区块链的市场分布与解决方案选择

====================================================================================

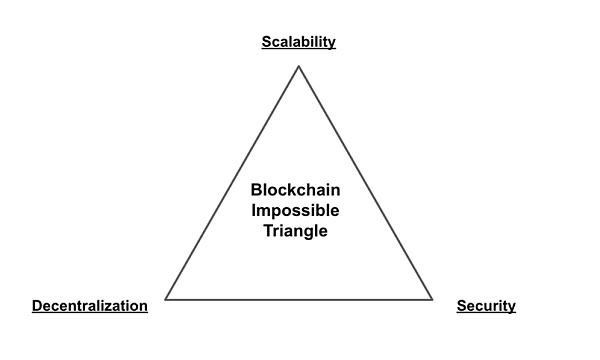
**Solana (SOL)：以太坊、币安链以外的第三大势力崛起**

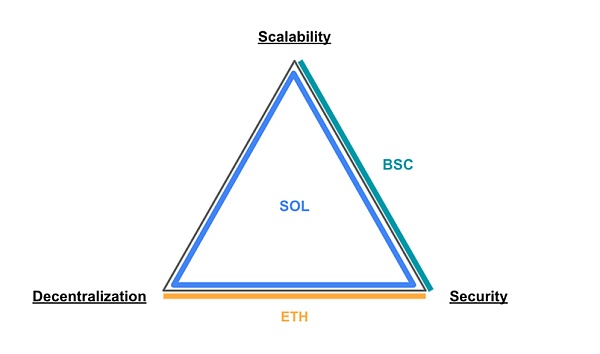
[Solana\_Universe](https://www.jinse.com/member/991540) 2021/07/05 18:22 1.1万

去中心化金融（DeFi）自去年夏季以来在加密货币市场爆发，DeFi 的迅速发展凸显出以太坊（Ethereum）区块链网络技术，无法迅速扩展规模的问题，成为大多数用户使用DeFi 时出现的最大难题。 NFT 加密货币收藏品的热潮，加上 DeFi 市场持续扩大，都加剧上述问题恶化。今年，市场视币安智能链（BSC）为以太坊的代替品。与以太坊相比，BSC 用户可享有较低的 Gas 费、更高速的交易处理速度，比较优势可反映在（BSC 上的）Pancakeswap 与（以太坊上的） Uniswap。撰写本文时，Pancakeswap 总资产锁仓额（TVL）约为81亿美元，而 Uniswap 只有 63.9亿美元。

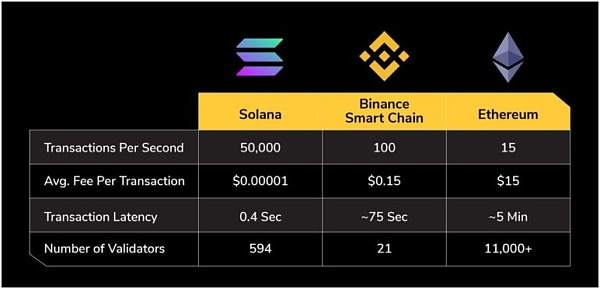
区块链的不可能三角问题

但是，假如了解区块链的真正潜在问题，相信都会发现币安智能链（BSC）只能在速度、交易成本方面，带来一个初步的代替方案，并不能解决区块链的「不可能三角」问题，即安全、可扩展性、去中心化，往往只能牺牲其中一个、去实现另外两个。以太坊就实现去中心化与和安全性；币安智能链就具有可扩展性与安全性 — 仍然不能同时实现三个目标。





这就是 Solana（SOL）在加密货币及区块链领域迅速增长，得到业内机构、尤其区块链开发者支持的原因。解决区块链不可能三角问题，可以透过以下关键指标反映：验证者/网络节点分布、交易延迟时间、每秒可处理交易量（TPS）、以及每宗交易平均费用等。以下图表概述 币安智能链、以太坊、以及 Solana 的分别：



以上数据清晰说明，相对于以太坊及币安智能链，Solana 拥有的技术优势。对于区块链及加密货币项目开发者而言，看重的是 TPS 及验证/网络节点分布。币安智能链有21个网络节点，而当中15个节点是由加密货币交易所币安（Binance）自家控制，这意味着除非增加更多节点，否则就可谓不存在去中心化。以太坊的 TPS 较低、交易费用较高，而无法迅速扩展规模，就是当前面对的问题。而在 Solana 区块链，就似乎可以真正实现了能迅速扩充规模、去中心化网络节点、以及网络安全性的三大指标。

