dGoods: 디지털 굿즈 토큰 스펙 v0.1

Cameron Thacker* Stephan Cunningham* Rudy Koch* John Linden*

Translation: Eon Ha* (ITAM.GAME)

1. 소개

dGoods 는 EOS 블록체인의 디지털 및 물리적 아이템의 가상표현을 위한 오픈소스 및 무료 스탠더드이다. 초기 초점은 비디오 게임을 위한 자산이지만, 다른 유형의 디지털 굿즈도 배제하지 않도록 디자인하고 있다. 이는 EOS 블록 체인에서 Non-Fungible, Semi-Fungible, 및 Fungible 토큰을 지원하는 스탠더드가 될 수 있다고 생각한다.

게임이나 회사는 하나의 메인 토큰 심볼 아래에 여러 가지 토큰을 생성을 원할 것이다. 이 중 Fungible 및 Semi-Fungible 토도 있지만 Non-Fungible 토큰도 있을 것이다. 이 스탠더드와 다른 NFT 스탠더드과의 차이점은 토큰의 계층적 명명 구조에 있다. Fungible 토큰은 symbol:category:name 으로 식별되지만 NFT 는 추가적인 토큰 ID 가 있다. 이를 통해 전례 없는 토큰 구성이 가능해진다. 또한 지갑 및 dApp 들이 토큰을 카테고리 또는 이름별로 표시하여 검색 및 필터링 기능을 제공한다.

우리의 철학은 작게 시작해 유연성을 유지하며 다양하고 혁신적인 개발 커뮤니티를 지원하는 강력한 디지털 굿즈 스탠더드를 제공하는 것이다. 따라서 이 규격의 대부분은 팀들이 가상 아이템을 쉽게 통합하고 표시할 수 있도록 도와주는 중요한 기능 개선 사항을 추가한 스탠더드 EOS 토큰에 논리적인 확장이다. 따라서 다음 섹션에서는 이 규격을 구현하기 위한 최소한의 필수 메소드 및 기능 세트들, 테이블 구조, 마지막으로 몇 가지 예제 메타데이터 템플릿들을 설명한다.

2. 필수 메소드

CREATE: create 메소드는 토큰을 인스턴스화한다. 이는 토큰을 발행하기 전에 필수이며 카테고리, 이름, 최대 공급, fungible vs non-fungible, 토큰발행 자격과 같은 속성을 설정한다. 또한, 처음으로 호출할 때 심볼이 기록되며 후속 호출에서는 그 심볼을 지정해야 한다. 심볼은 A-Z, 최대 7 여야 한다. Name type 은 a-z, 1-5, 최대 12 자 문자열이다.

---

* Mythical Games

ACTION create(name issuer, string symbol, name category, name token_name, bool fungible, bool burnable, bool transferable, int64_t max_supply);

ISSUE: issue 메소드는 토큰을 만들어 'to' account name 에게 소유권을 부여한다. 유효한 호출을 위해서는 symbol, category, 및 token name 이 먼저 생성되어야 한다. Non-Fungible 또는 Semi-Fungible 일 경우 수량은 1 이어야 하며, 그렇지 않을 경우 수량은 0.0001 이상이어야 한다.

ACTION issue(name to, string symbol, name category, name token_name, double quantity, string metadata_uri, string memo);

PAUSEXFER: 모든 토큰의 모든 전송을 일시 중지하는 기능. 계약에 의해서만 호출 가능. Pause 가 true 일 경우 중지되며, Pause 가 false 일 경우 중지 해제된다.

ACTION pausexfer(bool pause);

BURNNFT: Burn 메소드는 특정의 토큰을 파괴하고 RAM 을 해제한다. 오너 (owner)만 Burn 기능을 호출할 수 있으며 burnable 은 true 이어야한다.

ACTION burnnft(name owner, vector<uint64_t> tokeninfo_ids);

BURN: Burn 메소드는 Fungible 토큰을 파괴하고 모두 다 삭제되면 RAM 을 해제한다. 오너(owner)만 Burn 기능을 호출할 수 있으며 burnable 은 true 이어야한다.

ACTION burn(name owner, uint64_t global_id, double quantity);

TRANSFERNFT: Non-Fungible 토큰 전송을 위해 사용된다. 토큰 ID 목록을 전달하여 하나의 함수 호출에서 토큰을 일괄적으로 보낼 수 있게 한다. 토큰 오너 만이 이 함수를 호출할 수 있으며 transferable 은 true 이어야한다.

