

# 极客大学 前端进阶训练营

程劭非 (winter)

前手机淘宝前端负责人

# 结构化程序设计

# JS执行粒度

- 宏任务
- 微任务 (Promise)
- 函数调用 (Execution Context)
- 语句/声明
- 表达式
- 直接量/变量/this .....

# Realm

- JS Context => Realm
- 宏任务
- 微任务 (Promise)
- 函数调用 (Execution Context)
- 语句/声明
- 表达式
- 直接量/变量/this .....

# 函数调用

```
import {foo} from "foo.js"  
var i = 0;  
console.log(i);  
foo();  
console.log(i);  
i++;
```

```
function foo(){  
    console.log(i);  
}  
export foo;
```

# 函数调用

```
var i = 0;  
console.log(i);  
console.log(i);  
console.log(i);  
i++;
```

# 函数调用

```
import {foo} from "foo.js";  
var i = 0;  
console.log(i);  
foo();  
console.log(i);  
i++;
```

```
import {foo2} from "foo.js";  
var x = 1;  
function foo(){  
    console.log(x);  
    foo2();  
    console.log(x);  
}  
export foo;
```

```
var y = 2;  
function foo2(){  
    console.log(y);  
}  
export foo2;
```

# 函数调用

```
var i = 0;  
console.log(i);  
console.log(x);  
console.log(y);  
console.log(x);  
console.log(i);  
i++;
```



# 函数调用

i:0

```
var i = 0;  
console.log(i);  
  
console.log(i);  
i++;
```

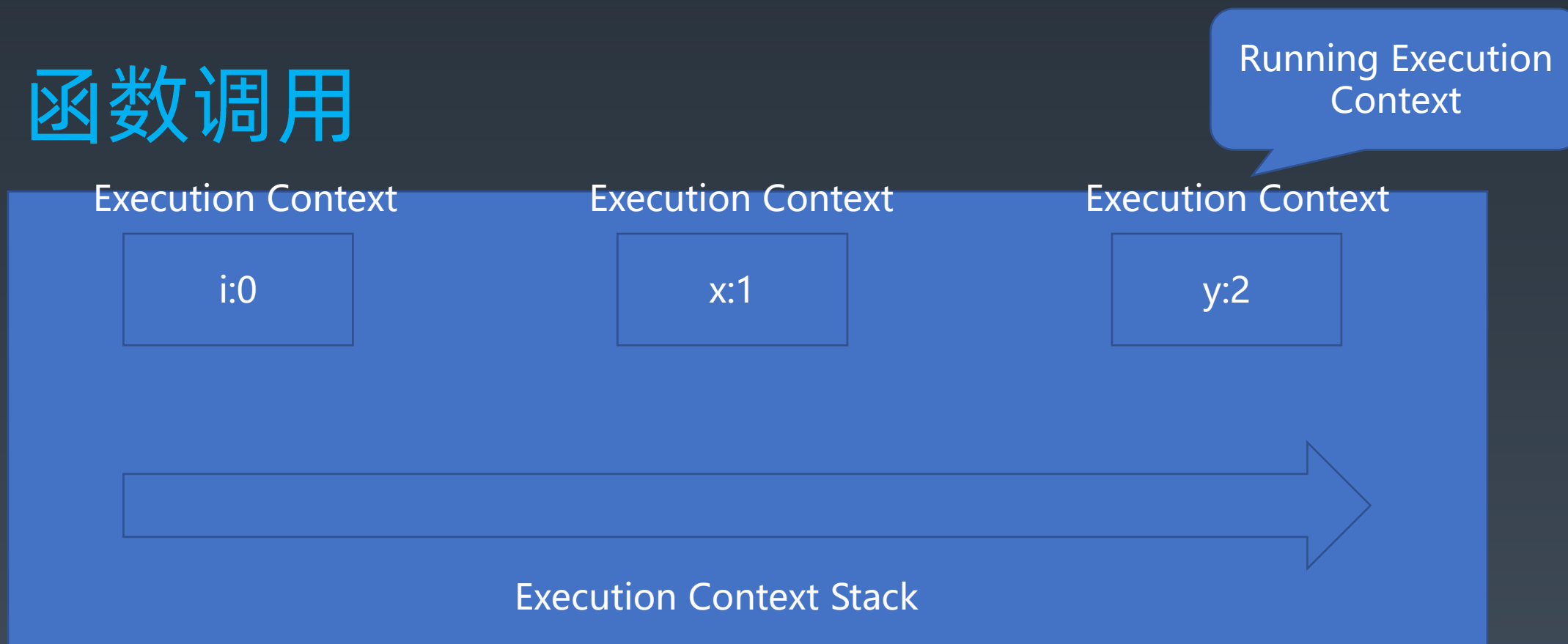
x:1

```
console.log(x);  
  
console.log(x);
```

