



AlphaWallet区块链门票测试报告

2018俄罗斯世界杯ERC875门票

AlphaWallet

二零一八年八月

作者简介

AlphaWallet 张中南

AlphaWallet(Stormbird Pte. Ltd.)联合创始人兼CEO。中国籍澳大利亚居民。超过10年的票务市场商务经验，在亚太各国（中国，澳大利亚，新加坡，日本，韩国，马来西亚）管理跨国团队和企业超过7年。成功帮助360Experience（Ticketbis 3 个业务部门之一）进入亚太市场，从零开始到2016年被eBay以1.65亿美金整体（Ticketbis）收购。连续创业者，在澳大利亚，中国北京，中国香港和新加坡创办过5个公司。技术爱好者，2014年开始接触区块链，曾尝试用区块链技术重构热门活动门票市场，持有少量比特币和以太币。咏春和拳击爱好者。

SHANKAI 郑来

现任盛开体育副总裁，2010/2014/2018 三届世界杯官方款待计划中国区负责人，曾经就职中国中旅集团负责公民出境业务，中信集团国际金融控股集团有限公司负责旅游，航空和体育板块的投资业务。

SHANKAI 高东东

盛开体育IT经理。曾负责运营多场国际大赛票务管理（2014-2018世界杯、2016年欧洲杯、欧冠、巴萨俱乐部等）。2016年接触区块链，并从2017年开始与 AlphaWallet(Stormbird Pte. Ltd.)紧密合作，以国际顶级赛事门票为基础尝试新型互联网技术，一直致力于通过票务通道打开国际赛事的中国大市场，推动中国体育事业的快速发展。

目录

背景介绍	1
ALPHAWALLET	1
盛开体育	1
一级市场	1
二级市场	1
区块链技术	2
当前票务市场的问题和区块链能带来的变化	3
二级市场及票务造假	3
支付欺诈	6
活动安全KYC	6
监管下的自由流通	7
2018俄罗斯世界杯ERC 875门票测试结果	9
技术方案	9
测试内容	13
测试目的和结果	13

背景介绍

ALPHAWALLET

AlphaWallet (Stormbird Pte. Ltd.) 是一家区块链创业公司，主要专注在Layer 2, Offchain的区块链协议开发，还有消费者终端应用平台的开发，从应用的角度入手提升区块链的可用性，性能和隐私。AlphaWallet手机应用是面向普通消费者的以太坊智能合约调用工具以及协议运行平台。ERC 875是服务于真实商用案例的不可替代性通证（non-fungible token）标准，开发人员和企业可以非常容易的用ERC 875 Token来指代物理或数字世界内的人/事/物/权，并实现高效的原子化交易。

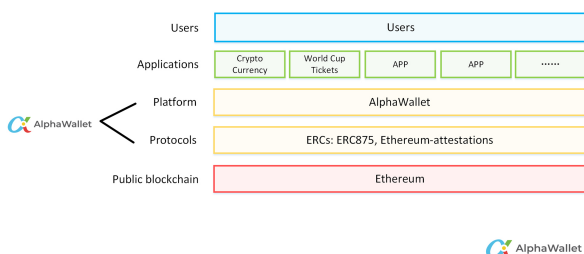


图1 AlphaWallet平台结构

盛开体育

北京盛开国际旅行社有限公司，是一家快速向前发展的年轻体育营销公司，为国内外的投资者提供体育商业运营方案。我们对于本土市场有着深入的了解掌握，并有着特殊的国内、外国际人脉网络关系。我们拥有丰富的从事大型国际体育赛事的经验，充分了解国际体育赛事参与者的需求。充分利用体育