ACTION transfernft(name from, name to, vector<uint64_t> tokeninfo_ids, string memo);

TRANSFER: standard transfer 메소드는 fungible 토큰에서만 호출할 수 있다. 토큰을 개별적으로 지정하지 않고, 해당 global id 와 전송할 금액으로 지정한다.

ACTION transfer(name from, name to, uint64_t global_id, double quantity);

---

3. 토큰 데이터

3.1 Symbol Info Table

Singleton 은 컨트렉트의 심볼을 보유하고 create 의 첫 번째 호출을 설정한다. Create 이 성공적으로 호출될 때마다 global id 는 증가한다.

```
//scopeisself   TABLE symbolinfo {
      string symbol;
      uint64_t
      global_id;
 };
```

3.2 Token Stats Table

하나의 카테고리, token name pair, 만 있을 수 있도록 한다. 주어진 토큰이 fungible 인지 burnable 인지 저장하며, 현재 및 max supply 를 저장한다. 이 정보는 토큰이 생성될 때 저장된다.

```
 //scopeiscategory,thentoken_nameisunique
 TABLE tokenstats  {
      uint64_t
      global_id; name
                issuer;
      string  symbol;
      name     category;
      name
                token_na
      me; bool
                fungible;
      bool     burnable;
      bool
```

```
            transferabl
    e;

    double current_supply;
    uint64_t  max_supply;


    uint64_t primary_key() const { return token_name.value; }

};
```

3.3 Token Info Table

이는 Non-Fungible 및 Semi-Fungible 토큰의 글로벌 리스트이다. 2 차 지수는 수유자, 카테고리, 또는 토큰명으로 검색을 제공한다.

```
//scopeisself
TABLE tokeninfo {

    uint64_t id;
    uint64_t
    global_id;

    name owner;
    name
    category;

    name token_name;
    uint64_t serial_number;
    string metadata_uri;


    uint64_t primary_key() const  {  return  id;  }
    uint64_t get_owner() const { return owner.value; }

    uint64_t get_token_name() const { return token_name.value; }
    uint64_t get_category() const { return category.value; }

};
```

3.4 Category Table

편리한 쿼리를 위해 모든 카테고리 이름을 보유한다.

```
//scopeisself
TABLE categoryinfo
  { name category;

  uint64_t primary_key() const { return category.value; }
};
```

3.5 Account Table

Account table 은 계정에 대한 Fungible 토큰 및 그 계정에 특정 유형의 NFT 를 얼마나 가지고 있는지에 대한 레퍼런스를 보유한다.

```
//scopeisowner
TABLE account {
    uint64_t
    global_id; name
    category;

    name
    token_name;
    double amount;

    uint64_t primary_key() const { return global_id; }
};
```

4. 메타데이터 템플릿

지갑 또는 dApp 들이 다양한 디지털 굿즈를 지원하기 위해서는 메타데이터와 관련된 스탠더드가 있어야 한다. 우리의 접근법은 굿즈의 유형에 따라 템플릿을 정의하는 것이다. 다음의 템플릿들은 우리가 생각한 제의들이지만 이 프로세스는 협업 업무이다. 커뮤니티가 합의한 템플릿 저장소를 제공하고 싶다. 모든 메타데이터는 JSON 형식이며 템플릿 유형을 지정해야 한다.

4.1 Type: 3dgameAsset (3d 게임자산)

{

//RequiredFields (필수 필드)

"name": string (문자열); 토큰이 나타내는 자산.

"description": string (문자열); 토큰이 나타내는 자산의 간단한 설명

"imageSmall": 이미지 리소스 사이즈 150 x 150 을 가리키는 URI

"imageLarge": 이미지 리소스 사이즈 1024 x 1024 을 가리키는 URI

"3drender": 3d object 렌더링을 위한 js webgl 을 가리키는 URI

"details": 디테일 뷰로 렌더링 할 Key Value pair. 예를 들어:

{"strength": 5}

//OptionalFields (선택 필드)

"authenticityImage": mime type image 리소스를 가리키는 URI. 정품 인증을 나타낸다

}

4.2 Type: 2dgameAsset (2d 게임자산)

{

//RequiredFields (필수 필드)

"name": string (문자열); 토큰이 나타내는 자산.

