Digital Goods

Namazu Elements Core Features · https://namazustudios.com/docs/namazu-elements-core/digital-goods/

Elements fully supports Digital Goods, which are items that can be owned by users in their inventory or purchased with In-App Purchases, such as potions or gold coins in a video game.

Elements supports traditional digital goods. Digital goods are items that can be owned by a user in their inventory. User inventories are independent of applications. Examples of digital goods may be consumable items such as potions or gold coins in a video game. Alternatively, they may be part of equipment sets such as swords, helmets, or weapon skins.

Digital goods are not limited to games and can be used to represent other content such as boosts in a dating app. Essentially, digital goods can be used to represent any type of content in an application that can be bought directly from your business.

The Digital Goods system in Elements manages three things:

- **Items** represent the core metadata of the items offered for sale.
- **Inventory** entries provide the association between a user and the Item they own.
- **Purchase Verification** ties In-App Purchases to specific Items and Quantities which Elements will deposit upon successful validation of purchases tied to IAP bundles

Note

Elements can verify purchases for Apple App Store and Google Play applications. First, you must set up Application Configurations for iOS or Android and then add the relevant info for your app to Elements to interact with the App Store or Google Play.

Items

Items are digital goods in the Elements database. These exist independent of Applications, so any application in your instance can interact with any item. Core APIs allow for the reading of items for sale.

These are the various fields that make up an item.

- **name:** This is a string representing the name of your item. It must be unique and have no spaces.
- **tags:** You may include a list of tags, as strings. Tags are an easy way to search for groups of items or sort them in your applications.
- **displayName:** This is a string representing your item's display name.
- **description:** This is a string that serves as a description for your item. Client code may opt to show this directly to the user.
- **metadata:** Metadata is optional. It can be any number of named strings or integers. This can have many uses, for example secondary descriptions, asset paths, types, or any arbitrary values assigned to your item.
- **Category:** Describes one of two particular inventory item values: Fungible and Distinct**.**
- **_id:** In the database, the item will also have an **_id** field. This is automatically generated when the item is created and serves as an internal reference for that item.

Note

See Managing Items below for more details about how to manage and modify items.

Inventory

Inventory is tied to each user, listing the items that user owns and the quantity. A user's inventory is accessible across all applications. Once an Item exists, you may use Cloud Functions to assign inventory items to users in the system. Additionally, you may use SUPERUSER APIs to award items to users for administrative purposes.

Fungible Inventory Item

Fungible inventory items exist as stacks in the user's inventory. Common examples for fungible items may be coins or potions in a game, credits for something else redeemable, or simply points accrued

for playing a game.

Fungible items exist at he USER level only and will be visible to all applications within Elements. See Application scoping rules for more details.

Distinct Inventory Item

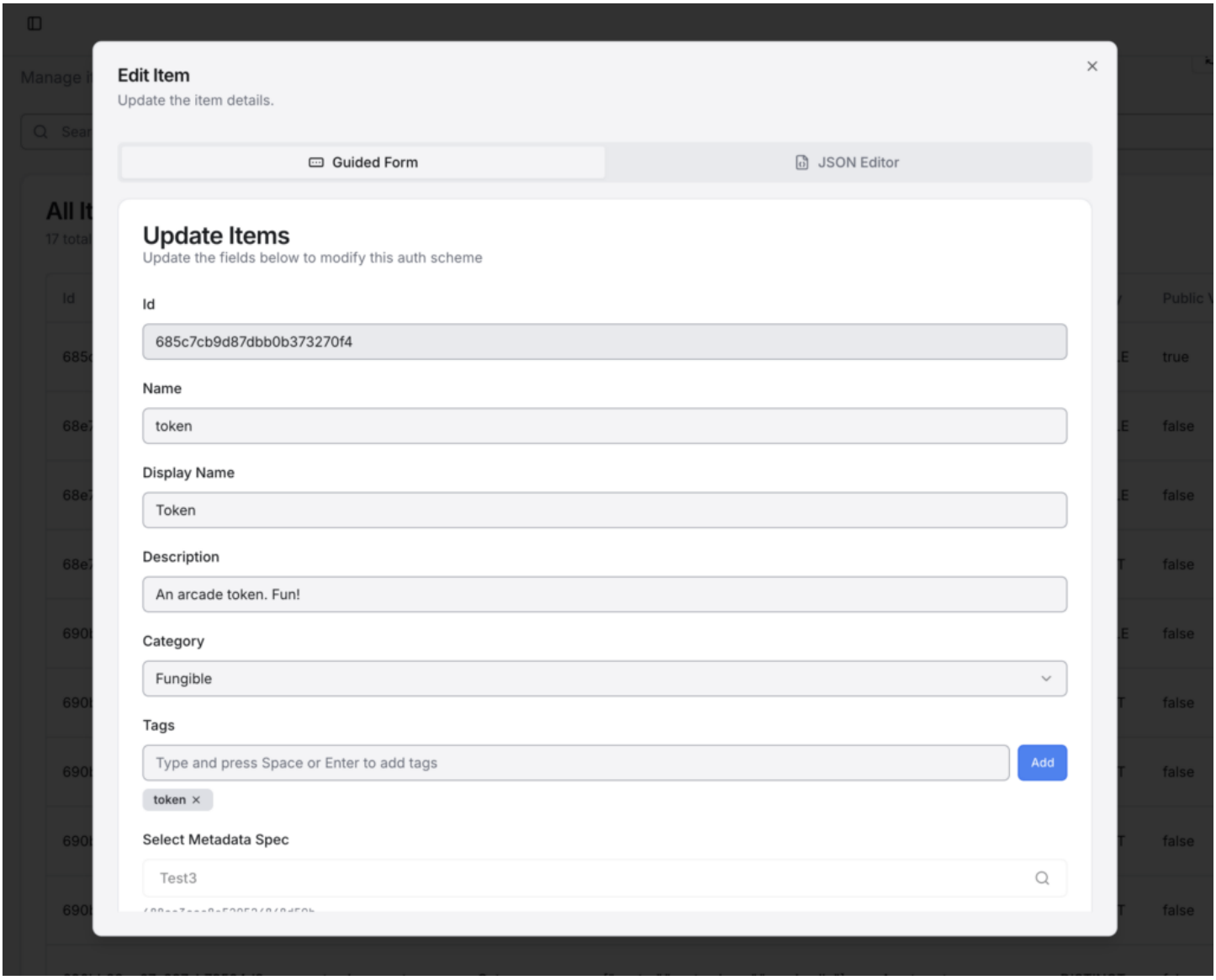
Distinct inventory items exist as single objects in the database. Unlike Fungible items, Distinct items may exist in multiples in the inventory. Additionally, they are not stackable. Distinct items would be useful to represent digital goods such as a sword, gun, or other in-game inventory item. In addition to the base Metadata, Distinct Inventory Items have their own metadata. This is useful, for example, if your game has degradation logic for particular items. For example, a sword can eventually wear out until it needs repair, or can gain experience as you defeat enemies with it.