作为交流平台帮助中国品牌不断成长，我们将世界顶级赛事引进中国并通过数字媒体技术让中国的体育爱好者可以近距离接触到更多的精彩体育项目。

一级市场

活动主办方监管下的官方授权的渠道组成的市场如主办方自己的官方销售网站，大麦，永乐等官方授权票务平台和其他官方授权销售机构。

二级市场

非主办方官方授权的销售渠道组成的市场包括个人专业票贩子，P2P市场平台等。

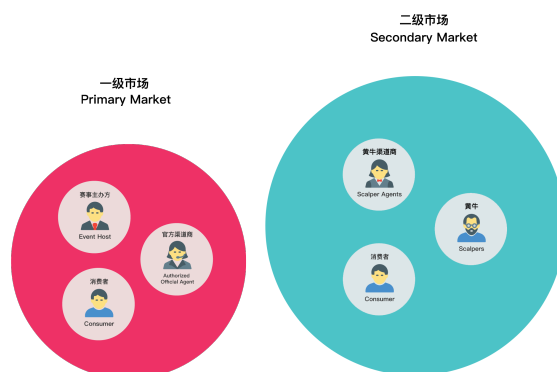


图2 一级市场和二级市场划分

区块链技术

区块链是一个分布式账本技术，能够保证“账本”内数据所代表的信息不容易被篡改，进而实现相应信息代表的事物所有权的唯一性。同时公有链因为是一个公开账本，没有准入门槛，所以能实现所有权的点对点自由流通。随着以太坊的兴起，在“记账”的同时还能支持智能合约实现各种流通逻辑。

现阶段区块链技术还处在非常早期阶段，在这个阶段，除实验性质的项目外，真正适合落地商业化的应用场景非常少。现阶段适合放上区块链的数据需要满足以下标准：

1. **数据所代表的信息是有高价值的**。如钱，重要的权益等等。
2. **数据所代表的信息在使用（体现价值）的时候需要被证明**。如使用支付宝支付商家的时候，需要支付宝/银行/清算系统来证明“资金”的流转。
3. **相应信息所代表的事物所有权，有流通转让变更的使用场景**。如各种兑换券等。

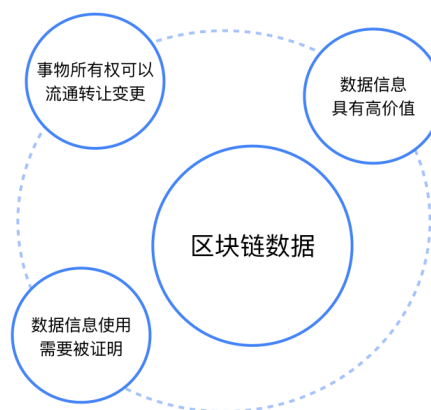


图3 区块链数据需要满足的标准

现阶段，如果数据不能同时满足1、2、3，那么有比区块链好很多的其他技术来满足相应需求。最典型的反例如身份信息，符合1和2，但是基本不存在转让所有权的使用场景，使用现有的成熟的认证（Attestation）技术可以更好的满足使用需求。又例如各种存证类应用等，区块链是“账本”不是数据库，不是用来存储各种所有权不需要转让的“证据”的。

当前票务市场的问题和区块链能带来的变化

二级市场及票务造假

二级市场一直以来都是让各方既爱又恨的存在，在大部分热门项目上，二级市场无论参与人数还是资金规模都比一级市场大很多。

受影响方	存在的问题
主办方	很难从二级市场获得应得的好处 难以获取市场数据但仍需要支付高额手续费 二级市场负面信息影响主办方品牌和声誉
消费者	二级市场无监管（大量黄牛票、假票，以及存在囤积行为） 导致消费者利益无法得到保障
官方授权的销售渠道	黄牛抢购囤积门票对官方授权销售渠道造成负面影响
二级市场销售渠道	缺乏官方授权，信任缺失 回款周期长 支付高额手续费给提供担保的市场（担保市场需要承担巨额风险）

表1 当前票务市场存在的问题

当前票务市场的问题

对于主办方 二级市场加大了活动的传播范围是很好。但同时因为二级市场完全不受主办方的监管，所以主办方没有办法更进一步的从二级市场拿到应得的好处。没有办法拿钱，也没有办法向赞助商证明二级市场的传播规模到底有多大。一级市场经常被大型渠道商垄断，主办方拿不到数据还要支付高额手续费。各种二级市场负面信息，如假票，天价票等等影响主办方品牌和声誉。

对于消费者 二级市场的存在满足了部分消费者的需求，同时因为二级市场完全无监管，消费者的利益没有办法得到保障，各种天价票和假票充斥着二级市场，各种囤积行为导致消费者买不到票，场地内却还有空座。

