TaiShang FaaS

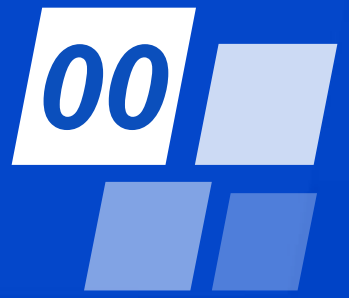
— a decentralized micro FaaS System based on Blockchain

NonceGeek

2022 / 03

- 本项目由「微芒社区」提供支持 -

Catalogue



Why Elixir -- 付费开源好选择

Why Elixir -- 付费开源好选择

Q. 在 Rust, Go 慢慢成为主流的今天是否推荐毕业生尝试 Elixir ？

A. 针对不同的场景，选择最适宜的那个「她」。

Why Elixir -- 付费开源好选择

```elixir
def select(target), do: do_select(target)
defp do_select('to big company'), do:  :no_elixir
defp do_select(target) when target in ['solo work', 'part time job', 'small business'] do
    :elixir
end
```

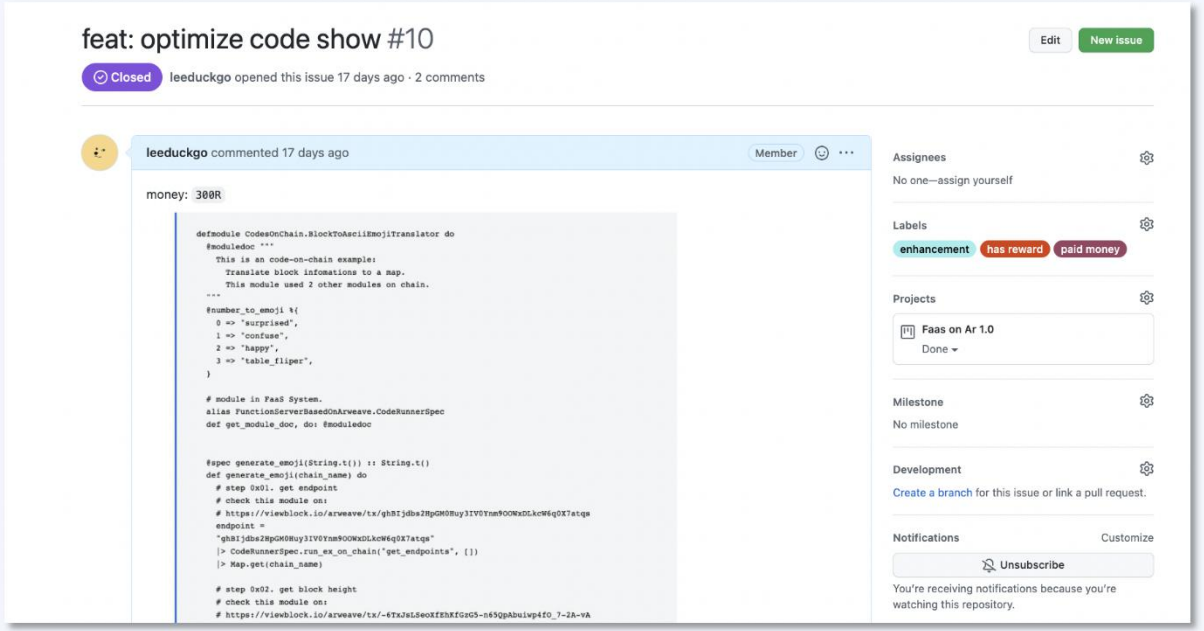
elixir

Why Elixir -- 付费开源好选择

更精准的适配场景：付费开源（Bounty）模式

Java? Java is a work, nobody want to work in rest time.

