Item Tweaker Instructions

Contents

Introduction

This mod allows you to edit item properties by making a query to select a certain range of items and applying changes to items in that selection.

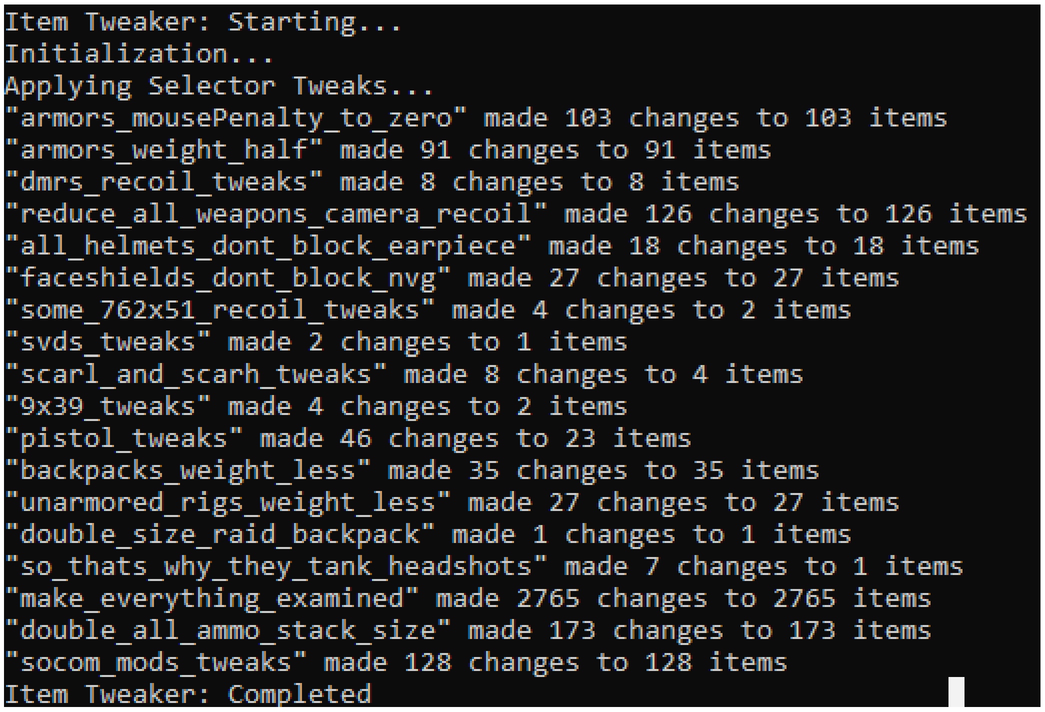
The config directory contains three JSON files which allow you to configure the mod – config, dynamic_selectors and manual_overwrite.

File ***config.json*** has an option to turn on/off verbose logging during mod's execution, which might be helpful to provide more detailed info to debug what is being done, but might flood the console with lots of lines.

File ***dynamic_selector.json*** contains selectors. Here you define which items you want to change and how.

