Novellia Token Standard PDF

Version 1.3.0

Rektangular Studios

Herein, we describe a metadata standard for Cardano tokens.

Novellia Platform

Get ready to take back gaming!

The Novellia Platform is a decentralized game marketplace, built on blockchain technology, ultimately reducing fees for everyone, and preventing censorship.

As a first slice in our initiative, we introduce our own metadata standard we feel will better serve the needs of the Novellia marketplace. Feedback is highly welcome. If our format is widely adopted, we expect this to help the Cardano token ecosystem while also ensuring our own tokens are interoperable with as many clients as possible.

To learn more, check out our website.

721 Token Standard

721 is the most used metadata standard for Cardano Tokens. When developers mint tokens using a common format, applications such as wallets, mobile apps, and games can easily support all tokens that conform.

The 721 metadata is documented here.

Some existing Cardano projects have already used variations of the 721 metadata format:

- CardanoKidz | Rare NFT's on Cardano! | Collect them all
  - Example Metadata On-chain
- SpaceBudz
  - Example Metadata On-chain

```
{
  "721": {
    "cbc34df5cdikojgoaijerg0e9urg904i3209ri09ferogjerge": {
      "Draculi": {
        "id": 1,
        "name": "Draculi",
        "image": "ipfs://ipfs/QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
        <other properties>
      },
      "IscaraTheTenThousandGuns": {
        "id": 2,
        "name": "Iscara the Ten Thousand Guns",
        "image": "ipfs://ipfs/QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
        <other properties>
      }
      ...
    }
  }
}
```

Off-chain version

```
{
  "721": {
    "cbc34df5cdikojgoaijerg0e9urg904i3209ri09ferogjerge": {
      "offchain": "ipfs://ipfs/QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv"
    }
  }
}
```

Where **offchain** points to a file like

```
{
  "Draculi": {
  "id": 1,
  "name": "Draculi",
  "image": "ipfs://ipfs/QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
  <other properties>
  },
  "IscaraTheTenThousandGuns": {
  "id": 2,
  "name": "Iscara the Ten Thousand Guns",
  "image": "ipfs://ipfs/QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
  <other properties>
  }
  ...
}
```

Unfortunately, this format has some limitations:"offchain": "ipfs://ipfs/<IPFS_HASH>"

- Single image asset
  - A large image would download slowly without a thumbnail also being provided
  - There is no indication of what kind of file the image is
  - Not all tokens have an image
- Limited metadata
  - No copyright, publisher
  - No description

Further, metadata on-chain has a 64 character limit for strings and the 721 standard does not provide a canonical way to circumvent this other than choosing one of:

- on-chain metadata
- off-chain metadata

This is particularly noticeable for URLs which may choose a strategy like the below to circumvent this:

```
{
    "host": "www.website.com",
    "path": "/path/to/my/nft"
}
```

A hybrid solution where off-chain metadata takes precedence over on-chain metadata is preferable due to the possibility of off-chain storage becoming defunct.

If we assume metadata resources will eventually become broken, we should design our metadata standard such that holders will at least be able to identify their tokens with some basic text information.

So what's the Novellia Token Standard?

We have two options:

- Extend the 721 format by fulfilling mandatory fields and adding our own. We retain the "721" symbol for compatibility.
- We use an entirely new "label" symbol and expect clients to support us.

We choose the first option because:

- we cannot expect clients to support multiple standards at this early stage
- we would like our own tokens to work to a passable degree on clients only supporting 721

We would prefer if other developers did not create a platitude of different ways of specifying 721 extensions, so we elect to make the Novellia Token Standard a specific extension where others could also be applied using composition.

We refer to the **Extended 721 Token Standard** as a metadata format supporting a few simple fields in addition to the base 721 features:

- Copyright
- Publisher
- Version
- Extension

We refer to the **Novellia Token Standard** as a specific extension of the **Extended 721 Token Standard**.

