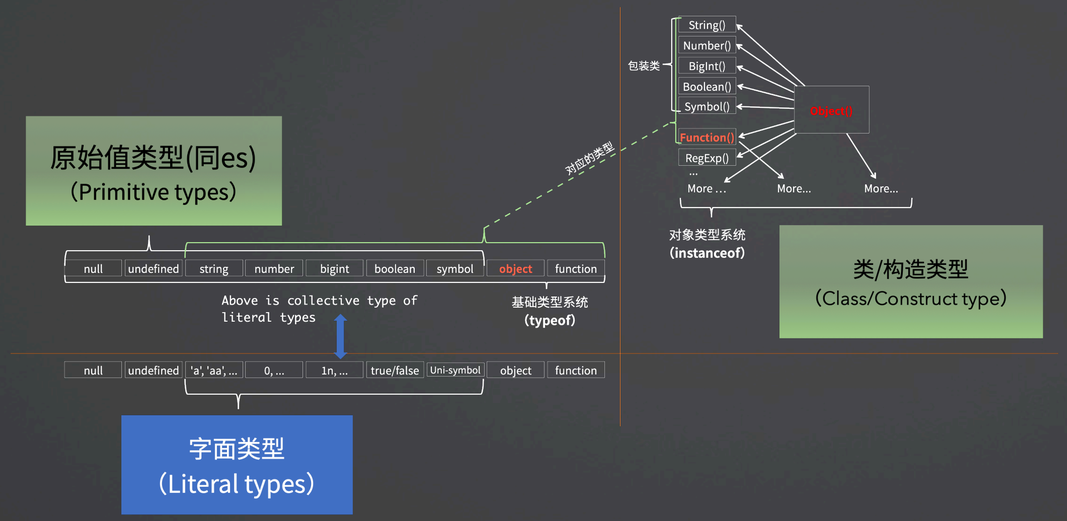
子类型兼容



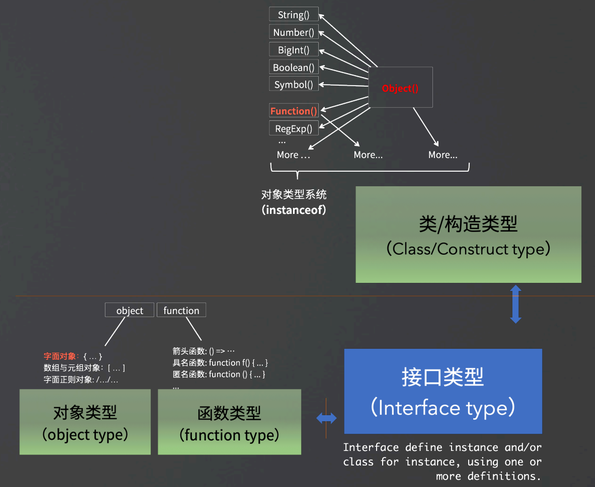
结构类型兼容

赋值兼容



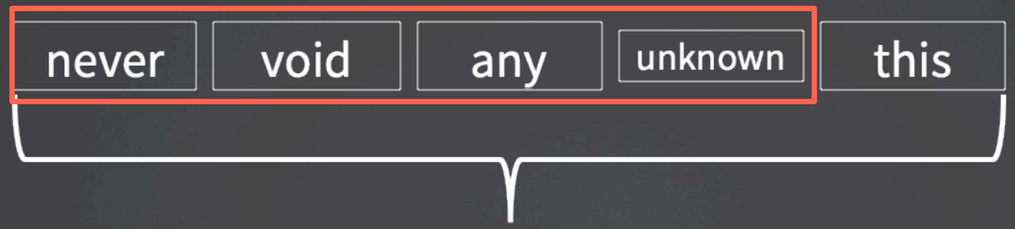


子类型兼容



结构类型兼容

赋值兼容



特殊类型
(Special types)

简单类型间的赋值兼容性

x \ T	null	undefined	void	never	any	unknown	'a'	1	true	false	string
T_null	TRUE	FALSE	FALSE	TRUE	boolean	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
T_undefined	FALSE	TRUE	FALSE	TRUE	boolean	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
T_void	FALSE	TRUE	TRUE	TRUE	boolean	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
never	FALSE	FALSE	FALSE	TRUE	boolean	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
any	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
unknown	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
'a'	FALSE	FALSE	FALSE	TRUE	boolean	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE
1	FALSE	FALSE	FALSE	TRUE	boolean	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE
T_true	FALSE	FALSE	FALSE	TRUE	boolean	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
T_false	FALSE	FALSE	FALSE	TRUE	boolean	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE
string	FALSE	FALSE	FALSE	TRUE	boolean	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE

* never 可以赋给任何类型，但只能被 any 和 never 赋值
* unknown 只能赋给 any 和 unknown，但可以被任何类型赋值

赋值兼容

```
type T1 = number | string; // both
type T2 = number & string; // never

type T3 = 'abc' | string; // string
type T4 = 'abc' & string; // 'abc'

type T5 = ... | any; // any
type T6 = ... & never; // never

type T7 = void | undefined; // both
type T8 = void & undefined; // undefined
```

2

3

1

总结

1. 联合与交叉的用法和类型规则
   - 联合会合并子类型，交叉会向子类型收敛
   - any、unknown、never 和 void 的特殊性
   - void & undefined 是特殊处理的

2. 子类型兼容和结构类型兼容，最终在概念上都统一为赋值兼容性
   - 特殊类型
   - 基础类型（或简单类型）
   - 复合类型、组合类型（或复杂类型）

名词/概念

名词、术语

联合类型：	Union types
交叉类型：	Intersection types
顶类型、底类型、单元类型：	Top type, Bottom type, Unit type

概念

- 联合类型（Union type）是由两个或多个其他类型形成的类型，表示可以是这些类型中的任何一种类型的值（*representing values that may be any one of those types*）。
  @see: https://www.typescriptlang.org/docs/handbook/2/everyday-types.html#union-types
- 交叉类型（Intersection type）是由两个或多个其他类型形成的类型，表示这些类型的公共子类型（Common subtyping）。
- 类型void 用于表示函数没有返回（without return statement），或不返回值（*don't return a value*）。对于TypeScript来说，它在语义上与 “无返回值的return语句” 相同。而在JavaScript中由于“无返回值的return语句”与“return undefined”等效（*implicitly return the value undefined*），这迫使TypeScript加入了void与undefined间的类型兼容。
  @see: https://www.typescriptlang.org/docs/handbook/2/functions.html#void
- 类型undefined和null是JavaScript内建类型（*Built-in types*）Undefined、 Null所对应的原始类型（*corresponding primitive types*），它们可以视为这些原始类型的唯一字面类型（literal types）。与此类似，类型boolean是与内建类型Boolean对应的原始类型，而true/false是它唯二的字面类型。
  @see: https://www.typescriptlang.org/docs/handbook/typescript-in-5-minutes-func.html#built-in-types

名词/概念

概念

- 所谓底类型（Bottom type），是指所有其它类型的子类型（*is a subtype of all other types*）；而所谓顶类型，有时也称为通用类型，它是指所有其它类型的父类型（*universal type, or universal supertype as all other types*）。所谓单元类型（Unit type）是指只包括一个值的类型，类似于JavaScript中的单值，或TypeScript中的字面类型；在TypeScript中它被定义为原始值类型（Primitive types）的“每一个具体值（*exactly one primitive value*）”，是其对应类型的子类型（*subtypes of primitive types*）。

  @see: https://en.wikipedia.org/wiki/Top_type

  @see: https://en.wikipedia.org/wiki/Bottom_type

  @see: https://www.typescriptlang.org/docs/handbook/typescript-in-5-minutes-func.html#unit-types
- 类型never是底类型（bottom type），用于表示从未被观察到的值（*never observed*），它表值不应存在、不可得到，或可以忽略，也用于表达类型运算的结果没有意义。

  @see: https://www.typescriptlang.org/docs/handbook/2/functions.html#never
- 类型any与unknown都是顶类型（*top type*），可以被赋以任何类型的值。any表示完全不设限制的“任何类型”，可以自由地赋值或被赋值（*from/to*）到其它类型；而unknown表示任何“不能确知”的类型，由于“不能确知”所以不能赋值给别的类型（*it's not legal to do anything with an unknown value*），只能被其它已知类型覆盖，安全性更高（*is safer*）。

  @see: https://www.typescriptlang.org/docs/handbook/2/functions.html#unknown

  @see: https://www.typescriptlang.org/docs/handbook/typescript-in-5-minutes-func.html#other-important-typescript-types

Q&A

Q: 什么是基础类型和复合类型

A: 自身不是其它类型的一部分的类型，称为基础类型，包括特殊类型、字面类型和原始类型；复合类型由其它类型按组合规则复合而成。在这种描述下，联合、交叉，以及所有能用接口描述的（对象、类、枚举、元组等）都是复合类型的。

