Nbt Crafting 3 Proposal

Siphalor

Contents

About

Nbt Crafting 3 is meant to revamp Nbt Crafting to be more maintanable, easier to use and better to extend.

Key aspects of this release will contain:

- Less ugly and unintuive syntax for **dynamic data** (esp. data merging)
- **Better ingredient matching**
- **Better extensibility** by exposing a lot of the internal APIs
- **Modularization**: Only rarely the full-blown Nbt Crafting v2 is needed and the overhead of unwanted functionality can be really huge

---

Still missing from this document and especially needed feedback:

- better ingredient matching
- changes to remainders?

Modules

- `nbtc-base`: Basic library utilities such as `NbtUtil`
- `nbtc-ingredient-types`: Ingredient types
- `nbtc-functions`: Functions
- `nbtc-remainders`: Allows to define remainders
- `nbtc-recipe-types`: Currently custom recipe types such as cauldron recipes:
  - *Move this to another mod?*
- ...

Ingredient types

If this makes the cut, this mod will no longer process ingredients preemptively but will look for a `type` property in ingredient objects.

It will look up the ingredient type in a registry and then run the specified deserializer.

```
{
    "ingredients": [
        {
            "type": "nbtcrafting:custom"
        }
    ]
}
```

Functions

Functions can be defined for recipe outputs and remainders. They iteratively transform the given stack in the given order.

They are extensibly defined in their own registry. For example KubeJS could hook into this and provide JavaScript expressions.

```
{
    "result": {
        "item": "hey",
        "nbtcrafting:functions": [
            {
                "type": "nbtcrafting:nbt_merge",
```

```
                "source": "i0"
            }
        ]
    }
}
```

nbtcrafting:nbt_set

This is equivalent to the current static `data` tag. This will *always* overwrite existing values at the given location.

- `value` specifies some kind of static nbt data to set
  - **or** `expression` specifies what's currently known as *dollar expression*.
- `target` optionally specifies the location, defaults to root element.
- `stringify` specifies whether the given value should be converted to a string. Useful for texts.

nbtcrafting:nbt_list_insert

Allows to add elements to lists. Currently only achievable through merges.

- Inherits all properties from `nbt_set`. `target` should point to some list(s).
- `index` optionally specifies the index to try to insert at.
  The index gets clamped at the list length. Defaults to the end of the list.

nbtcrafting:nbt_remove

Allows to remove element(s) from objects or lists.

- `target` specifies a path to some element(s)

nbtcrafting:nbt_merge

Would be the equivalent of the current *merge dollars*.

- `source` specifies the source ingredient.
- `target` specifies the location to copy the value to.
- `mode` allows to specify *merge modes* (previously *merge behaviors*).

nbtcrafting:count_set

Allows to set the count of the current stack. Will probably still be experimental because of how recipes work.

- `value` or `expression` as in `nbt_set`.

Better Path notation

The main goal of this redesign is to allow to target multiple elements.
This grants the chance to easily remove/change multiple values at once.

Thought: This could also allow to capture multiple values and concatenate them to a list.

Specification

Based on https://goessner.net/articles/JsonPath.

Operator	Name	Description
`.` or `[]`	child operator	Allows access to child elements
`..`	recursive descent	Matches all children in the current element and recurses further operators
`[,]`	array set	Allows to specify a list of indices to capture
`[(start):(end)(:step)]`	array slice	Allows to capture a consecutive number of array elements between `start` (inclusive) and `end` (exclusive). Optionally each *`step`-th* element is captured
`$`	root element	useful to capture all ingredients (e.g. `$..Damage`)

Note: regions in parenthesis are meant to be optional

Examples

Example	Explanation
`ingredient.Enchantments[0].id`	Gets the id of the first enchantment
`ingredient.Enchantments[:].id`	Targets all enchantment ids
`ingredient[Damage]` or `ingredient.Damage`	Gets the `Damage` value
`$..Damage`	Targets the `Damage` values of all ingredients

Hello, New World!

This section is meant to give an overview over the current state of this proposal and to visualize the suggested changes.

The example shapeless recipe consumes a diamond sword that has less than 40 damage and a diamond. It results in a diamond axe called “Battle Axe” which copies all the nbt data from the sword and additionally adds a sharpness enchantment at level 10.

The old world

```
{
    "type": "crafting_shapeless",
    "ingredients": [
        {
            "item": "minecraft:diamond_sword",
            "data": {
                "require": {
                    "Damage": "$..40"
                }
            }
        },
        { "item": "minecraft:diamond" }
    ],
    "result": {
        "item": "minecraft:diamond_axe",
        "data": {
            "display": {
                "Name": {
                    "text": "Battle Axe",
                    "nbtcrafting:strigify": true
                }
            },
            "Enchantments": [
                {
                    "id": "minecraft:sharpness",
                    "lvl": 10
                }
            ],
            "$": {
                "value": "i0",
                "paths": {
                    "/Enchantments\\[\\d+\\]/": "append"
                }
            }
```

```
        }
    }
}
```

The new world

```
{
    "type": "crafting_shapeless",
    "ingredients": [
        {
            "item": "minecraft:diamond_sword",
            "data": {
                "require": {
                    "Damage": "$..40"
                }
            }
        },
        { "item": "minecraft:diamond" }
    ],
    "result": {
        "item": "minecraft:diamond_axe",
        "nbtcrafting:functions": [
            {
                "type": "nbtcrafting:nbt_merge",
                "source": "i0"
            },
            {
                "type": "nbtcrafting:nbt_set",
                "target": "display.Name",
                "value": {
                    "text": "Battle Axe"
                },
                "stringify": true
            },
            {
                "type": "nbtcrafting:nbt_list_insert",
                "target": "Enchantments",
                "value": {
                    "id": "minecraft:sharpness",
                    "lvl": 10
                }
            }
        ]
    }
}
```