The first version of the **Novellia Token Standard** is denoted by the extension string **"novellia_1"**. The "1" indicates the major version of the specification.

Which fields are mandatory?

Anything required by **721** should be set for basic backwards compatibility, otherwise clients should assume that fields will be incorrect or broken, and try to recover from missing information. Basically, if you want your token to work, put the data you want a client to be able to use.

Note that we do not refer to this as an NFT standard as there is nothing forcing a native token to be non-fungible.

Extended 721 Token Standard

```
{
  "721": {
    "copyright": "Copyright Rektangular Studios Inc.; all rights reserved",
    "publisher": [
      "rektangularstudios.com"
    ],
    "version": 1,
    "extension": [
      "novellia_1"
    ],
    "cbc34df5cdikojgoaijerg0e9urg904i3209ri09ferogjerge": {
      "Draculi": {
        "id": 1,
        "name": "Draculi",
        "image": "ipfs://QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
        <other properties>
      },
      "IscaraTheTenThousandGuns": {
        "id": 2,
        "name": "Iscara the Ten Thousand Guns",
        "image": "ipfs://QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
        <other properties>
      },
      ...
    }
  }
}
```

Novellia Token Standard


```
{
  "721": {
    "copyright": "Copyright Rektangular Studios Inc.; all rights reserved",
    "publisher": [
      "rektangularstudios.com"
    ],
    "extension": [
      "novellia_1"
    ],
    "version": 1,
    "cbc34df5cb851e6flelelelelelelelelelelelelf2d4db94288b": {
      "IscaraTheTenThousandGuns": {
        "offchain": "ipfs://ipfs/QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
        "id": 2,
        "name": "Iscara the Ten Thousand Guns",
        "description": {
          "short": "Occulta Novellia Character",
          "long": "Occulta Novellia character play info such as stats and lore."
        },
        "tags": [
          "Game Character"
        ],
        "image": "ipfs://QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
        "resource": [
          {
            "resource_id": "Novellia",
            "description": "Off-chain Novellia extended metadata",
            "priority": 0,
            "multihash": "QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
            "hash_source_type": "ipfs",
            "url": [
              "sia://nAGazETmJJBvTnz5G50vt9gla_p6vctykk7uYy7N4cbbUQ",
              "ipfs://QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
              "https://rektangularstudios.com/novellia.json"
            ],
            "content_type": "application/json"
          },
          {
            "resource_id": "Artwork",
            "description": "Low Resolution",
            "priority": 0,
            "multihash": "QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
            "hash_source_type": "ipfs",
            "url": [
              "sia://nAGazETmJJBvTnz5G50vt9gla_p6vctykk7uYy7N4cbbUQ",
              "ipfs://QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
              "https://rektangularstudios.com/iscara_low.png"
            ],
            "content_type": "image/png"
          },
          {
            "resource_id": "Artwork",
            "description": "High Resolution",
```

```
          "priority": 1,
          "multihash": "QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
          "hash_source_type": "ipfs",
          "url": [
            "sia://nAGazETmJJBvTnz5G50vt9gla_p6vctykk7uYy7N4cbbUQ",
            "ipfs://QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
            "https://rektangularstudios.com/iscara_high.png"
          ],
          "content_type": "image/png"
        },
        "a"
      ],
      <arbitary-supported-metadata>,
      items: [
        {
          "phaser": "blue"
        },
      ],
      properties: ["diamondlike"],
      <other properties>
    },
  }
 }
}
```

	Extension	Description
root label	721	Indicates support for 721 feature set.
copyright	Extended 721	Attribution of rights.
publisher	Extended 721	List of publishers or entities involved in token creation. Useful for onlookers to determine token origin.
extension	Extended 721	List of extensions for parser to understand feature set.
version	Extended 721	Token version. May have future use such as for clients optionally using later metadata iterations, or reprints with updated metadata.
policy_id	721	Indicates policy_id to attribute metadata under. Useful when minting multiple tokens in one TX.
asset_id	721	Indicates asset_id to attribute metadata under. Useful when minting multiple tokens in one TX.
offchain	721	Indicates resource for offchain 721 metadata file. Clients need to support this for the complete 721 feature spec. This is optional even for 721, and should certainly not be provided for Novellia Standard Tokens.