Solana：专为 DeFi 而设的区块链

随着 DeFi 迅速普及，大量 DeFi 应用程式进入 Solana 区块链生态体系，包括 Serum、Raydium、Maps、Bonfida 等。 Solana 势将成为「以太坊杀手」、强大的区块链体系，而上述DeFi 项目代币，包括$SRM、$RAY、$MAPS、$FIDA，以及Solana 网络代币$SOL，近期价格飙升，亦可视为市场价格反映Solana 的潜力

在技术及价格分析以外，支持 Solana 的动力亦来自于其幕后重要人物 Sam Bankman-Fried（SBF）。 Sam Bankman-Fried（SBF），被公认为全球加密货币领域最有影响力的领袖之一，一直是 Solana 的支持者、关键倡导者。他的角色在 Solana 区块链发展过程不可低估，Solana 在业内获得的声誉好评，某程度上 SBF 亦有贡献重大力量。

Solana 生态发展与隐忧

Solana 区块链生态体系急速发展，大量加密货币及 DeFi 创新项目推出，可谓逐步与对手以太坊、币安智能链并驾齐驱。

但是，Solana 区块链有个隐忧：程式语言限制。目前，因为币安智能链兼容以太坊虚拟机（EVM），因此，以太坊上的区块链项目，开发者可以相当简便地将程式码迁移到币安智能链上，并可以继续使用以太坊程式语言Solidity 及相同程式工具继续开发。而相反，Solana 区块链并不兼容以太坊虚拟机 EVM，而 Solana 区块链是用另一程式语言 Rust 编写。因此，与以太坊及币安智能链目前的生态体系相比，用 Rust 开发 Solana 区块链程式的开发者，仍然较少。

总结

在区块链市场的一大主题，肯定是以太坊2.0 升级后，会否得以抢回币安智能链的人气，以及Solana 的迅速崛起能否追上什至超越兼容EVM 类型的区块链（以太坊及币安智能链）。当然，三大区块链各有优势，币安智能链就因为低费用好处、币安与领袖赵长鹏（CZ）支持而增加人气。区块链与加密货币领域，相信将会揭晓哪个区块链体系，将会成为 DeFi 应用的首选平台。

-------------------------------------------------------------------------------------------------------------

3.团队年底释放80%代币。

Solana从以前的项目中得到的一个重要教训是，**定期锁定**并不是token的健康解决方案。因为价格不断反映出社区每月或每6个月对“下一轮token解锁”的预期。

因此Solana修改了自己的**代币经济**，修改过的锁仓结构调整成一次释放，在这种情况下是9个月后释放。除了创始人的代币是在9个月后只释放50%，其余的将在接下来的24个月内解锁。

有太多的项目死于私募不解锁、拖延解锁、解锁的太慢。项目方不敢解锁，怕私募投资者一解锁就砸盘。即便解锁了，私募投资者也可能因为没有流动性而无法出货。二级市场又没有新的买家进入。久而久之便成为了一个僵尸项目。这是一个死循环。我们已经看过了太多教训。

Solana的这个锁仓条例中规中矩吧。只是希望Solana能逃脱这个魔咒。以及不要不信守自己的锁仓条约承诺。毕竟看到了太多临时改变锁仓条例的无耻项目方了。

如果你细心一些，你会发现，今后的新项目，无论是IEO上线还是正常上币方式上线，本质都是IEO的内核：极小的流通盘。上线运气好的话拉盘一波，大多数则是拉都不拉直接下跌。Solana和IEO项目在形式上的区别，只是把用户持有平台币打新这一环节替换成了荷兰拍卖。其余的没有任何区别。

很多人抱怨新项目没有机会投资，但情况也没有那么糟糕。诸如Bloackstack，Orogin，Solana，Celo这些项目，也还是有荷兰拍卖的机会。

荷兰拍卖其实挺好的。拍卖能在一定程度上解决交易条件的问题。用市场决定代币的价格。投资者只有在拍卖降到了合理的价格时才会参与代币融资，是主动选择而非被动接受项目方的定价。

比如这一次Solana的荷兰拍卖，如果你勇气可嘉参加了，那么便获得了400%的收益，而且是参加完拍卖一周后就上线币安。所以谁说新项目没有参与机会的呢。

Solana是目前性能最快的公链，关于这点应该不会有异议。

但真正需要像 Solana 这样的解决方案的项目可能并不多。我们都知道区块链的用户采用率总体上相对较低，说实话大多数项目都不需要像Solana那样的性能。Solana关注的是一小部分项目实际上已经开始被大量采用的项目，同时这些项目需要扩展性方案，Solana则帮助他们定制合适的解决方案。