Rust? Too difficult for freshman.（适合后期）



FaaS 极速简介

FaaS 极速简介

Serverless, MicroService and Faas

View 0x01

如果说微服务是以专注于单一责任与功能的小型功能块为基础，利用模组化的方式组合出复杂的大型应用程序，那么我们还可以进一步认为 Serverless 架构可以提供一种更加“代码碎片化”的软件架构范式，我们称之为 Function as a Services（FaaS）。而所谓的“函数”（Function）提供的是相比微服务更加细小的程序单元。

View 0x02

FaaS 是一种传统云服务器的补充服务形式，很多云服务厂商均有提供此类服务。对于很多 App/dApp 来说，可能仅需要在某些时段一些简单的后端功能，在这种情况下需要租用服务器是一种不经济的行为。在这种情况下，就可以选择租用函数服务，然后采用「按调用次数收费的方式」。

02

康康 Demo

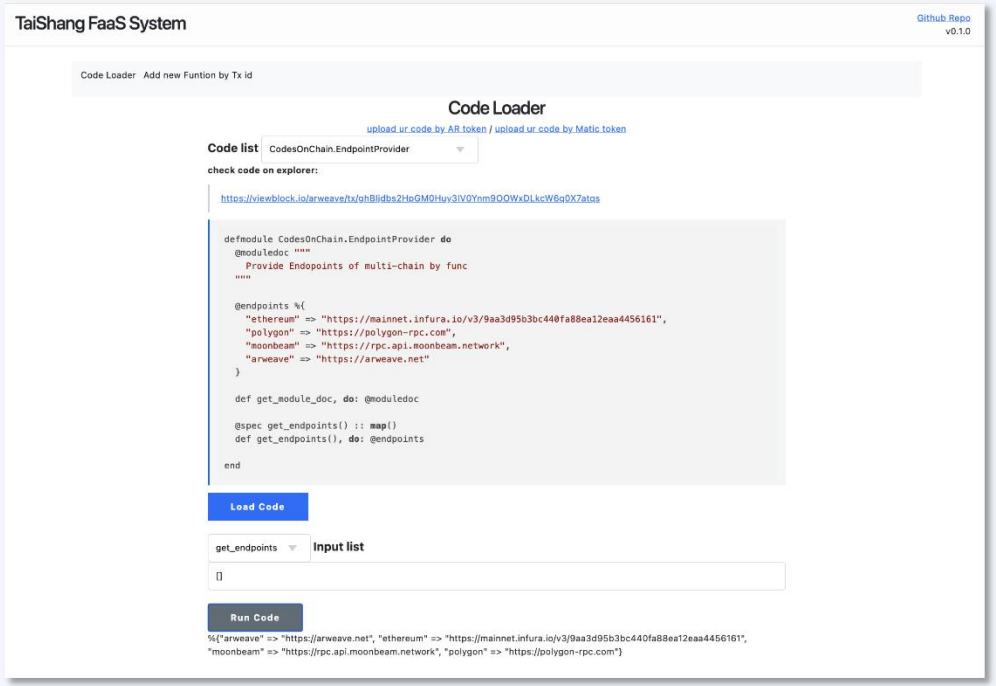
康康 Demo

Repo:

https://github.com/WeLightProject/function_server_based_on_arweave

Demo:

https://faas.noncegeek.com/



区块链对传统 FaaS 局限性的突破

区块链对传统FaaS局限性的突破

函数的封闭性导致难以跨用户使用

传统 FaaS 服务只能由需求方上传函数供自己使用。因为无法确保函数代码不可篡改且透明，且无从记录函数执行过程，因此在没有区块链技术辅助的情况下，出现一个公开的函数市场，供各使用方购买自己需要的函数进行使用，或者进行跨用户的函数组合都是几乎不可能的事情。总而言之，函数封闭不透明的特性极大的限制了 FaaS 的想象空间。