id	721	Token number in a set. Redundant field which makes no sense for tokens without a total-order. **Count from 1 as this is user facing information and users are not developers. This is not part of the 721 specification, but makes semantic sense.**
name	721	Display name for token. **Fill for support on clients not supporting extensions.**
image	721	Image URL without source redundancy. • For tokens without an image resource, put your favorite logo. • **For tokens with an image resource, use the low resolution version. This is not part of the 721 specification, but will cause saner load times for legacy clients.**
description	Novellia	Short and long descriptions (64 byte limit per string) of token purpose. A short description makes a good header, while a long description makes a good paragraph body. **The long description supports Markdown, but this is useless except for the field as defined in the Novellia resource (due to character limits on-chain).** **My Product Name** You should buy my product because it's cool.
tags	Novellia	A list of string tags for the client to categorize / filter the token. If a token contains NSFW content, it should be marked with the "nsfw" tag.
resource	Novellia	List of token resources. For a complex token with many resources it is recommended to use an off-chain Novellia resource JSON file.

Resource

These will be loaded in the order given in the array. If there are multiple images a client should use the first supported image MIME type as the primary image to display.

If a Novellia resource is resolved all other resources will be ignored. If there are multiple Novellia resources only the first will be used.

Redundant resource URLs should be attempted in the order they are given. If you want to minimize load on your servers / centralized CDN, put decentralized storage first.

We have chosen the following attempt order:

- Sia / Skynet
  - It's decentralized, really fast, but experimental.
- IPFS
  - It's decentralized, but slower and more widely in use than Sia / Skynet.
- Centralized Storage
  - Decentralized storage cannot be relied on for storage permanency. This is last to prevent server load on the behalf of token publishers.

Field	Description
resource_id	Resource identifier interpreted by application. These may correspond to specific names resources a client expects.
description	Short text describing resource. Helpful to know what the resource should have been if the resource breaks. **This is not to be used by the client code to interpret what the resource is. That is, this field should not affect client logic in any way. This is for readability only. Use the resource_id and content_type fields for interpretation.**
priority	Order to load resource. Lower numbers load first. Resources with the same name will progressively overwrite each other. **This field is optional. When not supplied, it will default to 0. Note that multiple identical resource_id fields will lead to undefined behavior if they duplicate priorities.**
multihash	The multihash of the resource. In most cases, this may be identical to an IPFS hash. It is required to validate that contents served from centralized storage or unrecognized URL formats are unchanged. Clients should elect to check the multihash of downloaded files and be prepared to reject them. multiformats/multihash IPFS uses sha-256 as the default hash algorithm Content addressing

hash_source_type	Hint regarding file state prior to hashing. Options are:
	
	• `ipfs`
	  • IPFS does not directly multihash files. It first loads files into Unixfs, so will get a different result than directly applying multihash.
	• `direct`
	  • multihash is applied directly to resource file. this is more straightforward, but not robust to filenames, metadata changing, system specific data, etc.
	
	This field is optional. When not supplied, it will default to "ipfs".

url	Redundant resource URLs. These should all point to the same file on different hosts. **Since these are limited to 64 characters, you may want to consider a short naming scheme for static hosting, especially on a CDN. E.g. it might make sense to store filenames as hashes, not something long like https://rektangularstudios.com/cards/1/IscaraTheTenThousandGuns/novellia.json**
content_type	MIME type of resource. Common MIME types

For consistency, we define a few common **resource_id** choices:

- Novellia
  - Novellia off-chain resource
- Thumbnail
  - Small image to display on overview pages, cart icons for products, etc. This should not duplicate actual artwork (do LOD using **priority** field for low resolution versions of **Artwork** for example). This is mostly useful if you are making a non-image token (audio, e-book, etc.) that needs an image to display for UI purposes.
  - If this is not defined, the Novellia Dashboard will use the lowest **priority** resource in the following attempt order:
    - **Thumbnail**
    - **Card**
    - **Artwork**
- Card
  - Digital artwork simulating physical card
- Artwork
  - Relating to the token or card
- Animation
  - Relating to the token or card, but animated
- Soundtrack
  - Music to play
- OccultaNovelliaCharacter
  - A JSON file containing stats and moves (and some more) for an Occulta Novellia character
- Lore
  - Generic **resource_id** for indicating text-based data for tokens with a greater universe (fantasy, science fiction, etc.)
- 3DModel
  - A 3D model for a client to render. Specific file-type support, including animations, may vary depending on client. In the future, we will likely develop this further.
- Interactive
  - An interactive NFT. The specific format is not well-defined and may need to be inferred from the mimetype or the linked file itself. For example, a token similar to those of Stellar Hood may use this resource_id. Or check out an example use case on Novellia.

resource_id is case-insensitive. That is, clients should understand both "interactive" as well as "Interactive".

Clients should automatically enumerate and list resources with supported file formats, even if they do not understand the **resource_id**. For example, a client may ignore GIFs, but if it can display PNGs, it should put all of them into a carousel for display.

721 fields should be filled regardless of using any extensions:

- Ensures interoperability with clients not supporting said extensions
- Ensures some data integrity if extension resources fail to resolve
  - Novellia will overwrite **721** fields with the contents of the Novellia resource. This is helpful for character restricted fields such as **description**.

How to Get a Multihash

You may elect to not use IPFS to generate your multihash. It's probably the easiest method, but IPFS does a multi-step process which is non-trivial. You can't just run `sha256sum` and expect to get the same result as IPFS.

1. Option 1: Use IPFS directly
   - `ipfs add draculi_concept.png --only-hash --hash sha2-256`
2. Option 2: Use a reproducible process that includes multihash
   - Specification
     - File systems
     - multiformats/multihash
     - multiformats/multibase
   - Recommended implementations
     - https://github.com/ipfs/go-unixfs
     - https://github.com/multiformats/go-multihash

You cannot just use an arbitrary hashing algorithm. It will probably not make sense when a client tries to read it as a multihash.

Arbitrary Metadata

Many tokens have been minted with arbitrary metadata such as character properties, items, etc. This is useful for 1/1 mints, especially where additional properties, such as if a character has a hat, would be burdensome to make as separate JSON files off-chain.

For extensive token data, it is still recommended to make an off-chain JSON. This would be the case for cards in a TCG that may have extensive information.

Clients should display arbitrary objects under the asset_id (e.g. "IscaraTheTenThousandGuns") with clean rendering of arrays, specifically.

That is, an array like `"weapons": ["grenade", "rifle"]` could be displayed as a list like:

```
Weapons:
- grenade
- rifle
```

To be clear, these fields are very dynamic, and thus are not explicitly part of the standard. This is only a provision to display them with some semblance of correctness.

Novellia Resource

A Novellia resource is a JSON file linked as a resource within a token's metadata. This metadata must be using the Novellia extension for the **Extended 721 Token Standard**. It extends metadata off-chain, providing a few benefits:

- Smaller on-chain metadata means lower minting fees
  - For significant data, it may not even be possible due to transaction size limits
- Off-chain JSON does not have string length limitations

The Novellia resource:

- duplicates all Novellia extension metadata found on-chain
- overwrites on-chain metadata with same field names, taking priority due to relaxed off-chain restrictions

Multiple tokens **from a single policy_id** can use the same Novellia file in an array format like:

```
{
  "IscaraTheTenThousandGuns": {
    "name": "Iscara the Ten Thousand Guns",
    ...
  },
  "Draculi": {
    "name": "Draculi",
    ...
  },
  ...
}
```