上述就是聚链网小编对SOL币是什么币这个问题的介绍，最后聚链网小编提醒投资者，在选择数字货币投资之前，尤其是新发行的数字货币，投资者一定要点进去看网站的设计风格，是比较丑的，还是比较美的，没关系，投资者哪怕没学过设计没关系，根据自己的主观判断就好，根据自己的标准给这些项目打分，因为但凡是认真做事的项目方，必然会去找靠谱的设计者设计网站，如果网站都不好好设计，可能骗局的可能性要更大一些。

【本文为聚链网所属，网页转载须在文首注明来源聚链网及作者名字。微信转载须在文章评论区联系授权。】

----------------------------------------------------------------------------------------------------------------------------

你可以把NFT理解为一种存储在区块链上的数字收据，代表着从艺术品、音乐甚至门票销售到演唱会的任何东西的所有权。

制作一只数字猿的费用是6个SOL，但随着该项目的售罄，Solanart上一件收藏品的底价目前是14.40个SOL。

-----------------------------------------------------------------------------------------------------------------------------------

**SOL涨了50倍，是什么让它成为区块链新贵？**

刘一非 中兴通讯股份有限公司 高级系统架构师

最近这几周区块链最爆点的话题并不是比特币暴跌1万美金然后又反弹到48000美元，我感觉比特币这次表演已经是夕阳前的蹦迪了。

随着区块链技术的不断翻新，凝聚共识为核心的第一代币，包括比特币、LTC、Doge币，无论共识有多么响，终究是带不来新的价值增加。

而第二代币以太坊为代表撑起了Defi金融生态链，大大提速了区块链与我们实际生活的对接与价值。



3个月就飞上天了

比以太坊的速度还要快1000倍的SOL，我不知道是否可以算第三代的[区块链](https://www.zhihu.com/search?q=%E5%8C%BA%E5%9D%97%E9%93%BE&search_source=Entity&hybrid_search_source=Entity&hybrid_search_extra=%7B%22sourceType%22%3A%22article%22%2C%22sourceId%22%3A411548004%7D)

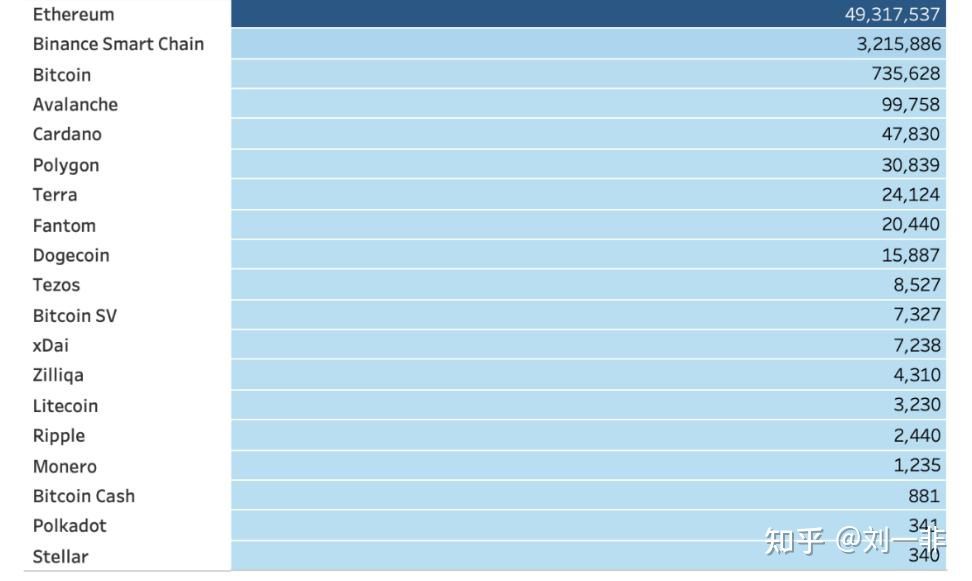
技术，以NFT为爆发点，即将迎来新的机遇和起飞！

**SOL的暴发原因**

目前的以太坊，仍然是区块链的第一大生态，但是，一方面业务越来越多，产生了巨大的通道堵塞，另一方面以太坊价格高昂，导致所谓的Gas燃气费不断提升，很多中小型创业者已经花不起这个费用了，不得不将目光转向别处。

