## Substrate Non-fungible Token Design

1. 综述

设计分为三层结构，即 Order -- Nft -- Erc721。中间层为NFT对象，每个NFT对象具有一个管理账号（称为发行人），和一些与管理、权限相关的属性集合。下层为Token对象，每个Token必须属于某一个NFT对象，Token所属的NFT对象的id称为Token的类型即NFT\_ID，发行人在发行Token时，可以指定该Token的属性，Token发行后，链上会为其分配一个唯一ID，用来标识该Token即TOKEN\_ID。上层为与交易相关的Order层，包括选择器的定义以及NFT专属的交易引擎的设计。

选择器：用户持有Token后，可通过选择器，选中一些Token进行操作。选择器有两类，第一类为Token ID集合，用于指定选择部分Token，第二类为属性选择器。属性选择器可以定义最大/最小过期时间，以及使用逻辑树选择Token自定义属性符合用户定义逻辑的Token。通过使用选择器，普通账号可以转让、出售、求购、销毁Token。出售/求购Token会产生一个卖单/买单，每个买卖单具有链上唯一ID，用户可以通过ID进行撤单操作。

引擎：1)下购买单后，节点会检查选择器中的token，若为ID选择器，则累加选中Token的amount作为总买量，若为属性选择器，则以选择器中的max\_amount作为总买量，向购买账号中预扣除买价\*总买量的资产，若余额不足，则下单失败。下单后立刻进行一次匹配，按价格优先、时间优先的顺序，遍历卖单队列中价格优于本买单的卖单，对于卖单的每一个关联token，若其与本买单的选择器匹配，则转移token所有权，向卖家支付以卖价为单价，乘以Token记账数量的资产，若卖价低于买价，则向购买人退还差价乘以Token记账数量的资产。匹配完成后，若买单数量仍有剩余，则检查此买单是否具有immediate\_or\_cancel属性，若有，则自动取消掉未撮合的余量部分。否则，买单剩余数量停留在买单队列中。2)下出售单后，节点会检查选择器选中的token，若为ID选择器，需要满足所选中的token属于seller，且不在出售状态。选中的token的bind\_order字段会与该卖单关联。下单后立刻与当前所有买单进行一次匹配。买单按照价格优先、时间优先的规则排序，价格高者排名更前。对所有买价高于出售价的买单，将所有选中的token与买单中的选择器匹配，若匹配成功，则转移该Token的所有权，并向卖家支付以买单价格为单价，乘以Token记账数量的资产。匹配完成后，若卖单中出售数量仍有剩余，则检查此卖单是否具有immediate\_or\_cancel属性，若有，则自动取消掉未撮合的余量部分。否则，卖单剩余数量停留在卖单队列中。

说明：Token不能分割，当用户转让、出售、求购Token时，Token所有权完整转移给另一个账户。

1. 实体
   1. NftMeta

"NftMeta":{

"total\_supply": "Balance", // amount of tokens issued

"issuer": "AccountId", // 发行人

"symbol":"Vec<u8>",// symbol name of this nft

"nft\_id": "Hash",

"option": "NonfungibleOption",// 其他属性

},

"NonfungibleOption":{

"permissions":"Vec<Permission>",// 权限设置，黑白名单

"max\_supply":"Balance",// 该NFT下最大可发行的token个数

"description":"Vec<u8>"