An example of a **partially compliant** multi-token Novellia file is given here: https://ipfs.blockfrost.dev/ipfs/QmcbtQDhtvunqpAWaoVsGffo5SaUH5KA5gxfgYaj9MHivX

Below is an example of a full description of a single token:

```
{
  "novellia_version": 1,
  "metadata": {
    "copyright": "Copyright Rektangular Studios Inc.; all rights reserved",
    "publisher": [
      "https://rektangularstudios.com"
    ],
    "version": 1,
    "extension": [
      "novellia_1"
    ],
  },
  "token": {
    "policy_id": "cbc34df5cb851e6flelelelelelelelelelelelelf2d4db94288b",
    "asset_id": "IscaraTheTenThousandGuns"
  },
```

```
  "details": {
    "name": "Iscara the Ten Thousand Guns",
    "description": {
      "short": "Occulta Novellia Character",
      "long": "A character token for the surreal horror game Occulta Novellia. This token will grant access to a playable character."
    },
    "tags": [
      "Game Character"
    ],
    "commission": [
      {
        "name": "Rektangular Studios Inc.",
        "address": "addr1q8chzck9gkzd8t2v3477klsw3hna0s0er5vwspxehelaryfw08lffp5n2kzt72ez93m5zev2v4fm9sawnrqnvllmyhmsdjfzg9",
        "percent": "0.03"
      }
    ],
    "resource": [
      {
        "resource_id": "Card",
        "description": "Low Resolution",
        "priority": 0,
        "multihash": "QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
        "hash_source_type": "ipfs",
        "url": [
          "sia://nAGazETmJJBvTnz5G50vt9gla_p6vctykk7uYy7N4cbbUQ",
          "ipfs://QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
          "https://rektangularstudios.com/static"
        ],
        "content_type": "image/png"
      },
      {
        "resource_id": "Card",
        "description": "High Resolution",
        "priority": 1,
        "multihash": "QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
        "hash_source_type": "ipfs",
        "url": [
          "sia://nAGazETmJJBvTnz5G50vt9gla_p6vctykk7uYy7N4cbbUQ",
          "ipfs://QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
          "https://rektangularstudios.com/static"
        ],
        "content_type": "image/png"
      }
      {
        "resource_id": "Artwork",
        "description": "Low Resolution",
        "priority": 0,
        "multihash": "QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
        "hash_source_type": "ipfs",
        "url": [
          "sia://nAGazETmJJBvTnz5G50vt9gla_p6vctykk7uYy7N4cbbUQ",
          "ipfs://QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
```

```
        "https://rektangularstudios.com/static"
      ],
      "content_type": "image/png"
    },
    {
      "resource_id": "Artwork",
      "description": "High Resolution",
      "priority": 1,
      "multihash": "QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
      "hash_source_type": "ipfs",
      "url": [
        "sia://nAGazETmJJBvTnz5G50vt9gla_p6vctykk7uYy7N4cbbUQ",
        "ipfs://QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
        "https://rektangularstudios.com/static"
      ],
      "content_type": "image/png"
    },
    {
      "resource_id": "Animation",
      "description": "Animated GIF of the artwork creation process.",
      "priority": 0,
      "multihash": "QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
      "hash_source_type": "ipfs",
      "url": [
        "sia://nAGazETmJJBvTnz5G50vt9gla_p6vctykk7uYy7N4cbbUQ",
        "ipfs://QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
        "https://rektangularstudios.com/static"
      ],
      "content_type": "image/gif"
    },
    {
      "resource_id": "OccultaNovelliaCharacter",
      "description": "Occulta Novellia character play information such as stats and moves.",
      "priority": 0,
      "multihash": "QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
      "hash_source_type": "ipfs",
      "url": [
        "sia://nAGazETmJJBvTnz5G50vt9gla_p6vctykk7uYy7N4cbbUQ",
        "ipfs://QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
        "https://rektangularstudios.com/static"
      ],
      "content_type": "application/json"
    },
    <other properties>
  ]
 }
}
```