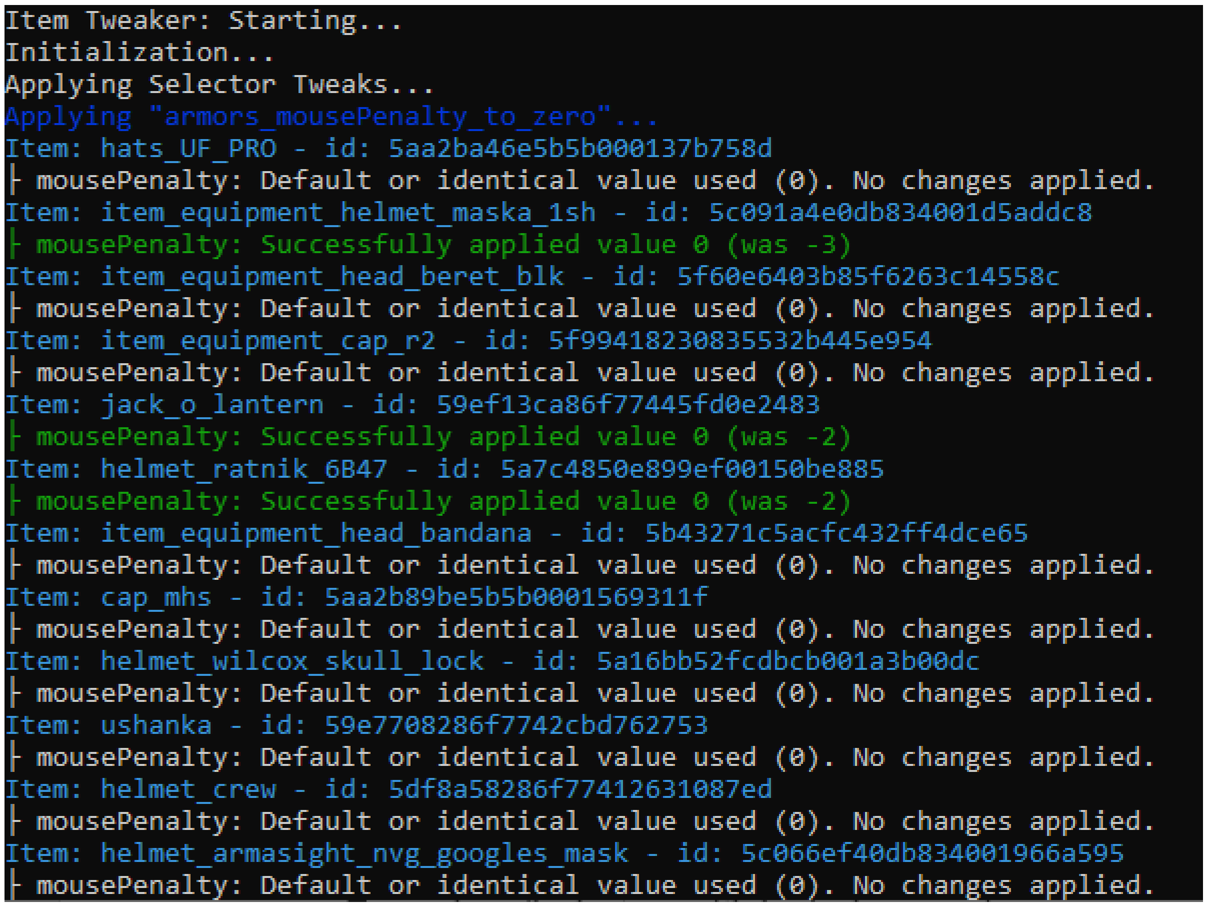
File ***manual_overwrite.json*** contains high priority changes to individual items determined by their name. High priority means that changes in this file will take priority over any changes in selectors if they affect the same item (i.e. if selector applies a multiplier of…WORK IN PROGRESS

A good way to see if everything works as you wanted it to is to run the mod without verbose logging and confirm that number of changes and items changed make sense.



```
Item Tweaker: Starting...
Initialization...
Applying Selector Tweaks...
"armors_mousePenalty_to_zero" made 103 changes to 103 items
"armors_weight_half" made 91 changes to 91 items
"dmrs_recoil_tweaks" made 8 changes to 8 items
"reduce_all_weapons_camera_recoil" made 126 changes to 126 items
"all_helmets_dont_block_earpiece" made 18 changes to 18 items
"faceshields_dont_block_nvg" made 27 changes to 27 items
"some_762x51_recoil_tweaks" made 4 changes to 2 items
"svds_tweaks" made 2 changes to 1 items
"scarl_and_scarh_tweaks" made 8 changes to 4 items
"9x39_tweaks" made 4 changes to 2 items
"pistol_tweaks" made 46 changes to 23 items
"backpacks_weight_less" made 35 changes to 35 items
"unarmored_rigs_weight_less" made 27 changes to 27 items
"double_size_raid_backpack" made 1 changes to 1 items
"so_thats_why_they_tank_headshots" made 7 changes to 1 items
"make_everything_examined" made 2765 changes to 2765 items
"double_all_ammo_stack_size" made 173 changes to 173 items
"socom_mods_tweaks" made 128 changes to 128 items
Item Tweaker: Completed
```

