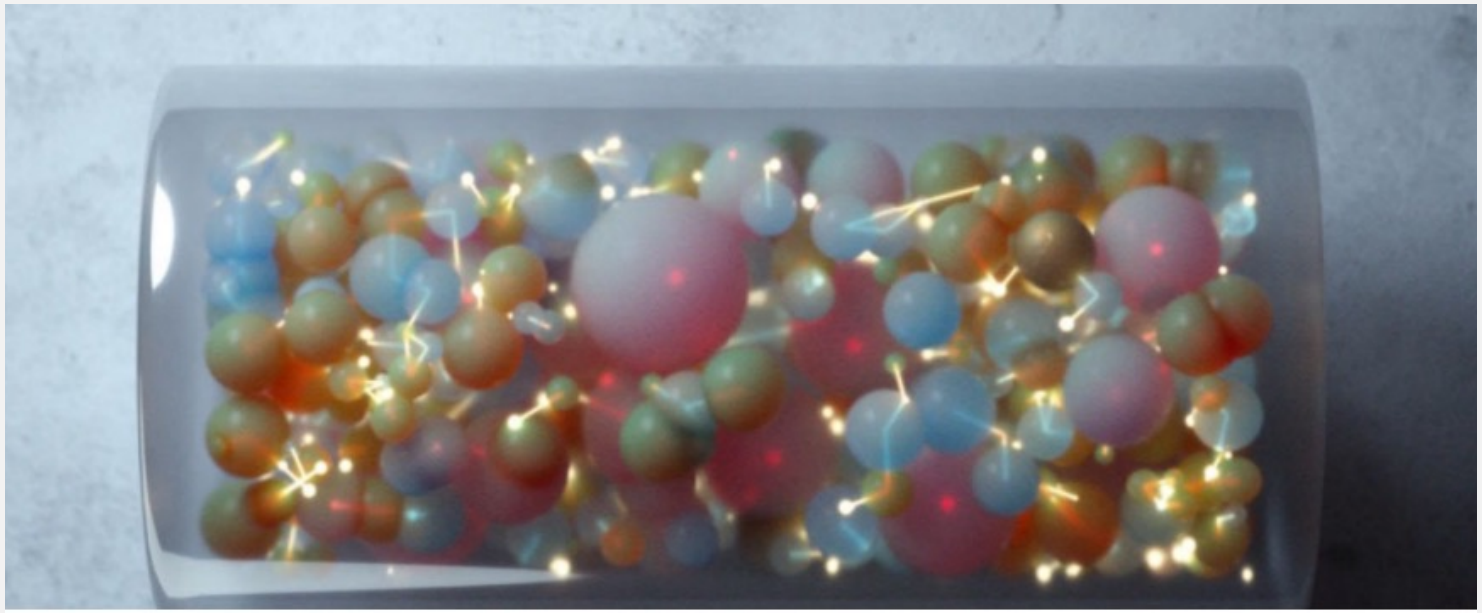
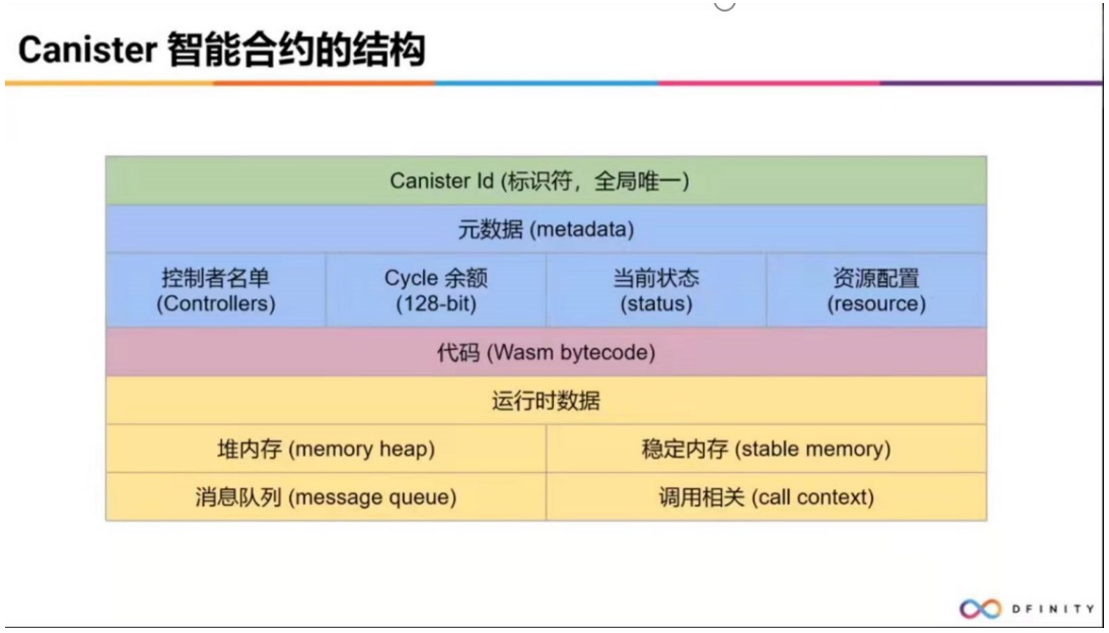