},

* 1. Token

"Token":{

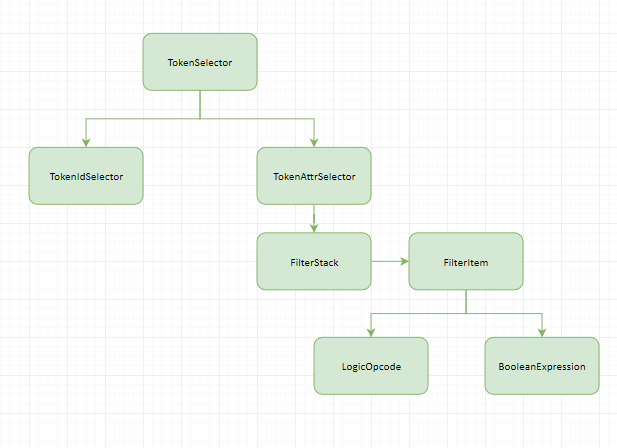
"token\_id": "Hash",

"symbol": "Vec<u8>", // symbol of this token

"nft\_id": "Hash", // 所属NFT 类型

},

* 1. 选择器selector



* + 1. TokenId选择器
    2. 属性选择器

"FilterItem": { "\_enum":{

"Uint8T":"LogicOpcode",

"BoolExp":"BooleanExpression"}

},

"FilterStack" : "Vec<FilterItem>" ,

"TokenParser" :{

"s":"VecDeque<LogicOpcode>",

},

"TokenAttrSelector":{

"max\_count": "u32",

"stack":"FilterStack",

},

"TokenIdSelector":{

"id\_set": "Vec<Hash>",// token\_id\_type : H::Hash

},

"SelectorType":{ "\_enum":{

"IdSelect":"TokenIdSelector<Hash>",

"AttrSelect":"TokenAttrSelector",}

},

"TokenSelector":{

"selector": "SelectorType",

"nft\_type": "Hash",

},

1.3.2.1操作符

"CompareOpcode" : {"\_enum":[ // 比较操作符

"TokenCmpEq",

"TokenCmpGt",

"TokenCmpLt",

"TokenCmpGe",

"TokenCmpLe",

"TokenCmpNe",

"TokenCmpMax",

]},

"LogicOpcode":{"\_enum":[ // 逻辑操作符

"TokenLogicAnd" ,

"TokenLogicOr",

"TokenLogicXor",

"TokenLogicMax",

"TokenCmpTrue",

"TokenCmpFalse",

]},

1.3.2.2 右值元素

"TokenAttrValType":{

"\_enum": {

"String":"Vec<u8>", // 字符串类型

"Uint64":"u64" // 整形

}

},

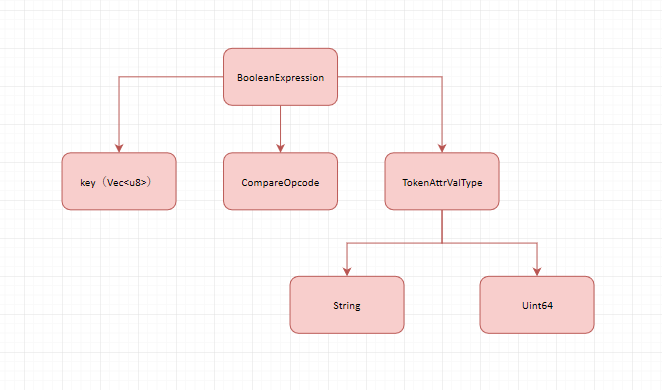
"BooleanExpression":{

"op": "CompareOpcode", // 操作符

"key": "Vec<u8>", //

"val" : "TokenAttrValType",

},



* 1. 订单order
     1. 买单

"BidOrderItem":{

"creator": "AccountId",

"order\_id": "Hash",

"selector": "TokenSelector",

"asset": "AssetId",

"price": "Balance",

"timepoint": "Moment",

"immediate\_or\_cancel": "bool",

"tk\_count\_to\_buy": "Balance",

"status": "OrderStatus",

},

* + 1. 卖单

"AskOrderItem":{

"creator": "AccountId",

"order\_id": "Hash",

"selector": "TokenSelector",

"asset": "AssetId",

"price": "Balance",

"timepoint": "Moment",

"immediate\_or\_cancel": "bool",

"bind\_tokens": "Vec<Hash>",

"status": "OrderStatus",

},

* + 1. 状态order status

"OrderStatus":{

"\_enum": [

"Open",

"PartialFilled",

"Filled",

"Closed",

"Canceled"

]

}

1. 操作
   1. Erc721 create
   2. Erc721 burn
   3. Erc721 transfer
   4. Erc721 approve (single token)
   5. Erc721 approve all (under one account)
   6. Token issue (under nft)
   7. Token burn
   8. Token reserve
   9. Token unreserve
   10. Nft create
   11. Nft update
   12. Set token attribute
   13. Remove token attribute
   14. 挂买单
   15. 挂卖单
   16. 取消买单
   17. 取消卖单
2. 查询Api