2020年，以太坊的峰值处理交易速度为5次/秒，如果映射到大型的股票交易所，是非常不适用的。

同时，由于近期各类NFT交易和金融交易的频繁，以太坊日均交易Gase费用达到了4931万美元，比后面的所有区块链交易费用还要高出10倍。



以太坊太贵

正是因为以太坊技术的陈旧导致费用的高涨，目前依托于比特币的NFT作品，一次的铸造费就接近1000元，这也让那些有志于通过Jpg赚钱的创造者们先思考一下，如果付出这么高的铸造费，是否能够收回成本。

**正是在这种情况下，无论交易费用和交易速度都领先的SOL应运而生。**

目前Solana区块链的出块时间400毫秒，交易速度可以达到5000次/秒，比以太坊要快100倍。

**SOL的8大核心技术**

那么是哪些技术导致SOL有这么快的速度呢，其核心在于8项区块链创新技术：

1. Proof-Of-History：历史证明；
2. Gulf Stream:高尔夫流；
3. Turbine：
4. Sealeve:海平面；
5. Cloudbreak：云间隔；
6. Tower Consensus:
7. Archivers：
8. Pipeling：流水线。

基于区块链的技术，不同的区块发生交易，最后会同步到每一个关键节点上，这个同步的过程非常消耗时间和资源。

**Sol采用了一个名叫历史证明(PoH）的过程，**通过递归的计算哈希函数将时间证明分布到多个节点上进行。

**最终新的区块能够增加，并不需要相互交流进行验证！**

**反正凭这个，速度能比以太坊快1000倍！**

**子弹一样快**

当然，并不仅仅只有这个技术，各自分别证明，证明错了怎么办，这就说一下其它几个技术。

Gulf Stream：这个技术在服务端加了一个缓存，如果各个单点的证明在超时的阶段没有确认，那么客户端就认为该交易无效，不会造成网络未确认的业务。

Turbine就是SOL区块链使用的一种特殊区块传播技术，该技术将区块划分为数据包，再散布到随机的peer节点，这些更上的数据包，可以更好地处理网络延时。

Sealeve:海平面技术，使用了一个并行的交易处理引擎，可以增加网络吞量，提前预期每个节点需要读写的数据量，加快了资源的预订和定向。

Cloudbreak是一种定制设计状态架构，该架构针对SSD的RAID 0的并发读取和写入进行了优化，进一步减少了交易的延时。

Tower Consensus是一种算法，基于POS的选举会提前选择一个节点作为master，其SOL代币越多，被 选中越可能，通过这个技术，区块链的出块时间只需要400毫秒。

**那你知道比特币的出块时间是多少吗？**

比特币的出块平均时间是10分钟，但是在2019年，由于网络延时的原因，对区块高度为597262的区块进行挖矿消耗了1个小时，对于597273的出块也消耗了1个小时。

**区块链的神奇技术**

**比特币这个设计根本就不是应对大规模金融交易的，要不是以太坊及时出来了，帮它扛住了大量的交易消耗，比特币可能活不到今天！**

正因为SOL的技术如此强大而有效，所以吸引了大量的开发者的注意，也撑起了巨大的生态系统。

**Solana生态系统的那些大拿们**

说到SOL生态，首当其冲的就是最近的BAYC项目，无聊猿猴俱乐部，当前最火热的NFT项目，原型是在SOL链上1万只猿猴头像。



库里这个炒作大师

库里在8月共18万美元购买了这个头像，如果按1万只猿猴计算，单这个项目就高达18亿美元之多。

而实际上，SOL早就在这个领域默默布局，扶植了超多的生态项目，建立起了一个仅次于以太坊的公链帝国。



SOL生态建设

SOL项目2017年发布白皮书，在不到5年的时间中，它先后在多个领域发力，包括Defi金融领域、工具领域 、钱包领域、游戏领域，NFT领域等多个领域中具备着给力的应用和推广方向，而且不少项目也开始有了Token证书，以便向更深的领域扩展。

这就叫桃李不言,下自成蹊！

做得好，自然就有人上来用这个平台，不过随着SOL的估值上新高，原来的这些开发应用的费用也要水涨船高了。

**那些有趣的SOL项目们**

我也列举几个有通证的SOL项目吧。

Kreechures项目-通证（KIN）：每12分钟可以**生成**一个NFT的宠物Kreechures，可以通过训练和游戏竞技升级，升级可赚取KIN，KIN可以用来支付游戏费用。