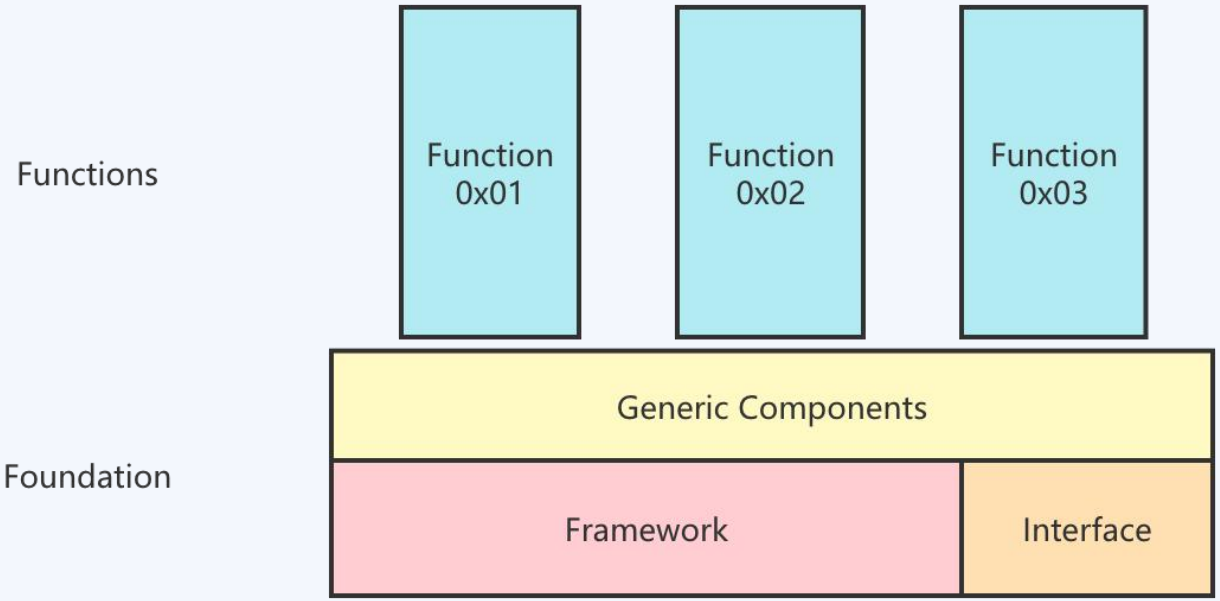
「FaaS 相当于函数积木，但是传统 FaaS 里，玩家要构建什么东西只能用自己的积木」

区块链对传统FaaS局限性的突破

传统 FaaS 是一种中心化闭源服务

闭源 + 中心化限制了 FaaS 服务的想象空间。

使用者无权对底层进行修改。



区块链对传统FaaS局限性的突破

透明、开放且不可篡改的链上函数

所有的代码片段（模块或函数）均存储在 Arweave 链上，在 FaaS 服务运行的时候动态加载到内存中。这些函数都是透明、开放且不可篡改的。因此，一个公开的函数市场与用户间分享自己上传的函数成为了可能。