Q: 在课程中，为什么在 id 的类型中联合了一个 Object 类型就出错了

A: 这里涉及到一个在第 13 讲中的知识点，当 id 的类型中包含 `Object` 时， getInputSomething() 的返回和赋值将导致 id 的类型被收窄至 `string | Object`，而非预期的 `string`，因为收窄时发生的交叉运算受到了 `Object` 这个类型的影响。

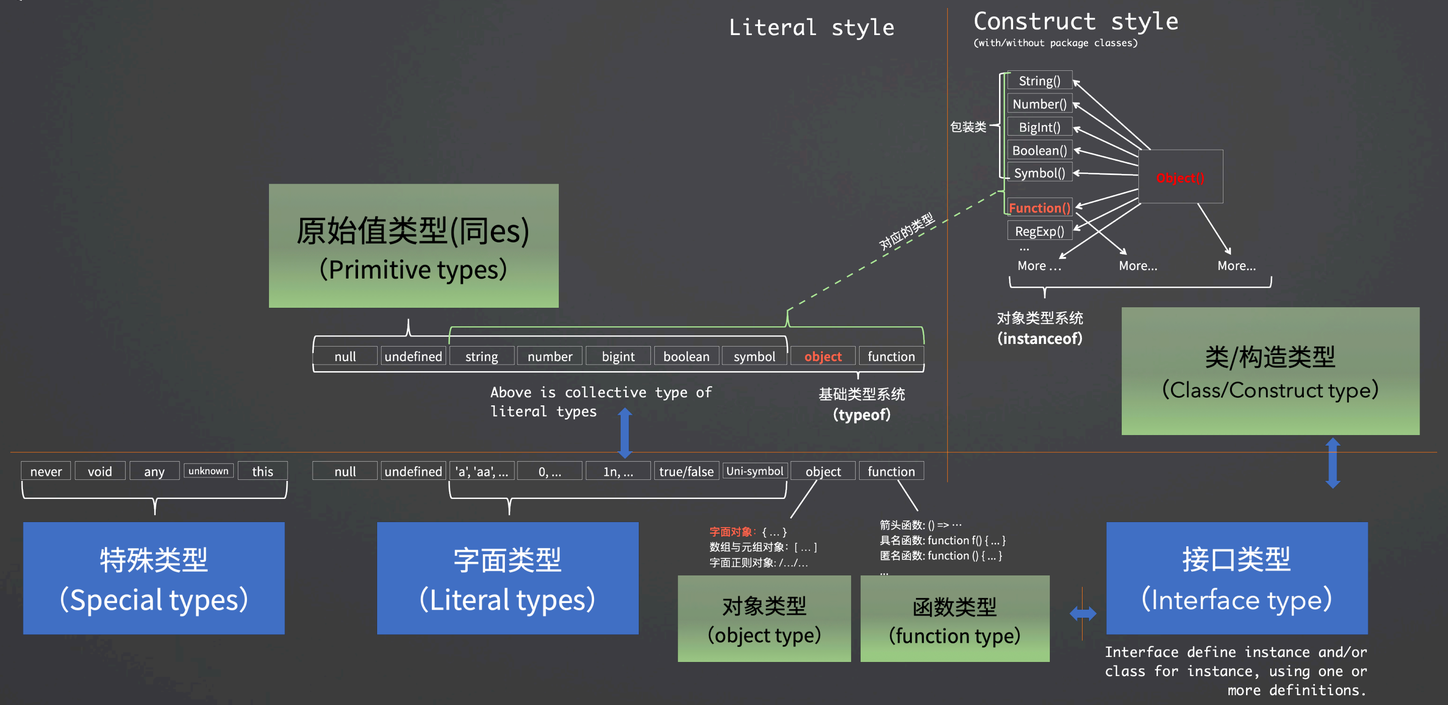
Q: PPT中的“简单类型间的赋值兼容性”表格是如何得到的

A: 它是使用“extends 表达式”穷举运算得出的，不过这涉及到下一篇（第15讲之后）的内容，因此具体的做法被略去了。如果你想初步了解一下这些兼容性规则，可以参考如下链接：

https://www.typescriptlang.org/docs/handbook/type-compatibility.html#any-unknown-object-void-undefined-null-and-never-assignability

课后作业

1. 思考题：为什么在之前的类型兼容的划分中（例如结构类型兼容），没有包括联合和交叉？




极客时间
THANKS