对于官方授权的销售渠道 二级市场很多时候是他们处理库存的必须渠道，但同时对于热门活动，大量黄牛抢购囤积门票，也对官方授权的销售渠道造成非常不好的影响。

对于二级市场销售渠道 二级市场销售渠道为整个票务市场注入了更多的流动性，同时为消费者提供

了更高额便利性。但是二级市场的卖家因为没有官方授权背书，所以需要为了得到消费者的信任而投入大量成本，超长的回款周期，支付高额的手续费给提供担保的市场如Viagogo，Stubhub，淘宝等。提供担保的市场同样承担着巨额的担保风险。

区块链能够带来的变化

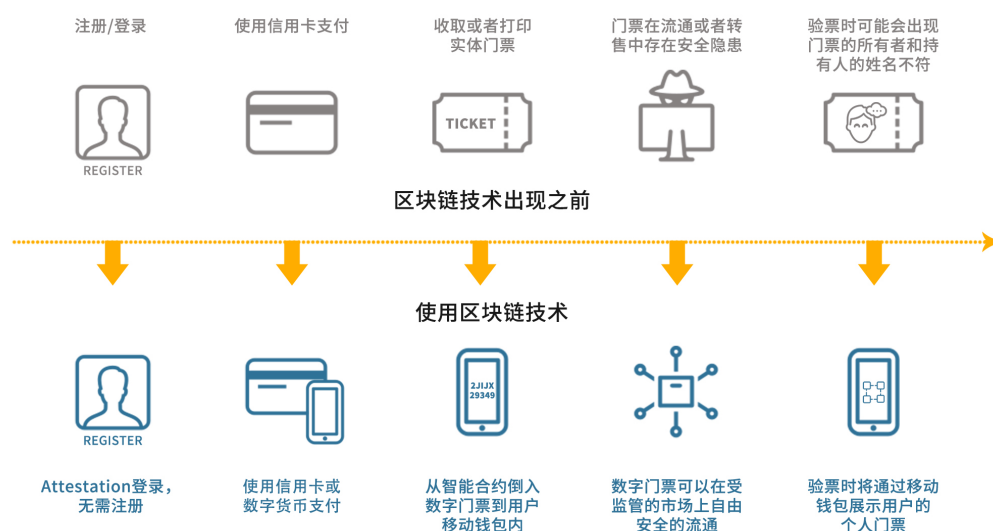


图4 区块链给消费者带来的便利

对于主办方

- 可在在智能合约内定义各种规则来实现对二级市场的监管。比如票A只允许转售5次，转售价格不得超过100，每次转售最高利润不得超过10%，不超过5%的部分FIFA不动，超过5%的部分返给FIFA里面的50%等；

- 可以直接采集到二级市场规模的具体数字，门票流通的全部过程可监控；
- 可以直接从每一笔二级市场的交易中获利，按规定的比例自动返回主办方钱包；
- 可以直接连接到每一个持票用户，知道每一个用户的钱包地址。

对于消费者

监管下的二级市场会更加健康，并且保持自由提供足够流动性。

- 票价变低；
- 不会再买到假票；
- 随时能买到票；
- 享受数字化带来的便利；
- 一些额外好处，比如持票在其他地方消费打折等。

对于官方授权的市场平台

- 各种销售平台市场没机会再垄断数据和资金流了。数据代表的信息所有权归主办方和消费者。主办方在消费者允许的情况下掌控全部数据。资金流在货款对付的原子化交易下，不再经过销售平台市场，直达主办方或者其他门票销售方。
- 销售平台市场的功能简化到：市场宣传和撮合交易，不再提供结算和数据整理分析。

对于官方售票机构

- 可以更容易的拓展自己的销售渠道，因为区块链门票的流通渠道没有准入许可，任何机构都

可以加入到系统帮忙售票，不再需要不用系统之间的集成；

- 之前依靠用户数据部分的收入降低；
- 之前依靠收单结算部分的收入降低。

对于二级市场平台方

- 类似于官方授权市场，二级市场平台方也没机会再垄断数据和资金流了；
- 票款对付的原子化交易彻底替代掉了二级市场平台方的担保功能，同时迫使二级市场平台方大幅降低收费比率（现阶段每一个交易15-30%的交易额用来支付二级市场平台方的服务费）。