Managing Items using the Console

Items are managed in the Digital Goods section of the admin console, which can be accessed from the upper nav bar or in the hamburger menu.

The "Add Item" button will open the new item panel.

Items can be edited by tapping the "Edit" button next to that item, or can be deleted by tapping the "Delete" button. Use the search function to more easily find specific items.



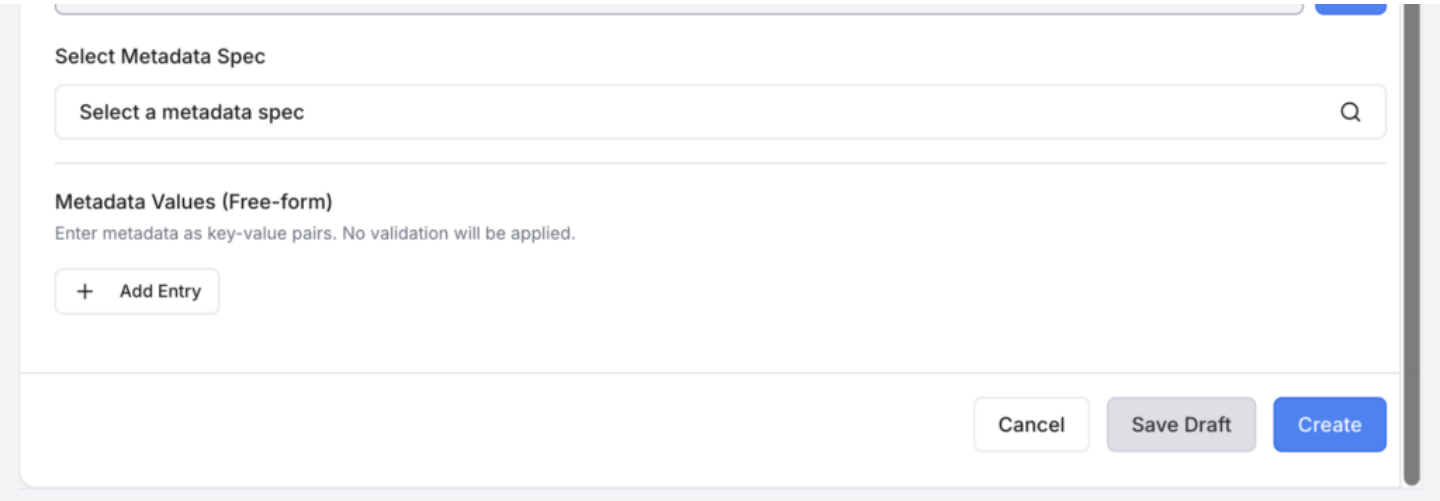
Item Editor

Editing Item Metadata

There are several to edit metadata on items. One is to add it free-form, adding key/value pairs directly. Simple click Add Entry and then add the values that you want.

The other way is to assign a Metadata Spec, which would have to be defined beforehand. Metadata Specs are a great way to pre-format what you want the metadata to look for for a set of items, and force the metadata to adhere to that structure.

You can also switch to the JSON Editor tab, where you can edit metadata directly in a JSON format.



JSON Structure of Items

This is a sample item represented in JSON:

```
[{
        "name": "name",
        "tags": ["tag1","tag2"],
        "displayName": "Display Name",
        "description": "This is a description.",
        "metadata": {
            "metadata_string": "This is a string",
            "metadata_int": 1
        }
}]
```