Canister智能合约

结构

- Canister
  - 元数据初始化
  - dfx canister create canister_name canister_args
- WebAssembly bytecode
  - dfx canister install wasm
- Runtime
  - Heap
  - Stable Memory
  - Message FIFO Queue

```
dfx-canister-create
Creates an empty canister on the Internet Computer and associates the Internet Computer assigned
Canister ID to the canister name

USAGE:
    dfx canister create [FLAGS] [OPTIONS] [--] [canister-name]

ARGS:
    <canister-name>    Specifies the canister name. Either this or the --all flag are required

FLAGS:
        --all         Creates all canisters configured in dfx.json
    -h, --help        Prints help information
    -V, --version     Prints version information

OPTIONS:
    -c, --compute-allocation <compute-allocation>
            Specifies the canister's compute allocation. This should be a percent in the range
            [0..100]

        --controller <controller>...
            Specifies the identity name or the principal of the new controller

        --memory-allocation <memory-allocation>
            Specifies how much memory the canister is allowed to use in total. This should be a
            value in the range [0..12 GiB] A setting of 0 means the canister will have access to
            memory on a "best-effort" basis: It will only be charged for the memory it uses, but at
            any point in time may stop running if it tries to allocate more memory when there isn't
            space available on the subnet

        --with-cycles <with-cycles>
            Specifies the initial cycle balance to deposit into the newly created canister. The
            specified amount needs to take the canister create fee into account. This amount is
            deducted from the wallet's cycle balance
```



消息处理模型

- Canister具有多个消息处理队列，分别是Canister-Canister Queue 和 Canister - User Queue。
- Iner-Canister Queue：在一个子网内， Inter-Canister Call时，每个Canister之间都有一个队列，采用RR调度，以防Canister DOS。对跨子网通信， 子网间只有一个消息队列。
- Canister-User Queue: 无限队列，IC保证。

Canister A	Canister B	Canister C	Canister D
Message 0	Message 0	Message 0	Message 0
	Message 1		Message 1

update - query

certificated data library

update-await & callback function

Defensive Programming

Query During Update

```
actor {

    var data = 0;

    public query func get() : Nat{
        data
    };

    public shared func set() : Nat{
        data := 1;
        data
    };

};
```

- 当Update消息执行，但是未写入memory时，query返回的是未修改的数据
- 初始化：
  - get：data = 0
- set start：
  - |此时query，那么get的值仍为0
  - | data：0 -> 1
  - set finish：1 ------此时query， get的值为1

Certificated Data

- 为了解决上述问题， DFINITY提供了Certificated Data Buffer ， 大小为 32 byte
- 我们可以在Update call执行时修改某些状态，且可以对query产生作用
- 初始化：
  - get : data = 0

Must be called from an update method, else traps. Must be passed a blob of at most 32 bytes, else traps.

- set start :
  - | data = 1 -> certificaed data buffer : query get的值为1
  - | data : 0 -> 1
  - set finish : 1 ------query， get的值为1

update-await & callback function

```
大致执行图

canister state :

    var a = 100;

从上向下代表时间函数执行时间流程图

Function A 执行

if(a >= 100){

    await other_canister.function(); // Function A 发送 await 消息

                                        Function B 执行

                                            a := 0;

                                        Function B 结束

Function A await 消息返回， 开始执行

    a := 200;

Function A 结束
```

- 已知： Canister执行函数时， 如果Update call函数内有await， 那么await前和await后会当作两个update call执行， 第二个update call在消息队列中的序号和第一次的update call无关， 作为回调函数到消息队列中执行。
- 场景： 首先调用函数A， 函数A执行到await时， 函数A将当前状态写入WebAssembly Memory， 然后canister执行函数B，B 执行的很快， 即B执行完后函数A的await消息才返回/开始执行。A await的消息返回， A继续执行。

Stable Memory

G Stable Memory

Set maximum number of pages available for library `ExperimentStableMemory.mo` (default 65536).

- 当前Stable Memory分为两部分， 一部分为Experimental StableMemory 可以访问的Memory， 为4G， 另外一部分是在升级时System function以及stable veriables使用的Memory。
- 当前版本的moc不能指定stable内存页，因此被限制在4G的内存范围， 后续可以指定更大的内存页，实现更大文件的存储。

Bucket

- 我们正在积极探索使用Stable Memory的最佳实践， 并积极探索指定数据存储标准的可行性，目前，我们已经实现了使用Stable Memory存储任意大小（4G内）文件的方法， 以及其get方法，我们将持续跟进Stable Memory的进展， 推进统一数据协议的进程。
- Bucket
  - 支持Put / Get 分片
  - 支持Stable Memory存储
  - 支持任意数据字节流
  - 可升级Canister
    - ....