**星舰游戏**

Star Altas，基于SOL的星际争霸游戏，正准备上线。



SOL上的星际争霸

Media Network：一个去中心化的媒体分发平台，尚未IDO。

Audius：音乐平台，号称用户100万。

Sollet：SOL的原生钱包，可以消耗SOL获得一些新应用；

Hedgehog：预测市场，相比于中心化预测市场，提供一个去中心化的博彩平台；

Serum DEX：交易市场，是核心DEX，提供4种可选交易功能，日交易4500万美元。

。。。

目前，在Sol公链上的质押金额已经突破100亿美元，仅次于以太坊和币安链，而且还在快速成长着。

**未来已来**

每当了解到区块链这些最新的信息，我不由地感觉到第三次互联网的浪潮要来了。

第一次的浪潮是微软、网景们，开始基础架构的建设；

第二次是亚马逊、腾讯、FB们，开始建构应用层的基础设施；

而区块链来了，突然得到了互联网前所未有的投资，更多的技术和颠覆性商业逻辑出来了，从这个Sol的发展中，我可以感受到美国的创造力和革新性。

去中心化的创新，不再像过去那样容易受到谷歌、微软等的资金干扰和技术压力 ，因为这些老牌科技公司太老旧了，没法轻快地掉头，更容易被政策阻碍。

**老旧的windows照样赚钱**

依靠巨额的垄断收入，这些科技巨头们照样能畅快地赚钱。

但是，区块链是给年轻人的新机会，只要你技术够好，能够跟上可信的技术链，就能通过去中心化的体系，很快地实现新的致富机会，无论是Defi、还是NFT，都代表了人们的一种良好的探索，它们可能是未来的方向，也可能是泡沫。

**但对于年轻人来说，这毕竟是一个新的机会。**

**每年几万亿的资金流向这里，最终不是白给的！**

=====================================================================================

1.元宇宙第一阶段

我们目前正处于现代VR技术的第一阶段，只需一个头盔和两个手控器就能在虚拟世界中获得相对沉浸的体验。然而，今天的VR只能影响我们的视觉、听觉和有限的触摸形式。它只有大约50%的时间可以实现完全的沉浸感。

**Web3.0的基础设施层**已经奠定，不同的区块链协议正在被元宇宙游戏使用，包括**以太坊、索拉纳、波场、卡尔达诺、Polygon链和BSC**。NFT的出现使人们可以拥有数字资产。数字土地、太空船、衣服、游戏中的货币、皮肤、艺术品现在可以由玩家拥有，而不是由游戏开发者拥有/控制。那真是个破坏性大的新闻，如果你现在还不认识到，你就不会错失良机。事实上，你错过了有史以来最大的一艘游轮!玩转游戏也在巩固其商业模式，并将完全重塑游戏。

元宇宙和Web3.0仍处于起步阶段，游戏质量没有达到3A大作，因此很难与成熟的3A游戏开发商竞争。

=============================================================================

**去中心化交易所产品（IDO**）是一种在去中心化交易所（**DEX**）上发行代币而无需中介的方式。 虽然传统的加密货币使用许多互连来参与去中心化金融，但像 IDO 这样的解决方案允许更多人参与。

**如何启动 IDO？**

我们来谈谈如何开发 DEX 报价。 我们的工作方式如下：

项目通过启动平台或专门从事 IDO 的平台接触人们和社区。 像你我这样的投资者可以在特定条件下在代币发行之前购买。 IDO 的代币在去中心化交易所 DEX 上发布。

一般销售有两种不同的应用程序：

1. 代币的发行者以不同的价格水平限制销售。
2. 代币的发行者创建一个拍卖，其中售价由供需决定。

代币上市是在基于自动做市商 (AMM) 的 DEX 上完成的，例如 Uniswap 或 Balancer。 发行人出售收入并创造矿池流动性，再次使用代币为代币创建流动性二级市场。 流动性激励代表代币发行者可以用来推出产品或服务的各种程序。

流动性挖矿（冻结项目中的硬币以提供流动性）也是一种非常流行的做法。 它使人们能够支持该平台。

**如何测试项目？**

正如我之前提到的，许多新工具都可能成为诈骗计划的一部分。 尽管如今去中心化金融中的筹款形式比过去得到了更好的保护，但了解一些细节以保护自己免受欺诈开发者的侵害还是很有用的。 **在 IDO 上轻松赚钱的最流行方法之一是发布虚假合约地址。 它实际上是控制令牌逻辑的实际合约的地址。 有时，IDO 发行的每一种新的加密货币也会发行 3-4 种同名的假加密货币。 因此，检查合约地址非常重要**。

有一种简单的方法可以做到这一点：在社交媒体、信使和网站上查看。 不同的地址意味着欺诈。

**IDO的未来**

在 DEX 模型中，发起筹款计划不需要许可，IDO 为该交易提供了有效的设置。 目前，只有 DeFi项目可以通过 IDO 组织筹款活动。 未来，我认为来自加密货币系统其他领域的项目也可以通过这种方式筹集资金。