对于二级市场内售票机构

- 不再需要中介平台方提供信用担保，大幅扩大销售面，降低交易门槛，大幅提高客户数量；
- 票款对付的原子化加一，及时收账；
- 受主办方监管，所以无法再赚取单张超高额暴利；
- 总的来看，变得薄利多销。

支付欺诈

支付欺诈是所有涉及收单结算业务的公司不可避免的一个大问题，大量的信用卡盗刷，假信用卡，信用卡争议等等造成商家巨额损失。对于活动门票行业来说，这个需要两部分同时支持：区块链门票加上区块链货币（加密货币）。只有两者同时使用的情况下，才能实现原子化的货款对付，彻底杜绝支付欺诈。

活动安全KYC

这部分并不是区块链门票带来的直接好处，而是说数字化门票可以带来的好处，区块链是数字化门票的最好解决方案。这里主要会用到认证技术（Attestation），这个技术最好落地位置是在用户端的客户端（加密钱包）上面。客户端通过自我生成或者导入第三方认证机构提供的Attestation（如用户自己通过手机短信认证手机号码，或如政府提供的身份证，护照等），智能合约和网站内置校验Attestation的方法。智能合约和网站在不同用户提交不同的Attestation时，给予不同的反馈。最终实现类似于：

- 任何不在黑名单上的用户都可以持有门票A；
- 只有中国用户可以持有门票B；
- 只有出生在1958年之前的用户可以持有门票C；
- 只有用户XXX可以持有门票D；
- 只有用户XXX可以持有门票D，而且在需要检票入场的时候，需要用户XXX的生物特征来解锁用户加密钱包内的门票D。

通过把门票绑定部分用户信息，可以实现各种级别的KYC。通过严格的KYC可以大幅提高活动安全性。

监管下的自由流通

在公有链上开发智能合约发行token

- 公有链帮助token确权，能让token实现无准入许可的自由流通。
- 智能合约能提前制定每一个token的流通/使用规则。

两者结合实现监管下的自由流通，下面来具体说一说：

在智能合约内，token可以代表各种事/物/权，可以是数字世界内的事/物/权，也可以是物理世界内的事/物/权。为了方便理解，可以把token按照代表的事/物/权属性，分成两种类型：

- **token物理** - token代表物理世界内的事/物/权
- **token数字** - token代表数字世界内的事/物/权

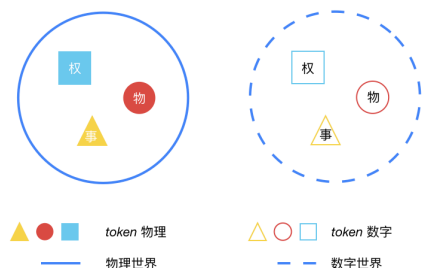


图5 token物理和token数字

token的流通和“使用”方式的四种方式：

1. token数字-在数字世界内流通-在数字世界内交易/使用
2. token数字-在数字世界内流通-在物理世界内交易/使用

3. token物理-在数字世界内流通-在物理世界内交易/使用
4. token物理-在数字世界内流通-在数字世界内交易/使用

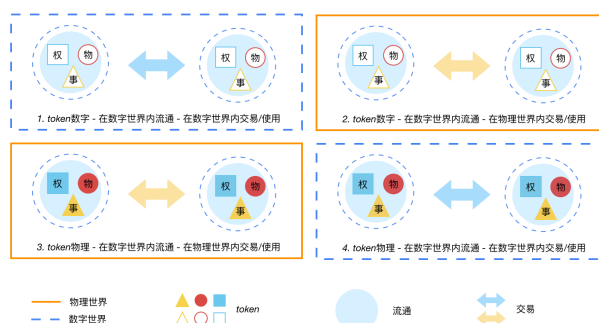


图6 token的流通和使用方式

token数字-在数字世界内流通-在数字世界内交易/使用，这个大家见的多了，如各种公链的消耗型代币，如ETH等等，用数字货币购买数字资产等等，各种数字货币之间交易等等。

token数字-在数字世界内流通-在物理世界内交易/使用，这个也很多，也是最热门的，如各种以替代货币为出发点的公链，如BTC等，使用token作为数字货币购买物理世界内的事/物/权等。

token物理-在数字世界内流通-在数字世界内交易/使用，使用比较少见，因为物理世界的事/物/权不容易在数字世界内使用，另外这类型的token现在很少。

token物理-在数字世界内流通-在物理世界内交易/使用，大部分连接真实世界的应用场景，如门票应用的都是“3.token物理-在数字世界内流通-在物理世界内交易/使用”。

