14 | 在 JS 和 TS 之间互通访问的技术

周爱民（Aimingoo）

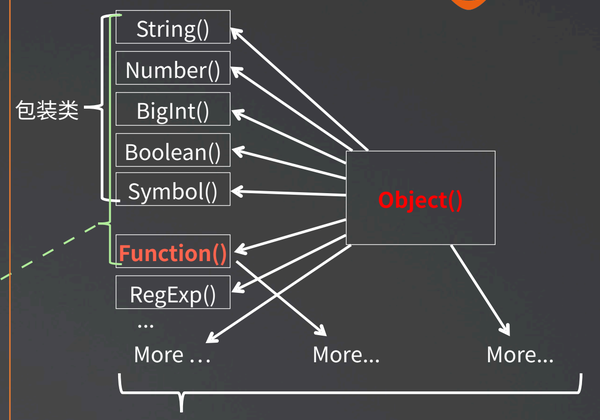
目录

JS类型 => TS类型

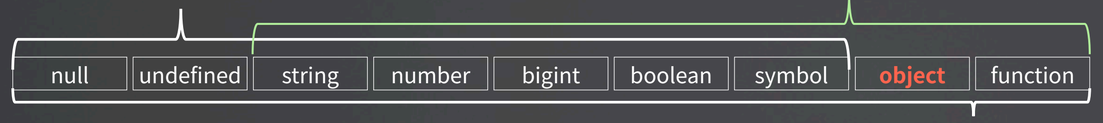
Literal style

Construct style
(with/without package classes)

极客时间



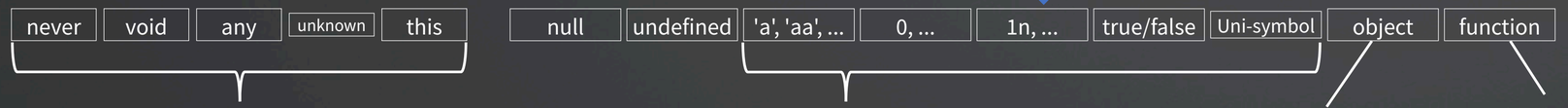
原始值类型(同es)
(Primitive types)



Above is collective type of literal types

基础类型系统
(typeof)

类/构造类型
(Class/Construct type)



特殊类型
(Special types)

字面类型
(Literal types)

字面对象：{ ... }
数组与元组对象：[...]
字面正则对象: /.../...

对象类型
(object type)

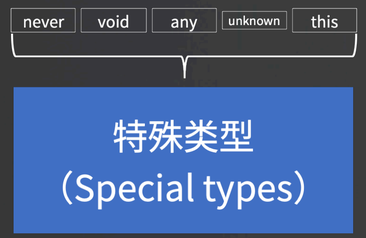
箭头函数: () => ...
具名函数: function f() { ... }
匿名函数: function () { ... }
...

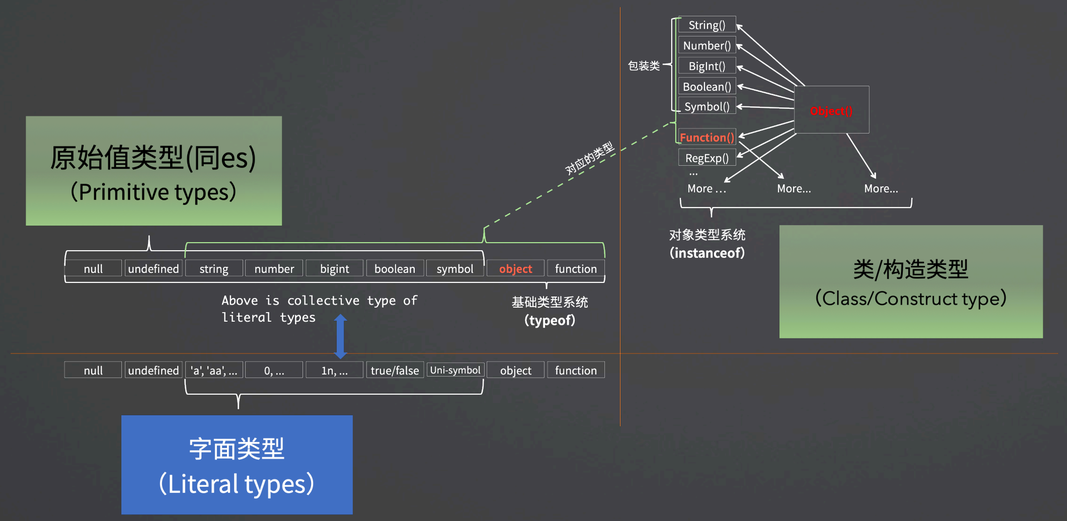
函数类型
(function type)