Verbose logging outputs much more detailed info about the whole process and outputs warnings and errors which aren't critical to the function of the mod but prevent the application of ceratin changes. In this example that would be if some items simply didn't have the "mousePenalty" property, not a big deal, the mod will simply skip that property change in the item. Whether these "skips" actually matter depends completely on what you want to achieve.



```
Item Tweaker: Starting...
Initialization...
Applying Selector Tweaks...
Applying "armors_mousePenalty_to_zero"...
Item: hats_UF_PRO - id: 5aa2ba46e5b5b000137b758d
├ mousePenalty: Default or identical value used (0). No changes applied.
Item: item_equipment_helmet_maska_1sh - id: 5c091a4e0db834001d5addc8
├ mousePenalty: Successfully applied value 0 (was -3)
Item: item_equipment_head_beret_blk - id: 5f60e6403b85f6263c14558c
├ mousePenalty: Default or identical value used (0). No changes applied.
Item: item_equipment_cap_r2 - id: 5f99418230835532b445e954
├ mousePenalty: Default or identical value used (0). No changes applied.
Item: jack_o_lantern - id: 59ef13ca86f77445fd0e2483
├ mousePenalty: Successfully applied value 0 (was -2)
Item: helmet_ratnik_6B47 - id: 5a7c4850e899ef00150be885
├ mousePenalty: Successfully applied value 0 (was -2)
Item: item_equipment_head_bandana - id: 5b43271c5acfc432ff4dce65
├ mousePenalty: Default or identical value used (0). No changes applied.
Item: cap_mhs - id: 5aa2b89be5b5b0001569311f
├ mousePenalty: Default or identical value used (0). No changes applied.
Item: helmet_wilcox_skull_lock - id: 5a16bb52fcdbcb001a3b00dc
├ mousePenalty: Default or identical value used (0). No changes applied.
Item: ushanka - id: 59e7708286f7742cbd762753
├ mousePenalty: Default or identical value used (0). No changes applied.
Item: helmet_crew - id: 5df8a58286f77412631087ed
├ mousePenalty: Default or identical value used (0). No changes applied.
Item: helmet_armasight_nvg_googles_mask - id: 5c066ef40db834001966a595
├ mousePenalty: Default or identical value used (0). No changes applied.
```

Selectors

When defining selectors you need to follow a proper structure. Here is the structure of a selector with a simple query.

```
"any_selector_name": {
    "query": {
        "key": "item_property_key",
        "operation": "greater_than"/"less_than"/"equals"/"starts_with"/"contains"/"ends_with",
        "values": [
            "value1",
            "value2",
            ...other values
        ],
        "negation": true/false,
        "strict": true/false
    },
    "multiply": {
        "propertyKey": 69, (any number)
         ...other changes
    }
    "set": {
        "propertyKey": 69, (any value of the same type as target property)
         ...other changes
    }
}
```

Lets go over each line in detail:

- **"any_selector_name"** – in this key you simply define **selector's name** which you desire.
- **"query"** – define **a query with conditions** to select items. In this structure example a basic expression with one condition and a set of values is provided, later a more complex query structure will be shown.
- **"key"** – define a property key (**name of the property**) which should be compared.

- **"operation"** – defines the comparison **condition to filter items**. Can have the following values: *"greater_than"*, *"less_than"*, *"equals"*, *"starts_with"*, *"contains"*, *"ends_with"*. Can be skipped and if it is – by default uses *"equals"* condition.
- **"values" –** has to be **an array of values** which are **used with "operation"**.
- **"negation" –** simply **a logical *"not"* operator**. Can be *true* or *false*. Used to invert the condition. Can be skipped, by default has value *false*.
- **"strict" –** determines whether condition should be true for every (logical *"and")* or at least one (logical *"or")* of the provided values. Can be *true (every)* or *false (at least one)*. E.g. all . Can be skipped, by default has value *false*.
- **"multiply" –** contains **property names and their multipliers**. Can only be applied to item properties of type "number". Can be skipped if you don't want to multiply anything.
- **"set" –** contains **property names and values you want to set**. New value type has to correspond to target property type. Can be skipped if you don't want to set anything.

Note: Values in the "set" object are applied after the multipliers, so if they affect the same properties, multiplier changes will be overwriten with set values.

Note: While unlikely for most use cases, selectors might intersect if they edit the identical properties in identical items. Changes will still be applied but you will be informed to expect overlapping changes. Such conflicts can be resovled by manually applying changes to conflicting items individually in the ***manual_overwrite.json***. Later I might introduce a priority system for selectors.

Examples of selectors with basic expressions

A selector which tweaks pistols. Looks for items where “weapClass” property has value “pistol”. Multipliers are set to 0.15, therefore vertical and horizontal recoil are reduced by 85%.

```
    "pistol_tweaks": {
        "query": {
            "key": "weapClass",
            "values": [
                "pistol"
            ]
        },
        "multiply": {
            "RecoilForceUp": 0.15,
            "RecoilForceBack": 0.15
        }
    }
```

A selector, which removes mouse sensitivity, changes from any armor or clothing. Looks for items where “ArmorType” property has value “Light” or “Heavy”, or “None”. Simply sets the “mousePenalty” to 0.

```
    "armors_mousePenalty_to_zero": {
        "query": {
            "key": "ArmorType",
            "values": [
                "Light",
                "Heavy",
                "None"
            ]
        },
        "set": {
            "mousePenalty": 0
        }
    }
```

Note: Compared to filtering by “_parent”, filtering by all possible values of “ArmorType” is a good way to capture all items, which might have “mousePenalty”, because this property can be present in any wearable item (helmets, armors, faceshields, aventails, ear armor, helmet plates) have different parents.