每当token需要和物理世界连接的时候都少不了中心信任机构，他们是把物理世界里面的事/物/权token化的网关，也是“使用”token实现物理事/物/权时的网关。这个中心信任机构要有足够的信用度来支撑他发行的token所代表的事/物/权，来完成网关功能：

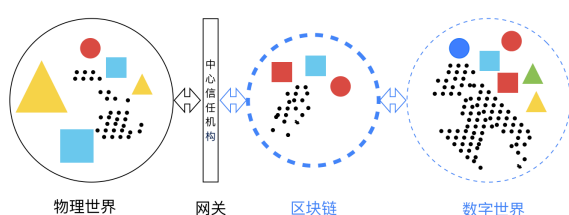


图7 中心信任机构：网关

- 比如包子铺能发行代表他一个一个包子的token，他的信用度可以让所有人相信，在用户“使用”这个token的时候可以得到一个包子。

- 比如房产所可以用token代表房屋的所有权，他的信用度可以让所有人相信，我拥有这个token就等同于拥有这个房屋的所有权。
- 比如房东和房屋中介一起可以用token代表自己房屋1-3月的使用权，他们的信用度可以让所有人相信，我拥有这个token就可以在1-3月使用这个房子。
- 比如国家可以发行替代货币的token，他的信用度可以让所有人相信，我“拥有和使用”这个token和“拥有和使用”现有货币是一样。

物理世界内的事/物/权在公有链上面token化，就会为事/物/权带来自由流通的属性。这种流通属性的产生原因是区块链确权，原本需要第三方机构确权的事情被一个公开的共享账本替代了，带来的好处就是，任何流通渠道都可以自由的加入token的流通，原本需要和第三方签合同，打通数据库，现在不用了。与此同时，中心信任机构可以在发行token的智能合约内设置各种规则，来实现每个token自带规则的在公有链上自由流通，从而实现监管下的自由流通。

2018俄罗斯世界杯ERC 875门票测试结果

技术方案

2018俄罗斯世界杯ERC 875门票技术方案	
公有链	以太坊
智能合约标准	ERC 875 NFT (Non-fungible Token) 标准
合约地址	0xA66A3F08068174e8F005112A8b2c7A507a822335
合约代码	基于ERC 875 开发 https://github.com/alpha-wallet/contracts/blob/master/TicketingContract.sol
主办方检票端	Usher Mobile App 近期开源，代码请参考 https://github.com/alpha-wallet
用户客户端	AlphaWallet 近期开源，代码请参考 https://github.com/alpha-wallet
主办方数据整理工具	Akio的解决方案

表2 2018俄罗斯世界杯ERC 875门票技术方案

公有链：以太坊

选用以太坊是因为：

1. 世界杯门票对流动性有着极高的要求，需要使用的公链是最广泛被人接受的；
2. 世界杯是全球顶级赛事，测试过程不能有任何闪失，以太坊是支持智能合约的公链内，最成熟稳定的；
3. 世界杯门票价值比较高，对比来看，以太坊的gas费用影响不大；

4. 世界杯门票都是提前销售，就算二级市场内也很少有最后一分钟确认交易的需求，所以现有以太坊的性能可以支撑。

智能合约标准：ERC875 NFT (Non-fungible Token) 标准

比较现有两个NFT标准ERC721和ERC875后，选用ERC875的原因是：

- a) 主办方通过任意渠道发送一个链接 (MagicLink) 给用户。用户通过MagicLink就能直接导入1张或多张世界杯门票。不需要用户持有以太币。
- b) 用户可以自己使用AlphaWallet创建MagicLink来导出自己的门票，发送MagicLink给接收方用户，接收方点击链接，盛开体育支付gas，接收方倒入门票。免费转让时，发送方和接收方都不需要支付gas。
- c) 用户可以使用AlphaWallet创建MagicLink来出售他们的门票，发送MagicLink给买家或者把MagicLink发布在各种市场内，买家点击MagicLink，在AlphaWallet倒入页面内检查门票信息和价格，点击确认购买，原子化交易发生，买家支付票价和gas费用收到门票，同时卖家收到钱门票消失。