同时，如果你想知道 2021 年哪家 DeFi 初创公司值得研究，我建议你阅读这篇文章。

对于初创公司筹集资金来说，IDO 是一个更具吸引力的选择。 这样的车型才有机会在市场上占据重要的位置。 具有管理机制的 IDO 平台也可以增加 DEX 中主要产品的受欢迎程度。

=============================================================================

=====================================================================================

Solana solidity到solana 开发第一讲

Neo老师 2021-05-15 15:37:42 2617 收藏 11 分类专栏： solana 文章标签： 区块链

说明：本文主要是讲解Solana的一些基本概念，对于详细的编程还有更多细节需要了解和学习，往后的课程可以给大家讲解更多

solana 简介

solana 是一个高性能的，无许可的区块链，其中共识算法采用**POH**（历史证明），POH很好的解决了单个可信赖时间源的分布式网络中的时间问题，就是通过时间搓来验证真实性，POH算法是采用BTC使用的Has256算法在本地生成时间戳，Solana的高频交易可以让金融的高频交易在区块链中成为可能。

Solana的数据排序结构按时间顺序在区块链上链接消息。这提供了有关区块链中唯一时间以及历史记录中每个消息时间的相对验证的加密证明。

solana 账户模型

作为一个开发者，最先要了解的就是账户模型

以太坊账户模型

大家都知道以太坊账户模型是分两种，一种是主网币，一种是合约币，主网币里面的的account 就有一个balance字断，用来记录用户的ETH 的数量，如果某个地址拥有其他的token ，是在合约里面给这个地址存了个mapping，用来记录这个用户token 的数量。因此用户拥有的ERC20 token其实记账是记录在合约里面，以太坊合约具有存储功能，是因为合约用一个世纪状态永远存储在区块里面。

**solana 账户模型**

solana 的账户模型不一样的地方在于，solana 账户除了可以存储本币sol以外，还增加了一个data字段，这个data字段

具体可以参考[(https://github.com/jstarry/solana-wiki/blob/main/content/zh-cn/docs/Account%20Model/\_index.md)]

先看一下账户的字段

字段 描述

lamports The number of lamports owned by this account.

owner The program owner of this account.

executable The program owner of this account.

data The raw data byte array stored by this account.

rent\_epoch The next epoch that this account will owe rent.

详细描述一下

lamports ：这个就是和以太坊的balance一样，描述这个账号有多少个SOL 可以使用

owner：owner这个字段保证了用户对账户的决定拥有权利，如果某一个地址是某一个account的owner，那么那个地址就可以对这个account资产进行转移。所以solana的合约token其实就是solana程序会帮忙生成一个account，然后把owner赋予给你。

executable，这个表示用户的状态是不是可执行的状态。在solana的账户体系里面的data字断是表示存储，或者可执行。可执行的data是帐户数据专门用于不可变字节代码该字节代码用于处理交易。

data：这个和executable 描述相似

rent\_epoch：这个比较有意思，就是solana的合约和solidity合约不一样的地方，solana的合约部署是要按大小定期收费的，目前是费用是2天一收，应该是0.01SOL 1K的大小。如果没有费用solana系统就会清除这些账号。

solana 合约介绍

由于账户模型的不一样，那么刚开始理解solana的合约就很困难，对于solidity的开发者最想知道的几个问题

1、solana 合约开发环境

2、solana 合约开发语言

3、solana 合约部署

4、solana 合约逻辑

5、solana 合约前端后端交互。

针对以上问题会从几个方面介绍一下。这几天研究的心得，感谢Solana 开发大神 Justin的解答我在学习过程中的疑惑。

solana 合约是什么以及和solidity 的区别。

先聊一下solana的合约，**solana的合约**其实就是一段程序，这个程序没有存储功能，程序编译好以后，是一个BPF字节码，这个和solidity 编译后是一段二进制码是一样的。solidity是通过合约地址找到合约存在得状态数，然后再通过input参数，拿到需要执行得方法和参数，再执行对于得合约方法。其实solana 这个部分是一样得，solana 发布到连上得合约有一个programID，这个programID也是一个Hash256，和solidity合约地址一样，solana 合约得方法通过一个枚举来定义，也就是，方法分别是0-1-2-3-4，参数就是解析data，那么solana 合约最难得就是要解析这个data，因为这个data是用户自己定义和排序的，solana不提供标准的map，struct，这些数据结构存储， 都是以文本得形式存储在各个account里面得data里面，所以solana合约最困难得就是要对data进行编码，解析再解码，你可以想象一下一个token合约，solidity 最多两个map，一个是balance，一个是allowance，就够了，但是solana不一样，他没有存储，他需要为每一个人创建一个临时得account，然后解析出来，给他们做tranfer或者approve操作。还要有一堆的安全性判断。最重要得是你要得存储进行排版，解析。

solana 合约开发语言

solana 目前支持 **rust** 和C 对合约进行开发，但是建议大家学习Rust，因为整个链都是rust的开发的。整个接口或者其他的框架也大部分是rust，这样可利用的开发工具就会更多。