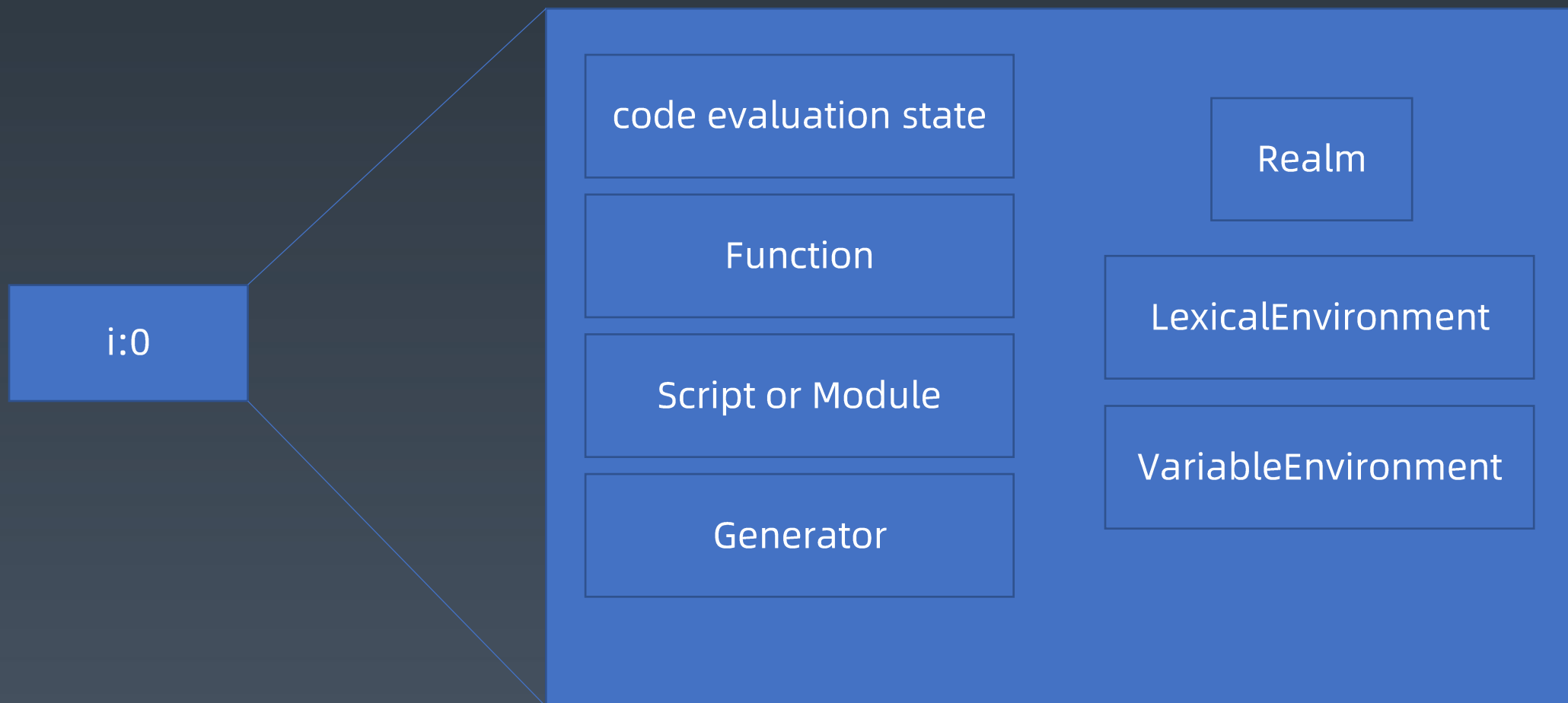
y:2

```
console.log(y);
```

# 函数调用



# Execution Context



# Execution Context

- ECMAScript Code Execution Context
  - code evaluation state
  - Function
  - Script or Module
  - Realm
  - LexicalEnvironment
  - VariableEnvironment
- Generator Execution Contexts
  - code evaluation state
  - Function
  - Script or Module
  - Realm
  - LexicalEnvironment
  - VariableEnvironment
  - Generator

# LexicalEnvironment

- this
- new.target
- super
- 变量

```
this.a = 1;  
super();  
x += 2;  
new.target;
```

# VariableEnvironment

VariableEnvironment是个历史遗留的包袱，仅仅用于处理var声明。

```
{  
  let y = 2;  
  eval('var x = 1;');  
}  
  
with({a:1}){  
  eval('var x;');  
}  
console.log(x);
```

# Environment Record



# Function - Closure

```
var y = 2;  
function foo2(){  
  console.log(y);  
}  
export foo2;
```

Function: foo2

Environment Record:  
y:2

Code:  
console.log(y);



# Function - Closure

```
var y = 2;  
function foo2(){  
  var z = 3;  
  return () => {  
    console.log(y, z);  
  }  
}  
var foo3 = foo2();  
export foo3;
```

Function: foo3

Environment Record:  
z:3  
this:global

Code:  
console.log(y, z);

Environment Record:  
y:2

# Realm

在JS中，函数表达式和对象直接量均会创建对象。

使用`.`做隐式转换也会创建对象。

这些对象也是有原型的，如果我们没有Realm，就不知道它们的原型是什么。

```
var x = {};
```

作业：直观感受一下Realm

THANKS! |  极客大学