- 合约地址：
0xA66A3F08068174e8F005112A8b2c7A507a822335

<https://github.com/alpha-wallet/contracts/blob/master/TicketingContract.sol>

[illegible]

```
pragma solidity ^0.4.17;
contract TicketPro
{
    mapping(address => bytes32[]) inventory;
    uint16 ticketIndex = 0; //to track mapping in tickets
    address organiser;
    address paymaster;
    uint numOfTransfers = 0;
    string public name;
    string public symbol;
    uint8 public constant decimals = 0; //no decimals as tickets cannot be split

    event Transfer(address indexed _to, uint16[] _indices);
    event TransferFrom(address indexed _from, address indexed _to, uint16[]
_indices);
    event Trade(address indexed seller, uint16[] ticketIndices, uint8 v, bytes32 r,
bytes32 s);
    event PassTo(uint16[] ticketIndices, uint8 v, bytes32 r, bytes32 s, address
indexed recipient);

    modifier organiserOnly()
    {
        if(msg.sender != organiser) revert();
        else _;
    }

    modifier payMasterOnly()
    {
        if(msg.sender != paymaster) revert();
        else _;
    }
}
```



```

function() public { revert(); } //should not send any ether directly

constructor (
    bytes32[] tickets,
    string nameOfContract,
    string symbolForContract,
    address organiserAddr,
    address paymasterAddr,
    address recipientAddr) public
{
    name = nameOfContract;
    symbol = symbolForContract;
    organiser = organiserAddr;
    paymaster = paymasterAddr;
    inventory[recipientAddr] = tickets;
}

function getDecimals() public pure returns(uint)
{
    return decimals;
}

// example: 0, [3, 4], 27,
"0x9CAF1C785074F5948310CD1AA44CE2EFDA0AB19C308307610D7BA2C74604AE98",
"0x23D8D97AB44A2389043ECB3C1FB29C40EC702282DB6EE1D2B2204F8954E4B451"
// price is encoded in the server and the msg.value is added to the message digest,
// if the message digest is thus invalid then either the price or something else in the message is
invalid
function trade(uint256 expiry,
    uint16[] ticketIndices,
    uint8 v,
    bytes32 r,
    bytes32 s) public payable
{
    //checks expiry timestamp,
    //if fake timestamp is added then message verification will fail
    require(expiry > block.timestamp || expiry == 0);

    bytes32 message = encodeMessage(msg.value, expiry, ticketIndices);
    address seller = ecrecover(message, v, r, s);

    for(uint i = 0; i < ticketIndices.length; i++)
    { // transfer each individual tickets in the ask order
        uint16 index = ticketIndices[i];
        assert(inventory[seller][index] != bytes32(0)); // 0 means ticket gone.
        inventory[msg.sender].push(inventory[seller][index]);
        // 0 means ticket gone.
        delete inventory[seller][index];
    }
    seller.transfer(msg.value);
}

emit Trade(seller, ticketIndices, v, r, s);
}

function loadNewTickets(bytes32[] tickets) public organiserOnly
{
    for(uint i = 0; i < tickets.length; i++)
    {
        inventory[organiser].push(tickets[i]);
    }
}

function passTo(uint256 expiry,
    uint16[] ticketIndices,
    uint8 v,
    bytes32 r,
    bytes32 s,
    address recipient) public payMasterOnly
