ICP区块链开发入门课程

2. Motoko 语言简介

主讲：Paul Liu - DFINITY 工程师

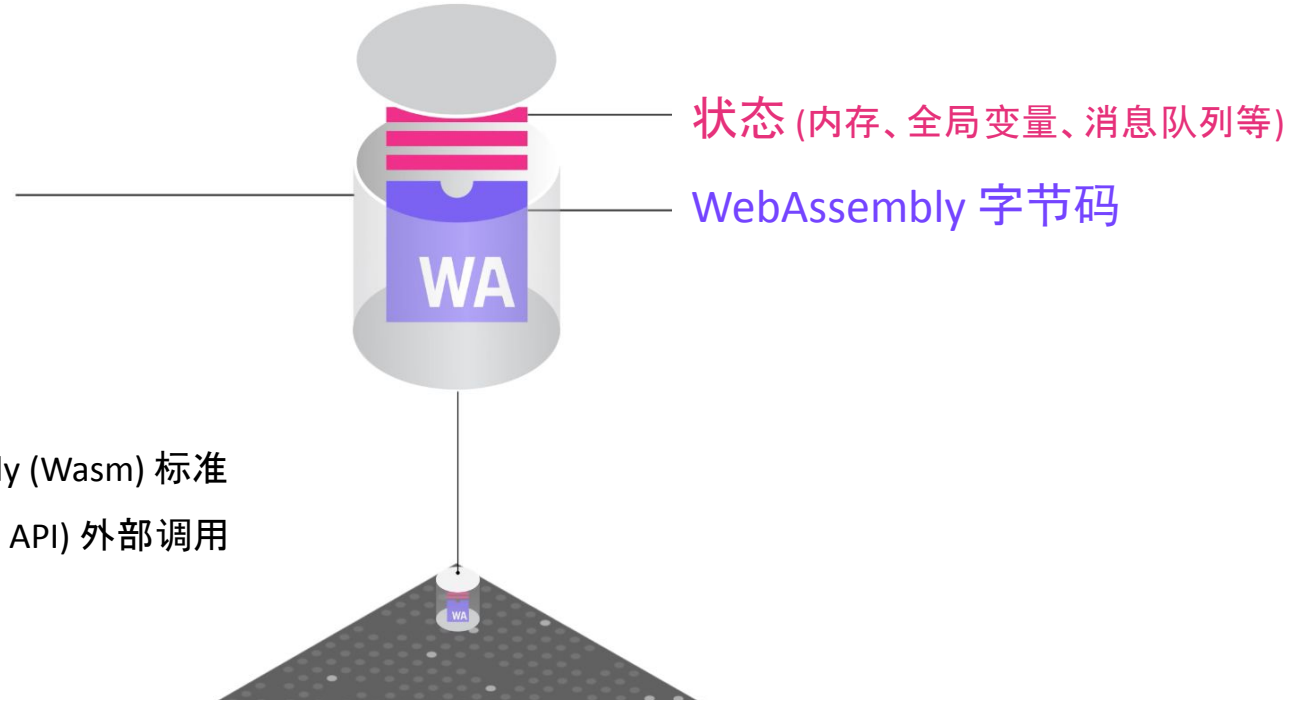
课程大纲

1. 使用 SDK 搭建一个简易网站
2. Motoko 语言简介
3. Canister 智能合约
4. 用 Motoko 做后端
5. 用 Javascript 做前端