接口类型
(Interface type)

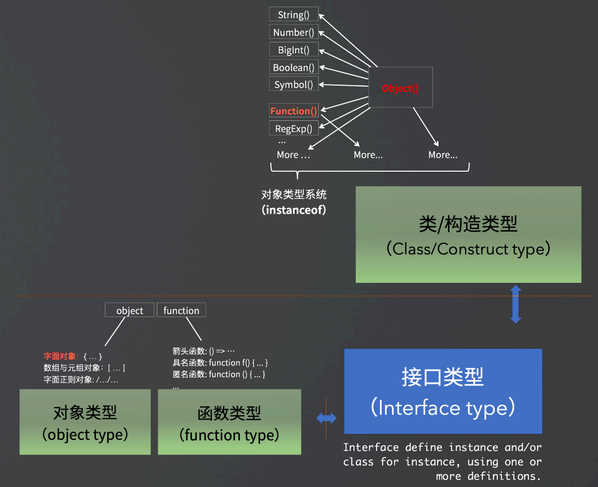
Interface define instance and/or class for instance, using one or more definitions.

赋值兼容

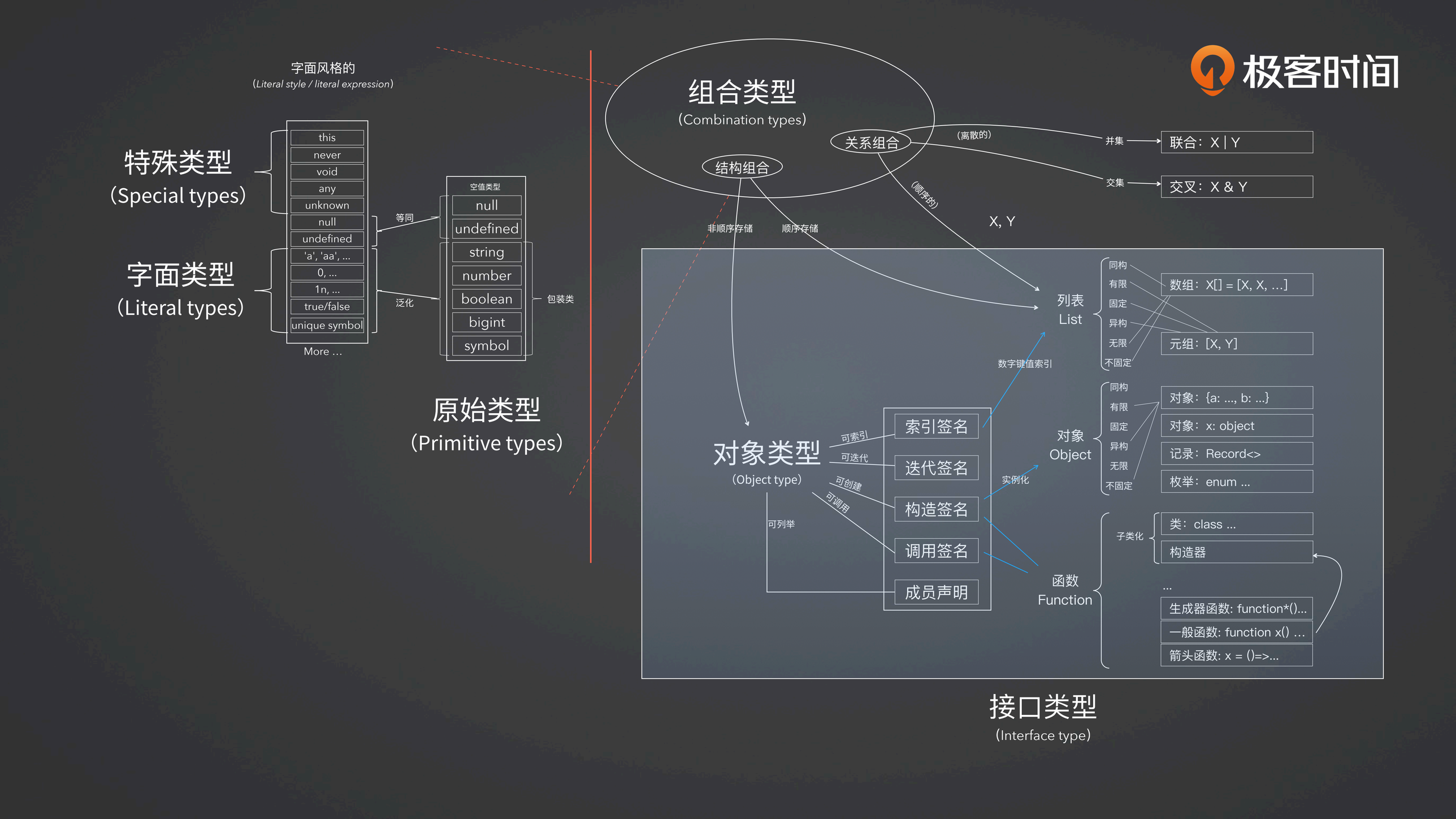




子类型兼容



结构类型兼容


字面风格的
(Literal style / literal expression)
特殊类型
(Special types)
字面类型
(Literal types)
this
never
void
any
unknown
null
undefined
'a', 'aa', ...
0, ...
1n, ...
true/false
unique symbol
More ...
等同
泛化
空值类型
null
undefined
string
number
boolean
bigint
symbol
包装类
原始类型
(Primitive types)
组合类型
(Combination types)
关系组合
结构组合
(离散的)
(顺序的)
并集
交集
联合：X | Y
交叉：X & Y
X, Y
非顺序存储
顺序存储
对象类型
(Object type)
可索引
可迭代
可创建
可调用
可列举
索引签名
迭代签名
构造签名
调用签名
成员声明
数字键值索引
实例化
列表
List
对象
Object
函数
Function
同构
有限
固定
异构
无限
不固定
数组：X[] = [X, X, ...]
元组：[X, Y]
对象：{a: ..., b: ...}
对象：x: object
记录：Record<>
枚举：enum ...
子类化
类：class ...
构造器
...
生成器函数: function*()...