{
    require(expiry > block.timestamp || expiry == 0);
    bytes32 message = encodeMessage(0, expiry, ticketIndices);
    address giver = ecrecover(message, v, r, s);
    for(uint i = 0; i < ticketIndices.length; i++)
    {
        uint16 index = ticketIndices[i];
        //needs to use revert as all changes should be reversed
        //if the user doesn't hold all the tickets
        assert(inventory[giver][index] != bytes32(0));
        bytes32 ticket = inventory[giver][index];
        inventory[recipient].push(ticket);
        delete inventory[giver][index];
    }
}

emit PassTo(ticketIndices, v, r, s, recipient);
}

//must also sign in the contractAddress
function encodeMessage(uint value, uint expiry, uint16[] ticketIndices)
    internal view returns (bytes32)
{
    bytes memory message = new bytes(84 + ticketIndices.length * 2);
    address contractAddress = getContractAddress();
    for (uint i = 0; i < 32; i++)
    { // convert bytes32 to bytes[32]
        // this adds the price to the message
        message[i] = byte(bytes32(value << (8 * i)));
    }

    for (i = 0; i < 32; i++)
    {
        message[i + 32] = byte(bytes32(expiry << (8 * i)));
    }

    for(i = 0; i < 20; i++)
    {
        message[64 + i] = byte(bytes20(bytes20(contractAddress) << (8 * i)));
    }

    for (i = 0; i < ticketIndices.length; i++)
    {
        // convert int[] to bytes
        message[84 + i * 2] = byte(ticketIndices[i] >> 8);
        message[84 + i * 2 + 1] = byte(ticketIndices[i]);
    }

    return keccak256(message);
}

function name() public view returns(string)
{
    return name;
}

function symbol() public view returns(string)
{
    return symbol;
}

function getAmountTransferred() public view returns (uint)
{
    return numOfTransfers;
}

function balanceOf(address _owner) public view returns (bytes32[])
{
    return inventory[_owner];
}

function myBalance() public view returns(bytes32[]){
    return inventory[msg.sender];
}

function transfer(address _to, uint16[] ticketIndices) public
{
    for(uint i = 0; i < ticketIndices.length; i++)
    {
        uint index = uint(ticketIndices[i]);
        assert(inventory[msg.sender][index] != bytes32(0));
        //pushes each element with ordering
        inventory[_to].push(inventory[msg.sender][index]);
        delete inventory[msg.sender][index];
    }
    emit Transfer(_to, ticketIndices);
}

function transferFrom(address _from, address _to, uint16[] ticketIndices)
    organiserOnly public
{
    for(uint i = 0; i < ticketIndices.length; i++)
    {
        uint index = uint(ticketIndices[i]);
        assert(inventory[_from][index] != bytes32(0));
        //pushes each element with ordering
        inventory[_to].push(inventory[msg.sender][index]);
        delete inventory[_from][index];
    }
}

emit TransferFrom(_from, _to, ticketIndices);
}

function endContract() public organiserOnly
{
    selfdestruct(organiser);
}

function isStormBirdContract() public pure returns (bool)
{
    return true;
}

function getContractAddress() public view returns(address)
{
    return this;
}
}