跑在 ICP 上面的智能合约

智能合约 Canister

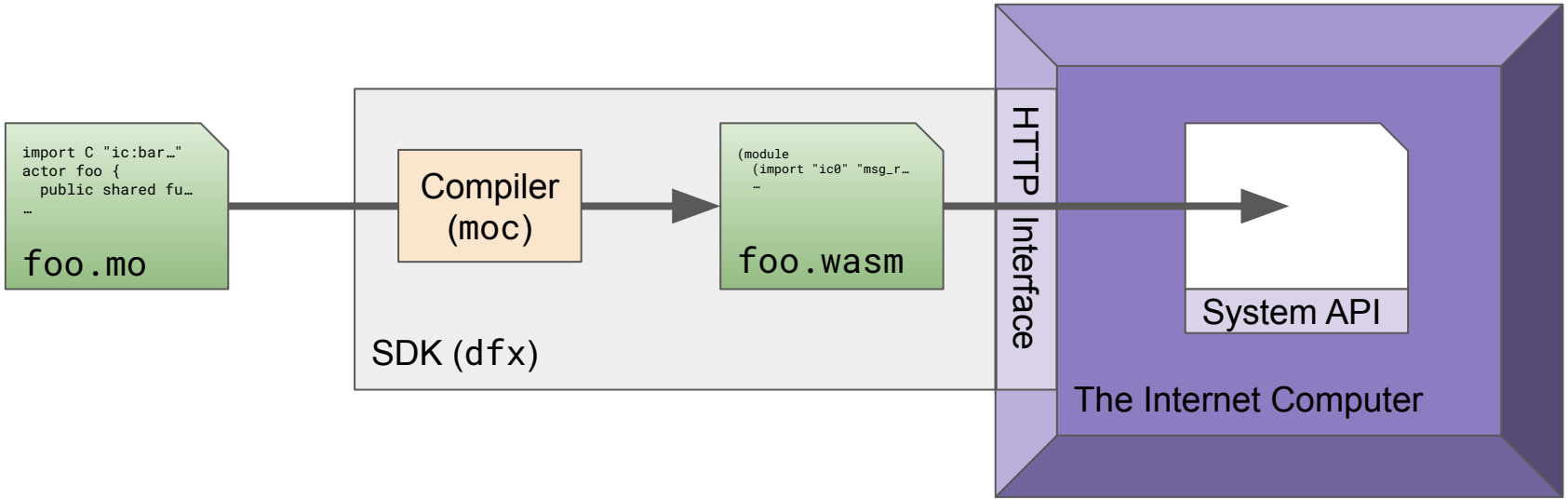
- 容器封装
- Actor 模型处理消息
- 状态自动持久化
- 代码使用 WebAssembly (Wasm) 标准
- 通过系统接口 (System API) 外部调用



Motoko 编程语言

ICP 为什么需要一门新的语言？

- 不是必须的，C, C++, Rust 都可以编译到 Wasm
- 缺少一个高级语言同时满足：安全、高效、容易上手
- 适配平台特性：Actor 模型，权限管理，代码升级，跨语言调用



Motoko 语言的特点

- 静态类型，语法接近 JavaScript/TypeScript
- 面向对象，但不支持继承
- 支持 await/async 异步通信
- 结构化类型推断
- 安全的数值计算
- 没有 NULL 指针
- 自动内存回收机制 (GC, copying/compacting/generational)

Motoko 基础概念

- 程序 (program)
- 声明 (declaration)
- 表达式 (expression)
- 值 (value)
- 变量 (variable)
- 类型 (type)

https://smartcontracts.org/docs/language-guide/motoko-introduction.html

Motoko 语法

- 注释 `/* … */`, `// …`
- 表示分隔:空格、逗号, 分号
- 变量名 `x`, `foo_bar`, `test132`, `List`, `Map`
- 圆括号 `(1 + 2) * 3`, `()`, `(1, 2)`
- 花括号 `object { a = 1; b = 2 }`
- 类型标注 `func f(x: Nat): Nat { let y = x + 1; x + y }`

Motoko 关键字

- 声明 `type var let actor func module import object label`
- 流程 `switch case try catch if then else return loop for in while break continue`
- 函数 `not or and`
- 修饰 `private public shared query stable flexible system async`
- 命令 `assert await debug debug_show ignore`
- 值 `true false null`

Motoko 示例

```
let x = 1;
let y = x + 1;
x * y + x;
```

```
let z = do {
  let x = 1;
  let y = x + 1;
  x * y + x
};
z * 2;
```

```
let x = 40; let y = 2;
let z = do {
  let x = 1;
  let y = x + 1;
  x * y + x
};
z * 2;
```

```
let x = 40; let y = 2;
ignore do {
  let x = 1;
  let y = x + 1;
  x * y + x
};
x + y
```

```
var s = 0;
var i = 1;
while (i < 10) {
    s := s + i;
    i := i + 1;
};
```

```
let a = [var 1, 2, 3, 4, 5];
var i = 1;
while (i < a.size()) {
    a[i] := a[i - 1] + a[i];
    i := i + 1;
};
```