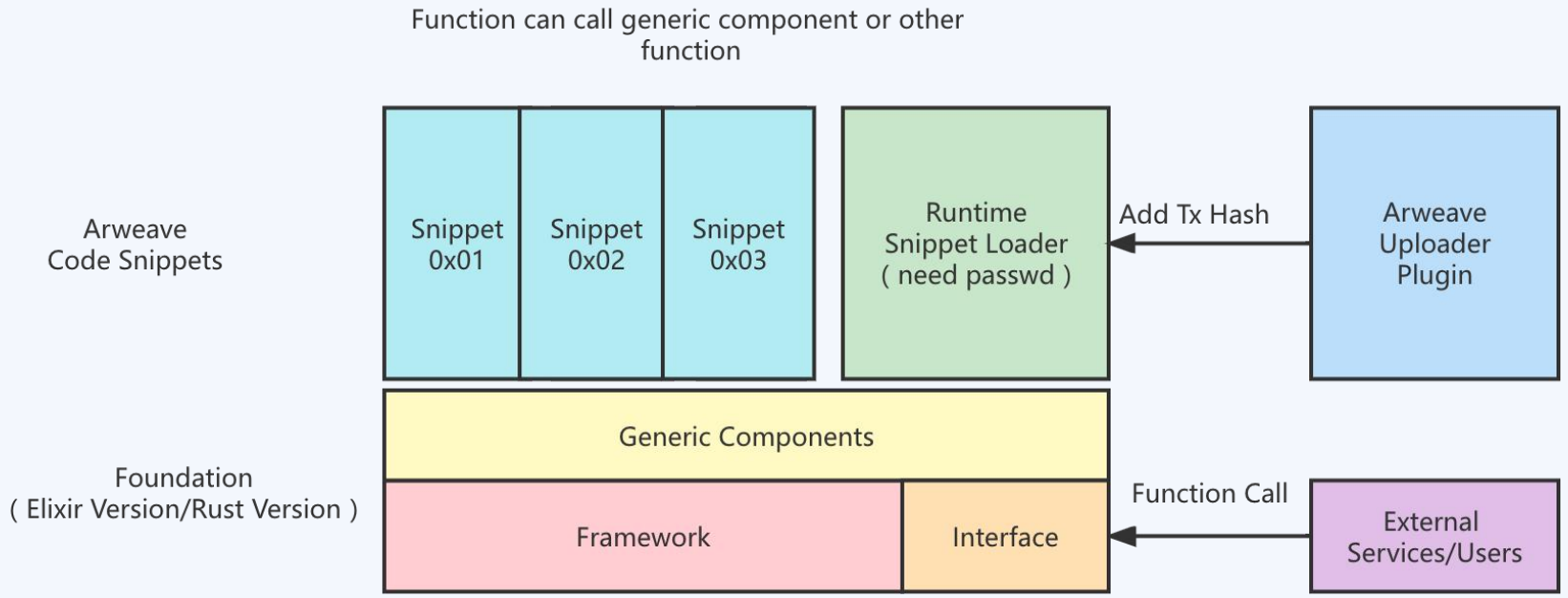
新型 FaaS 里，玩家可以从公开积木市场里挑选积木了！

区块链对传统FaaS局限性的突破

开源的分布式的 FaaS 系统

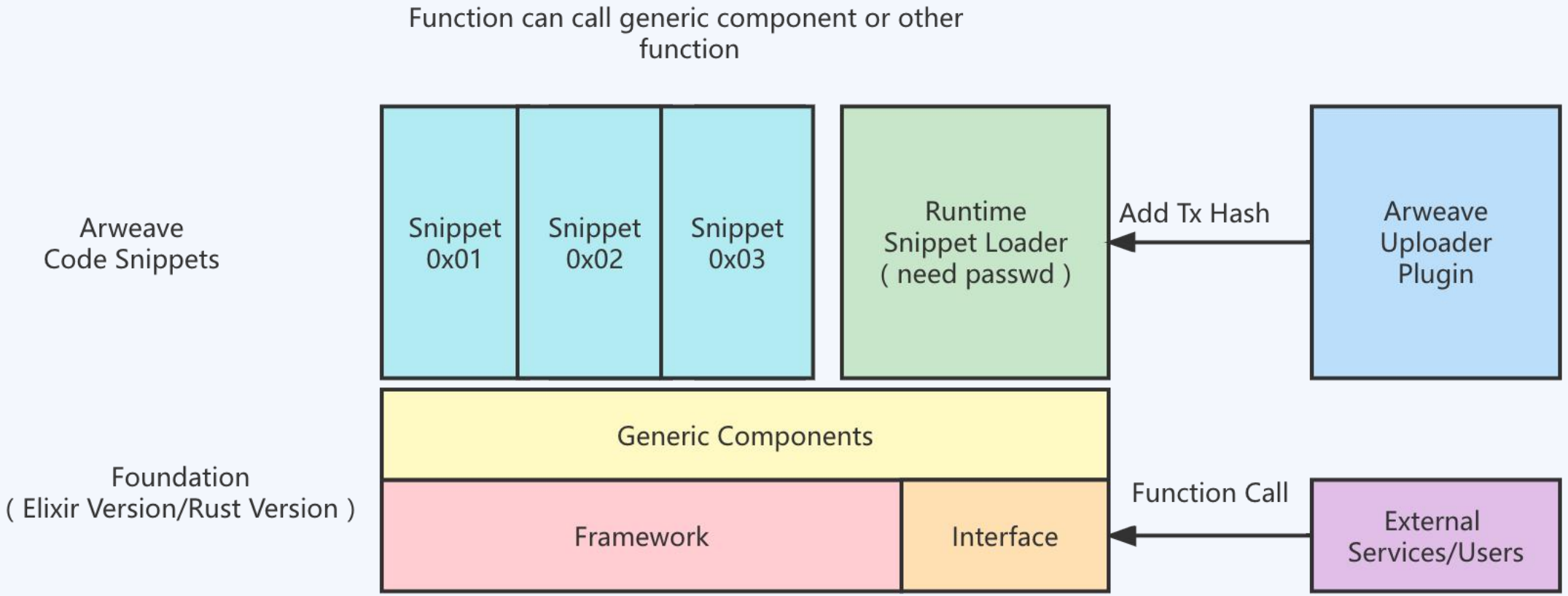
任何人都能低成本地运行起一个自己的小型 FaaS。

所有代码自由修改。



一种新型的 FaaS 系统的设计

一种新型的 FaaS 系统的设计



Why is Elixir Cool for Snippet?