```

近期开源，代码请参考<https://github.com/alpha-wallet>

- <https://github.com/alpha-wallet/contracts/blob/master/TicketingContract.xml>


- AlphaWallet为自定义XML文档提供4个不同安全级别的签名，在自定义的同时保证安全性。

- AlphaWallet可以在平台内直接运行盛开体育的门票应用，无需跳转其他应用，也不使用用户体验极差的DApp浏览器。世界杯门票token在需要检票的时候，可以转化为动态线下二维码（每30秒钟变一次，不需要用户端在线）

近期开源，代码请参考<https://github.com/alpha-wallet>



<https://akiolabs.com/product/nft/>



SHANKAR
盛开体育

FIFA WC2018

Shankar is a young, fast moving sports marketing company. For the upcoming World Cup in Russia, Shankar is releasing a new football selection which is powered by **AdvaVoter** (Disrupted PTE, U2D) and powered by **blockchain** technology. The tickets will be issued on the **ethereum** blockchain using **ERC20**, a unique token standard to handle real world use-cases.

Market Cap
₹0

Total Assets
67

Average Price
₹0

Activity

1 day 7 days 30 days All Time

Users Reached
62

Items Sold
67

Transactions
116

Volume
₹8

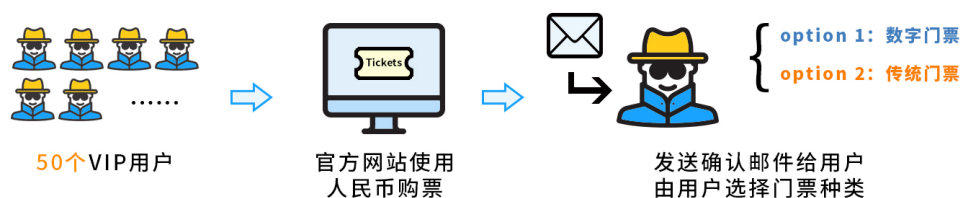
Item	Current Price	# Transfers	# Users Reached	Volume
Ticket #0	₹0	12	8	₹0
Ticket #1	₹0	6	5	₹0
Ticket #2	₹0	5	4	₹0
Ticket #3	₹0	3	3	₹0
Ticket #4	₹0	3	3	₹0
Ticket #5	₹0	3	2	₹0
Ticket #6	₹0	3	2	₹0
Ticket #7	₹0	3	3	₹0

图8 盛开合约内全部门票信息

[illegible]

图9 单张门票转手信息

测试内容



我们抽取了50个开幕式VIP用户，进行了测试。这些用户在盛开体育官方售票网站，使用人民币购票之后，收到了盛开体育发给他们的确认邮件，在邮件内提示用户可以选择数字门票解决方案或者传统方式，数字门票部分含有一个MagicLink以及相应的使用说明，用户可以选择是否把他们的门票兑换卷转化为区块链门票并导入AlphaWallet。如果转化成区块链门票，原有门票兑换卷自动作废。用户可以在开赛前到盛开体育在俄罗斯各处设立的门票兑换柜台，兑换最终的纸质门票。

测试目的和结果

1. 用户对全数字门票方案的接受程度

最终有**28人**选择区块链门票，并顺利使用区块链门票兑换到了纸质门票，转化率高于预期，我们有理由相信，如果强制用户只能使用数字门票，用户也不会有意见。

2. 用户在使用过程中是否有我们没有预料到的问题发生（用户端APP）

一些国家出于保护消费者远离数字货币欺诈的原因采取了一刀切的政策，不允许区块链应用上架相应的国家的app市场。

3. 主办方在使用过程中是否有我们没有预料到的问题发生（主办方端APP）

很多用户不希望自己的区块链门票被销毁掉，希望可以留下来做纪念。基于本次测试只有28人使用区块链门票，最后盛开体育同意用户保留门票，没有把用过的区块链门票销毁，而是选择了额外记录28人使用信息的方式来防止双花。

4. 以太坊不断变化的使用环境下，对用户体验造成的影响（如表3所示）

用户体验	造成的影响	
	对用户	盛开体育
门票导入	影响比较大 门票只有在以太坊确认交易后才会显示。以太坊拥堵情况下，交易确认最长延迟达14小时。	有一定影响 用户有可能会不断打电话给盛开，要求确认是否已经成功导入门票。
门票使用	基本没有影响 动态二维码可以离线使用。影响仅限于，被销毁的票什么时候从用户端消失。	基本没有影响 有可能有用户，尝试在门票销毁前重复使用，但不会成功。
门票转让	影响比较大 门票只有在以太坊确认交易后才会显示。以太坊拥堵情况下，交易确认最长延迟达14小时。	有一定影响 用户有可能会不断打电话给盛开，要求确认是否已经成功转让。
门票出售	有一定影响 用户需要学习了解什么是以太币，为什么出售后收到的是以太币，有了基础知识后，对于交易确认延迟也会有一定理解。建议用户要交易的话，最好留出富裕时间。	基本没有影响 因为买卖双方都对以太坊有了一定了解

表3 以太坊不断变化的使用环境下对用户体验造成的影响



官方网站 <https://www.awallet.io>

联系邮箱 info@awallet.io

联系我们

-  <https://www.facebook.com/AlphaWallet/>
-  <https://www.linkedin.com/company/qwallet/>
-  https://twitter.com/Alpha_wallet
-  <https://www.reddit.com/r/AlphaWallet/>
-  <https://github.com/alpha-wallet>