A selector which doubles ammo stack sizes. Looks for items with ammo parent id. Multiplies the property "StackMaxSize" by 2. If you play around you can also make other items stackable, by default the ones which don't stack have a "StackMaxSize" of 1.

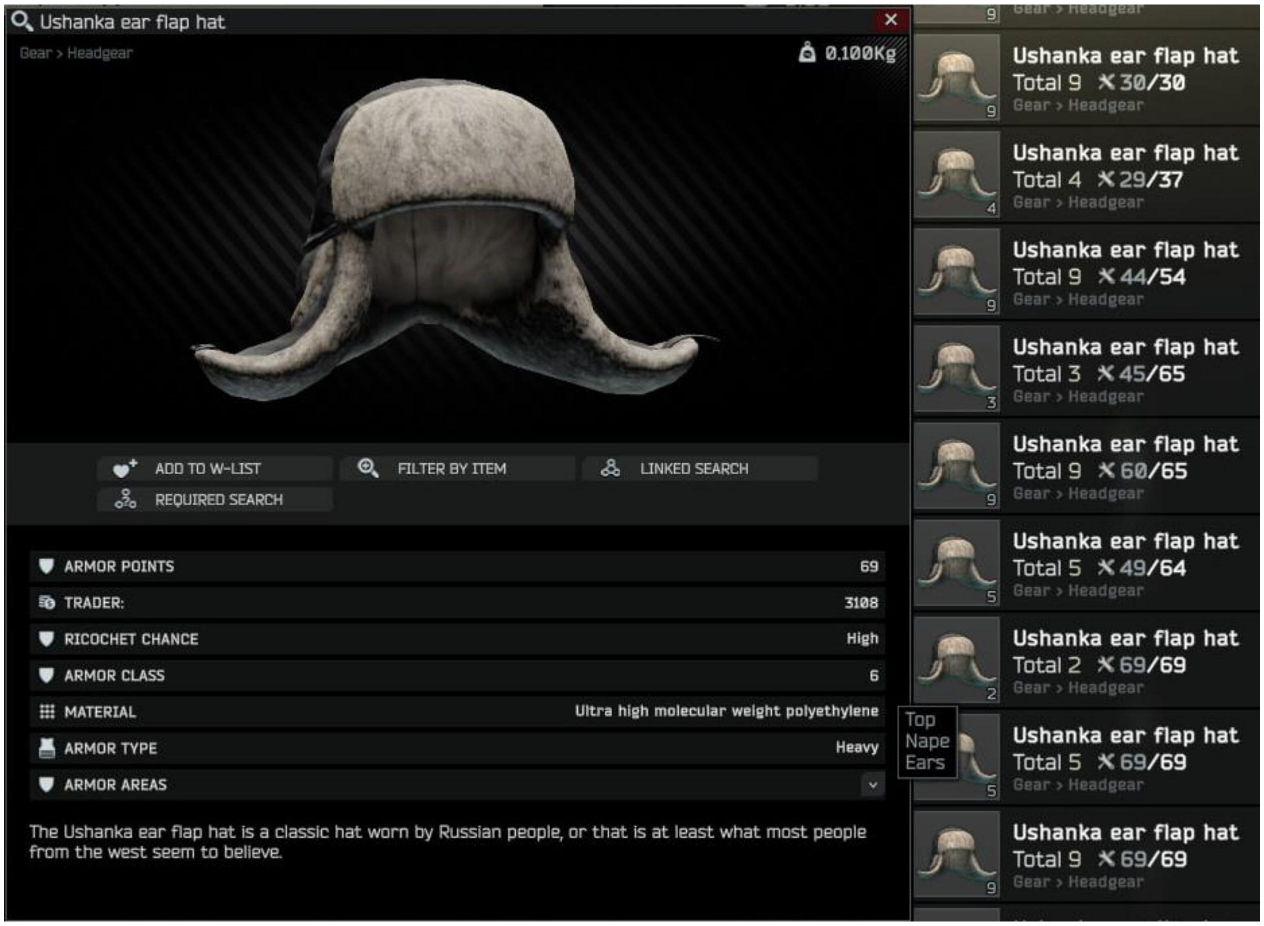
```
"double_all_ammo_stack_size": {
        "query": {
            "key": "_parent",
            "values": [
                "5485a8684bdc2da71d8b4567"
            ]
        },
        "multiply": {
            "StackMaxSize": 2
        }
    }
```

A selector which makes everything examined by default. A good way to select all items is to simply filted by "_type" property and look for value "Item".

```
"make_everything_examined": {
        "query": {
            "key": "_type",
            "values": [
                "Item"
            ]
        },
        "set": {
            "ExaminedByDefault": true
        }
    }
```