- **清晰易读 —— 代码即文档**
- **模式匹配**
- **恰到好处的 Spec**
- **管道运算符**
- **GenServer?LiveBook?……**

FaaS Demo 源码分享

FaaS Demo 源码分享 —— Snippet 源码

知识点：函数层面模式匹配

```
defmodule CodesOnChain.AsciiDrawer do
    @moduledoc """
    This is an code-on-chain example:
      Draw ascii pic with param
  """

  def get_module_doc, do: @moduledoc

  @spec get_ascii(String.t()) :: String.t()
  def get_ascii("surprised"), do: "( ✧Д✧) OMG!!"
  def get_ascii("confuse"), do: "(๑•﹏•)⋆* ⁑⋆*"
  def get_ascii("happy"), do: " (´ ∀ ` *)"
  def get_ascii("table_fliper"), do: " (ﾉ ͡° ͜ʖ ͡°)ﾉ ┻┻"

end
```

FaaS Demo 源码分享 —— Snippet 源码

知识点：常数

```
defmodule CodesOnChain.EndpointProvider do
  @moduledoc """
    Provide Endopoints of multi-chain by func
  """

  @endpoints %{
    "ethereum" => "https://mainnet.infura.io/v3/9aa3d95b3bc440fa88ea12eaa4456161",
    "polygon" => "https://polygon-rpc.com",
    "moonbeam" => "https://rpc.api.moonbeam.network",
    "arweave" => "https://arweave.net"
  }

  def get_module_doc, do: @moduledoc

  @spec get_endpoints() :: map()
  def get_endpoints(), do: @endpoints

end
```

FaaS Demo 源码分享 —— Snippet 源码

知识点：管道运算符

```elixir
defmodule CodesOnChain.BestBlockHeightGetter do
  @moduledoc """
    This is an code-on-chain example:
      Shows how to get the current block height on dif chain.
  """

  def get_module_doc, do: @moduledoc

  @spec get_best_block_height(String.t(), String.t()) :: integer()
  def get_best_block_height("ethereum", endpoint) do
    {:ok, height} = Ethereumex.HttpClient.eth_block_number(url: endpoint)
    height |> String.slice(2..-1) |> String.to_integer(16)
  end

  def get_best_block_height("arweave", endpoint) do
    ArweaveSdkEx.block_height(endpoint)
  end

end
```

05

新鲜人上手 Ex 的一些建议

新鲜人上手 Ex 的一些建议

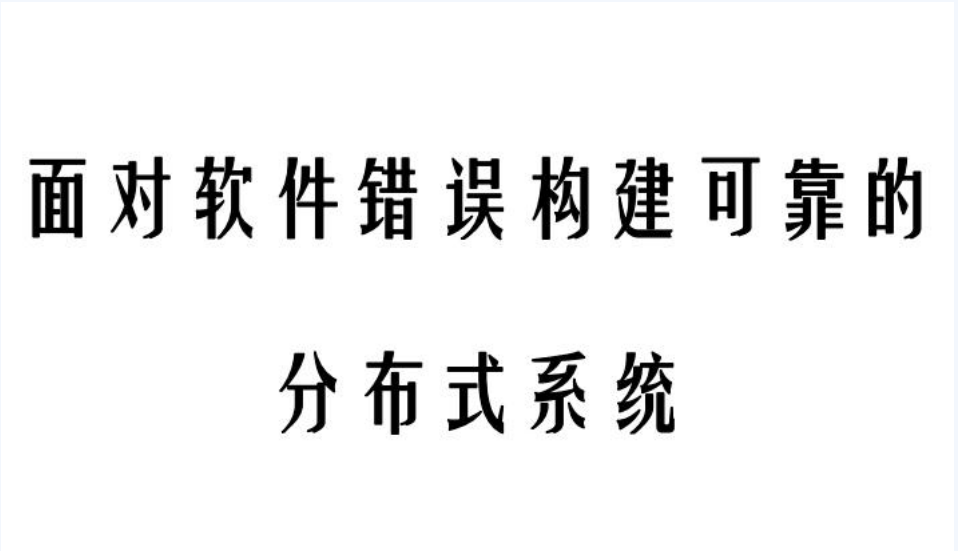
- 从写函数开始
- 了解经典思想

http://www.kaiyuanba.cn/content/other/erlang.pdf

- Play with GenServer
- Play with Phoniex
- Play with Liveview

https://github.com/chrismccord/phoenix_live_view_example

- Play with Livebook



Thanks !

my wechat↓