一般函数: function x() ...
箭头函数: x = ()=>...
接口类型
(Interface type)

declare scope

名字空间 类型 值

Declaration Type	Namespace	Type	Value
Namespace	X		X 2
Class		X	X
Enum		X	X
Interface		X	
Type Alias		X	
Function	X 1		X
Variable			X

https://typescriptlang.org/docs/handbook/declaration-merging.html

关键的类型运算

```
type X = typeof x
type U = keyof T

type T1 = Omit<T, never>
type U1 = Exclude<U, ...>

type T2 = ... | ...
type T3 = ... & ...

// 其它（例如可计算常量表达式等）
const x = … as const
enum X {
  ...
}
```

执行流分析与类型推断

1. 类型识别（类型推断）

```
x = ...
...

if (typeof x)
  ...
```

2. 谓词签名
   断言签名

3. 可辨识联合类型

总结

1. 在 js 和 ts 之间互通访问
   - 在 ts 中从 js 声明中获取和推断类型（例如 typeof 运算）
   - 在 js 代码中通过 typeof、instanceof、in 以及定制的守护函数来实现流程中的类型收窄
   - 在 js 中能访问的两种 ts 类型（namespace 和 enum）

2. 从 js/es 的类型系统，到 ts 扩展的类型系统
   - 三种类型兼容性

3. 一些简单的类型计算，例如 keyof 和 Omit<>


极客时间
THANKS