Field	Description
novellia_version	Identifies the JSON file as a Novellia resource and defines the version of the specification in use.
metadata	Metadata for the Novellia resource. Defines a few fields from the **721 Extended Token Standard**: • copyright • publisher • version • extension • it is not required to put **novellia_1** in here again. This will be ignored. Put additional extensions for proper parsing of the **details** field.
token	The token that the Novellia resource is intended for. This is helpful if someone finds the Novellia resource but isn't sure what it's being used by.
details	Any information that would have been filed under the specific token on-chain within the **Extended 721 Format**. Extensions other than **novellia_1** should put additional fields here to be parsed by Novellia Dashboard plugins. This may also include data additional to the on-chain metadata, withheld for reasons such as string length limitations. **Again, the long description supports markdown.**

Note that the Novellia off-chain resource does not support the short form description fields.

Commission

This field cannot be added to on-chain metadata due to the length of a Cardano address.

A list of addresses to send commissions or donations to. It is up to clients if they want to respect this when trading tokens. It is likely safer to use a mutable registry for these sorts of values, but we've included this field because:

- some tokens may not want to register to a mutable registry
- tokens registered to a mutable registry could have their commission address hijacked
- donation addresses seem like a very basic feature

Field	Description
name	Resource identifier interpreted by application. These may correspond to specific names resources a client expects.
address	Cardano address to deliver donations or commissions.
percent	Commission percent in [0,1]. E.g. 0.03 is a 3% commission.

Cyclic Hash Dependency

If you are uploading resources from a folder (e.g. on IPFS), take care that you cannot upload the Novellia resource in the same folder as other resources. This is because the Novellia resource requires the hashes of other resources, and changing the Novellia resource will change the hash of a resource folder.

This is a cyclic dependency.

Occulta Novellia Character Resource

An Occulta Novellia character resource is a JSON file that describes an Occulta Novellia character. This does not contain references to other resources like illustrations: it only contains the play information.

We provide this as an example of how the Novellia resource can act as an index for other resource files including game information. We neglect to describe the format as it does not pertain to the Novellia Token Standard.

```
{
  "occulta_novellia_version": 1,
  "name": "Iscara the Ten Thousand Guns",
  "card": {
    "number": 2,
    "release_set": "Presale 1",
  },
  "progression": {
    "class": "Iscara of Grevania",
    "stage": 1
  },
  "stats": {
    "health": 8,
    "attack": 6,
    "move": 1,
    "slots": 2
  },
  "attributes": [
    "Operator's Guild",
    "Human",
    "Tainted",
    "Rifle",
    "Sure Shot",
    "Operator",
    "Ten Thousand Guns"
  ],
  "description": "Rumors say the leader of the Operator's Guild is possessed by a demonic entity that lets her morph into monsters and spit in the face of death itself. The Grevans refuse to acknowledge her humanity, referring to her as Ten Thousand Guns. They hide their children at dawn, hoping that the pretty woman, usually seen in a dress, won't take them."
}
```

Security Concerns for Files Hosted on Centralized Storage

It is a requirement that all files listed under the token metadata and Novellia resource contain at multihash. It so happens that multihashes are used by IPFS. This choice is partly because:

- We expect IPFS to be more stable than Skynet
- IPFS has easier tooling than Sia for executing a dry run to get a hash
- A hash of a file hosted on a CDN must match or else the file can be rejected as malicious. This makes centralized hosting tamper proof.

Since decentralized storage is still experimental, we do not feel it is reasonable to exclude a link to centralized storage that a developer can trivially maintain.

Concerns over fair opportunity to decentralized storage

We use the default "ipfs" hash_source_type because it is a typical workflow. The use for multihash is standalone as a good format for encoding arbitrary hash types in a generalized field.

This standard, however, does not prefer IPFS, however. A key motivator for the multiple URL resource scheme is not only redundancy, but to avoid a tight coupling between CNFTs and IPFS, especially as better decentralized storage solutions may arise. We feel a tight-coupling like that, within a metadata standard, would be detrimental in the long-term.