初步学习的人，建议大家学习完helloworld以后可以看看 token的代码，合约，还有web3j调用都有。

helloworld 链接[solana Helloworld]

token 合约代码链接[solana Token]

solana 的安装

solana需要安装一些环境，可以参考官方文档，下面我也给了链接，大家可以直接用

环境安装

[Install the Solana Tool Suite]

[Using Solana CLI]

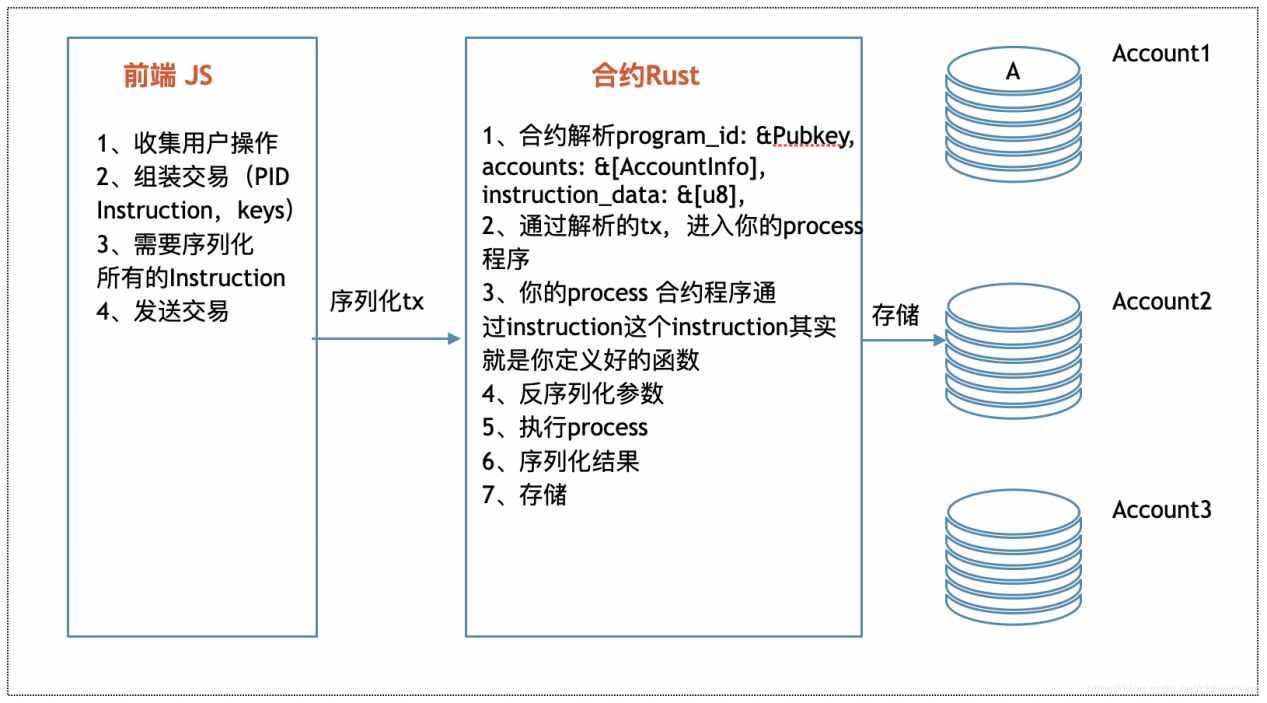
anhore安装

anhore 是一个开发工具，类似solidity 的 truffle 和 hardhat

[anhore 官方介绍]

solana 合约框架

先整理清楚solana 合约是怎么运行的。



我们先来看看token 合约，为什么不讲helloworld，Helloworld 太简单，很多东西都没有，我们看看token 合约

token 合约路径：[solana Token]

token 合约目录结构

–cli

–js

---- cli

---- client

--------token.js

–program

----src

------entrypoint.rs // 这个整个程序的入口，也就是solana节点会把交易里面的参数都传入到这个接口

------error.rs // 错误信息

------instruction.rs // 这个才是序列化/反序列化的/结构体的定义问题

------lib.rs //这个是rust 用来串联本目录或者其他目录文件用的

------native\_mint.rs // 这个是mint 方法

------processor.rs //这个是token 的所有方法调用的地方，所有方法实现的地方，进入到这些方法，需要传入的参数包括（函数，参数 都要解析出来）