"description": string (문자열); 토큰이 나타내는 자산의 간단한 설명

"imageSmall": 이미지 리소스 사이즈 150 x 150 을 가리키는 URI

"imageLarge": 이미지 리소스 사이즈 1024 x 1024 을 가리키는 URI

"details": 디테일 뷰로 렌더링 할 Key Value pair. 예를 들어:

{"strength": 5}

//OptionalFields (선택 필드)

"authenticityImage": mime type image 리소스를 가리키는 URI. 정품 인증을 나타낸다

}

4.3 Type: ticket (티켓)

{

//RequiredFields (필수 필드)

"name": string (문자열); 티켓이 나타내는 이벤트.

"description": string (문자열); 티켓이 나타내는 이벤트의 간단한 설명

"location": string (문자열); 이벤트가 개최되는 장소의 이름

"address": string (문자열); 이벤트가 개최되는 위치의 주소

"date": 유닉스 시간; 이벤트 시작 날짜

"time": 유닉스 시간; 이벤트 시작 시간

"imageSmall": 이미지 리소스 사이즈 150 x 150 을 가리키는 URI
"imageLarge": 이미지 리소스 사이즈 1024 x 1024 을 가리키는 URI

"details": 디테일 뷰로 렌더링 할 Key Value pair. 예를 들어:

{"openingAct":"Nickelback"}

//OptionalFields (선택 필드)

"authenticityImage": mime type image 리소스를 가리키는 URI. 정품 인증을 나타낸다

"duration": string (문자열); 이벤트 지속시간

"row": string (문자열); 좌석의 열

"seat": string (문자열); 좌석번호 또는 GA (General Admission/일반 티켓)

}

4.4 Type: art (아트)

{

//RequiredFields (필수 필드)

"name": string (문자열); 토큰이 나타내는 자산.

"description": string (문자열); 토큰이 나타내는 자산의 간단한 설명

"creator": string (문자열); 아트의 creator(s)

"imageSmall": 이미지 리소스 사이즈 150 x 150 을 가리키는 URI

"imageLarge": 이미지 리소스 사이즈 1024 x 1024 을 가리키는 URI

"date": 유닉스 시간; 작품이 만들어진 날짜

"details": 디테일 뷰로 렌더링 할 Key Value pair. 예를 들어:

{"materials": ["oil","wood"],"location":"NortonSimonMuseum"}

//OptionalFields (선택 필드)

"authenticityImage": mime type image 리소스를 가리키는 URI. 정품 인증 또는 서명을 나타낸다

}

4.5 Type: jewelry (보석)

{

//RequiredFields (필수 필드)

"name": (문자열); 토큰이 나타내는 자산.

"description": string (문자열); 토큰이 나타내는 자산의 간단한 설명

"imageSmall": 이미지 리소스 사이즈 150 x 150 을 가리키는 URI

"imageLarge": 이미지 리소스 사이즈 1024 x 1024 을 가리키는 URI

"image360": 360 이미지 리소스를 가리키는 URI

"manufactureDate": 유닉스 시간; 자산이 제조된 날짜

"manufacturePlace": string (문자열); 자산이 제조된 장소

"details": 디테일 뷰로 렌더링 할 Key Value pair. 예를 들어:

{"caret": 1}

//OptionalFields (선택 필드)

"authenticityImage": mime type image 리소스를 가리키는 URI. 정품 인증 또는 서명을 나타낸다

}

---

5. 정의

- **RAM** - EOS 블록체인의 영구 저장소. EOS 를 전송하여 EOS 블록체인에서 구입해야 한다. 사용하지 않을 경우 팔 수 있다.
- **Account** - EOS 계정은 12 이며 a-z, 1-5, 그리고 네임 서비스 사용시 점 (.) 만 포함한다.
- **Symbol** - 최대 7 자의 대문자 A-Z
- **Fungible** - 이 문맥에서는 EOS 자체처럼 상호 교환이 가능한 토큰을 의미한다. 토큰이 Fungible 일 경우 어떠한 토큰을 전송하든 동등하다.
- **Semi-fungible** - 고유 식별자 (unique identifier)를 가지고 있지만 다른 토큰과 동일하거나 유사한 데이터를 공유하고 있다는 점에서 기술적으로 고유한 토큰을 의미한다. 많은 물리적인 제품들이 이 카테고리에 포함된다 (고유한 일련 번호를 가지고 있지만 이외에는 구별할 수 없기 때문).
- **Non-fungible** - 고유한 토큰 (unique tokens)