Light-weight Novellia Standard

Where existing viewers such as pool.pm may display additional metadata in a visually poor manner, it may be preferable to use a simplified the Novellia Standard.

```
{
  "721": {
    "copyright": "Copyright Rektangular Studios Inc.; all rights reserved",
    "publisher": [
      "rektangularstudios.com"
    ],
    "extension": [
      "novellia_1"
    ],
    "version": 1,
    "cbc34df5cb851e6flelelelelelelelelelelelelf2d4db94288b": {
      "IscaraTheTenThousandGuns": {
        "offchain": "ipfs://ipfs/QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
        "id": 2,
        "name": "Iscara the Ten Thousand Guns",
        "description_short": "Occulta Novellia Character",
        "description_long": "Occulta Novellia character play info such as stats and lore.",
        "tags": [
          "Game Character"
        ],
        "image": "ipfs://QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
        "novellia_json": [
          "sia://nAGazETmJJBvTnz5G50vt9gla_p6vctykk7uYy7N4cbbUQ",
          "ipfs://QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
          "https://rektangularstudios.com/novellia.json"
        ],
        "novellia_multihash": "QmSXVm14pVec78vjGgQBMygwKYvDeyKyLs4UCic7UspxEv",
        "novellia_hash_source_type": "ipfs",
        <arbitary-supported-metadata>,
        items: [
          {
            "phaser": "blue"
          },
        ],
        properties: ["diamondlike"],
        <other properties>
      },
    }
  }
}
```

Field	Description
description_short	**Simplified form of "description" object. Supplying both the short form fields and "description" object will have undefined behavior.**
description_long	Short text describing resource. Helpful to know what the resource should have been if the resource breaks. **This is not to be used by the client code to interpret what the resource is. That is, this field should not affect client logic in any way. This is for readability only. Use the resource_id and content_type fields for interpretation.**
novellia_json	See `url`. This field is exclusively for the Novellia resource, and avoids needing an array of resource elements. The priority is implied to be `0`.
novellia_multihash	See `multihash`. This field is exclusively for the `novellia_json` resource.
novellia_source_hash_type	See `hash_source_type`. This field is exclusively for the `novellia_json` resource. **This field is again optional.**

Notice that there is no "novellia_priority" field. This is always implicitly 0 since LODs make no sense for a JSON file.

Changelog

1.0.0

- Initial public release

1.1.0

- Add support for "tags" in **novellia_1**
  - Define "nsfw" tag
- Add support for "offchain" in **721**
- Add more canonical **resource_id** kinds
  - Thumbnail
    - With highly specific use cases
  - Soundtrack
  - Lore
  - OccultaNovelliaCharacter
  - Rename ArtworkAnimation to Animation
- Warn about cyclic dependency for Novellia resource
- Pertaining to **resource**
  - Explicitly indicate **description** should not affect client logic
  - Explicitly indicate **priority** does not require client load all LODs for an asset
- Add **"Which fields are mandatory?"** section
- Remark that this standard does not refer to NFTs specifically

1.2.0

- Add "Arbitrary Metadata" section
  - support arbitrary arrays in metadata such as commonly used "items", "properties", "attributes" fields.
- Add "Light-weight Novellia Standard" section
  - Add "light-weight" alternative fields for compatibility with pool.pm
    - "description" object can be specified without nesting as
      - "description_short"
      - "description_long"
    - "resource" field can be specialized for singular Novellia resource
      - "novellia_json"
      - "novellia_multihash"

        "novellia_hash_source_type"
        - which is implicitly "ipfs" and need not be specified
- Implicit fields
  - "resource" elements may omit the following fields for their default values
    - "priority" has default 0
    - "hash_source_type" has default "ipfs"
- Add 3DModel resource_id

1.3.0

- resource_id is now case-insensitive
- Add "interactive" resource_id
- Novellia resource supports arrays similar to on-chain metadata.