------state.rs // 这个是状态

如果大家使用过Go 写solidity 程序，那么再来理解solana 合约程序那么就很好理解了。go 语言版本的solidity是需要运行在主网代码上，所以有几个要点：

1、哪个程序

2、哪个函数

3、实现什么功能

4、结果在哪里

我们就用token 的tranfer 方法来对整个问题进行解答。

1、哪个程序

前面也介绍了solana 的智能合约的指向是一个programID，programID 就和solidity合约的合约地址是一个意思，所以你发送交易的时候要指定programID。

2、哪个函数

我们的合约可能有很多方法，那么我一笔交易，指定了programID，我还需要指定是哪个函数，以及对应的参数，所以再processor方法里面，我们看到一个match 用来分发函数，里面的第一个就是获取用户输入的调用的哪个方法。

pub fn process(program\_id: &Pubkey, accounts: &[AccountInfo], input: &[u8]) -> ProgramResult {undefined

let instruction = TokenInstruction::unpack(input)?;

同样，我们也可以通过看token.js方法里面的交易组装函数（createTransferCheckedInstruction）,就有一个instruction ：12这样一个赋值，所以这个12就是对应的函数，那么这个定义一般我们要怎么定义呢？

dataLayout.encode(

{undefined

instruction: 12, // TransferChecked instruction

amount: new u64(amount).toBuffer(),

decimals,

},

data,

);

我们可以通过看instruction.rs文件，里面有定一个enum,里面定义了每个函数的instruction，也就是说，前端需要知道有哪些函数，这个就和ABI一样，但是更简单。复杂的事数据的处理,我们看看12是哪个函数：TransferChecked 这个函数。

TransferChecked {undefined

/// The amount of tokens to transfer.

amount: u64,

/// Expected number of base 10 digits to the right of the decimal place.

decimals: u8,

},

3 、这个函数做什么事情

从 pub fn process(program\_id: &Pubkey, accounts: &[AccountInfo], input: &[u8]) -> ProgramResult 这个函数里面找到12 这个instruction对应的TransferChecked 然后去分发进入的是这个 【process\_transfer】

TokenInstruction::TransferChecked { amount, decimals } => {undefined

msg!(“Instruction: TransferChecked”);

Self::process\_transfer(program\_id, accounts, amount, Some(decimals))

}

那么process\_transfer这个函数做了什么事情呢？

1、账户状态的准备检查

2、账户转账处理

3、账户信息打包（pack 函数）

4、 怎么存储

由于整个rust 都是用指针对内存文件进行读写操作，所有在pack 结束后就进行了存储。

solana 合约交易

对于前端而言，和solidity有什么区别呢？

solidity前端只需要使用ABI 对input data 进行bytecode 就可以了，然后进行组装交易，发送签名交易。就可以完成合约的调用，使用的是web3j的库。

那么solana 的前端调用合约交易如何组装呢？

这个是交易的参数，很明确。就不一一解释了

static createTransferInstruction(

programId: PublicKey,

source: PublicKey,

destination: PublicKey,

owner: PublicKey,

multiSigners: Array,

amount: number | u64,

输出参数的定义

const dataLayout = BufferLayout.struct([

BufferLayout.u8(‘instruction’),

Layout.uint64(‘amount’),

]);

调用合约函数的绑定

const data = Buffer.alloc(dataLayout.span);

dataLayout.encode(

{undefined

instruction: 3, // Transfer instruction

amount: new u64(amount).toBuffer(),

},

data,

);

from /to 压缩到keys 里面

let keys = [

{pubkey: source, isSigner: false, isWritable: true},

{pubkey: destination, isSigner: false, isWritable: true},

];

if (multiSigners.length === 0) {undefined

keys.push({undefined

pubkey: owner,

isSigner: true,

isWritable: false,

});

} else {undefined

keys.push({pubkey: owner, isSigner: false, isWritable: false});

multiSigners.forEach(signer =>

keys.push({undefined

pubkey: signer.publicKey,

isSigner: true,

isWritable: false,

}),

);

}

组装好的交易返回出去

return new TransactionInstruction({undefined

keys,

programId: programId,

data,

});

所以总的来说 交易发送也是对交易的组装，对input的组装，和solidity 几乎一样。

======================================================================================