DFINITY

Motoko 基础库

数字类型:`Int Int8 Int16 Int32 Int64 Nat Nat8 Nat16 Nat32 Nat64 Float`

常用类型：`Bool Char Array Text Option Result Iter Func None Hash`

数据结构：`Buffer List AssocList Stack Deque Heap RBTree HashMap Trie TrieMap TriSet`

系统工具：`Principal Blob Random CertifiedData Time Debug Prelude`

https://smartcontracts.org/docs/base-libraries/stdlib-intro.html

https://github.com/dfinity/motoko-base

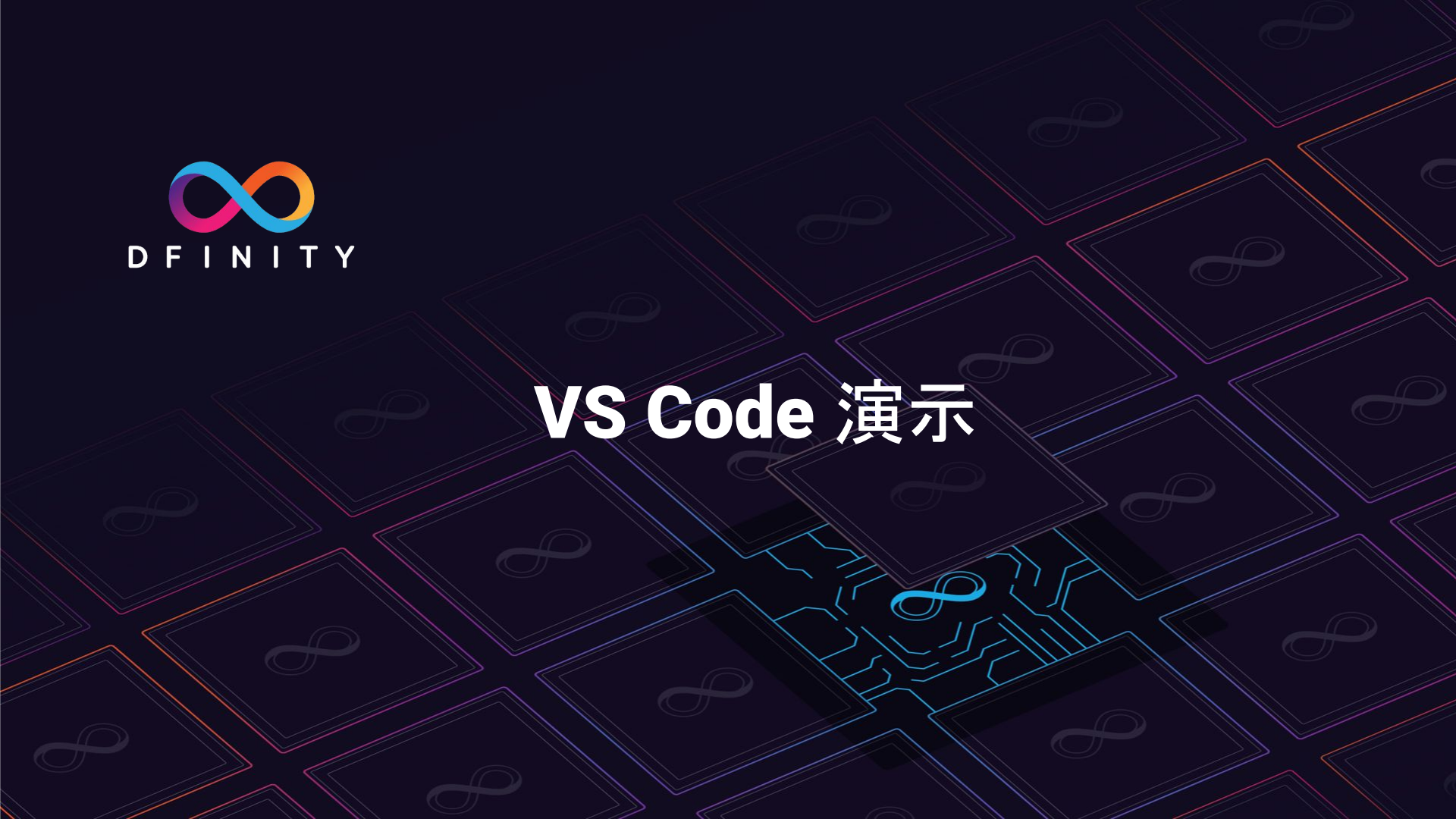
Motoko Canister

每个 Canister 都是一个 Actor，它的公共方法 (public method) 可供异步调用。

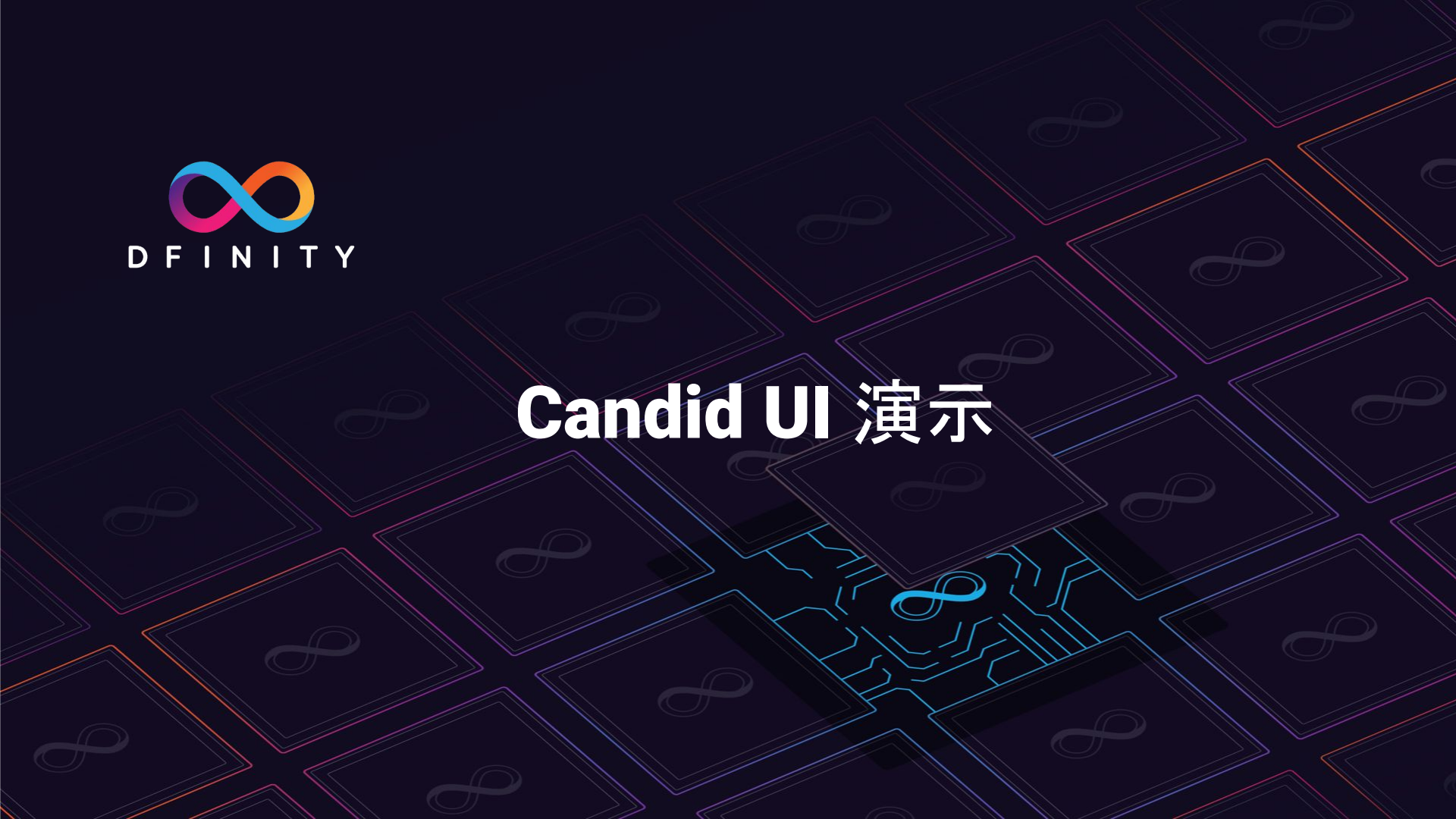
```
actor {
    public func greet(name : Text) : async Text {
        return "Hello, " # name # "!";
    };
};
```

数据描述语言 Candid 用来规范 Canister 所提供的数据类型，服务接口等。

```
service : {
  greet: (text) -> (text);
}
```



VS Code 演示



Candid UI 演示

Motoko Playground 演示

课程作业

用 motoko 实现一个快排函数：

```
quicksort : [var Int] -> ()
```

要求：

1. 用 moc 调试运行
2. 把函数封装在一个 canister 里面

```
public func qsort(arr: [Int]): async [Int]
```

3. 部署到主网
4. 使用主网的 Candid UI 调试运行

https://a4gq6-oaaaa-aaaab-qaa4q-cai.raw.ic0.app

下一节：Canister 开发实例

- 公共接口调用 (query vs. update)
- Canister 的生命周期
- Cycles 计费标准
- 网页请求 (http_request) 接口标准