A selector which sort of makes sense. Looks for Ushanka's id. Gives it appropriate protection qualities so that scavs have a real basis to tank headshots.

```
    "so_thats_why_they_tank_headshots": {
        "query": {
            "key": "_id",
            "values": [
                "59e7708286f7742cbd762753"
            ]
        },
        "set": {
            "Durability": 69,
            "MaxDurability": 69,
            "armorClass": "6",
            "armorZone": ["Head"],
            "headSegments": ["Top", "Nape", "Ears"],
            "RicochetParams": {
                "x": 0.9,
                "y": 0.4,
                "z": 65
            },
            "ArmorMaterial": "UHMWPE",
            "ArmorType": "Heavy"
        }
    }
```


Ushanka ear flap hat

Gear > Headgear

0.100Kg

ARMOR POINTS	69
TRADER:	3108
RICOCHET CHANCE	High
ARMOR CLASS	6
MATERIAL	Ultra high molecular weight polyethylene
ARMOR TYPE	Heavy
ARMOR AREAS	

Top
Nape
Ears

The Ushanka ear flap hat is a classic hat worn by Russian people, or that is at least what most people from the west seem to believe.

Ushanka ear flap hat
Total 9 30/30
Gear > Headgear

Ushanka ear flap hat
Total 4 29/37
Gear > Headgear

Ushanka ear flap hat
Total 9 44/54
Gear > Headgear

Ushanka ear flap hat
Total 3 45/65
Gear > Headgear

Ushanka ear flap hat
Total 9 60/65
Gear > Headgear

Ushanka ear flap hat
Total 5 49/64
Gear > Headgear

Ushanka ear flap hat
Total 2 69/69
Gear > Headgear

Ushanka ear flap hat
Total 5 69/69
Gear > Headgear

Ushanka ear flap hat
Total 9 69/69
Gear > Headgear

Selectors with logical expressions

If you want to filter items with more than one condition you can use a logical expression query to combine them. Make sure the structure is as follows:

```
    "scarl_and_scarh_tweaks": {
        "query": {
            "condition": "and"/"or",
            "expressions": [
                {
                    "key": "item_property_key",
                    "operation": "greater_than"/"less_than"/"equals"/"starts_with"/"contains"...
                    "values": [
                        "value1",
                        "value2",
                        ...other values
                    ],
                    "negation": true/false,
                    "strict": true/false
                },
                ...other expressions
            ]
            "negation": true/false,
        },
        "multiply": {
            "propertyKey": 69, (any number)
             ...other changes
        }
        "set": {
            "propertyKey": 69, (any value of the same type as target property)
             ...other changes
        }
    }
```

There are three properties in a logical expression:

- **"condition"** determines whether all (and) or at least one (or) expression has to be true.
- **"expressions"** property can contain multiple other expressions. Can contain both basic expressions and logical expressions.

- **"negation"** property inverts the result, same as in the basic query. Can be skipped, by default has value *false*.

Query evaluation is recursive, so you can build a query tree of needed depth and make your query as precise as you need. For most use cases one or several basic expressions suffice but that completely depends on what you want to do.

Examples of selectors with logical expressions

A selector which applies tweaks to SCAR-L and SCAR-H. Looks for items which have a weapon parent id (so we know they are weapons) AND their "_name" contains string "weapon_fn_mk". This way we affect 4 items – both FDE and Black versions of Mk16 (SCAR-L) and Mk17 (SCAR-H).

If you, for example, wanted to affect only L or H version, you'd just have to check if "_name" contains "weapon_fn_mk16" or "weapon_fn_mk17".

Note: In the same way you can tweak other item groups. For example Zenit grips. Make sure that items has grip parent id and the "_name" contains "zenit".

```
    "scarl_and_scarh_tweaks": {
        "query": {
            "condition": "and",
            "expressions": [
                {
                    "key": "_parent",
                    "values": [
                        "5447b5f14bdc2d61278b4567"
                    ]
                },
                {
                    "key": "_name",
                    "operation": "contains",
                    "values": [
                        "weapon_fn_mk"
                    ]
                }
            ]
        },
        "multiply": {
            "RecoilForceUp": 0.65,
            "RecoilForceBack": 0.65
        }
    },
```

Selector which tweaks unarmored rigs. A pretty simple tweak that affects rigs which offer no protection. Look for items with tactical rig parent id AND “ArmorType” of “None”. Multiplier simply halves the weight.

```
"unarmored_rigs_weight_less": {
        "query": {
            "condition": "and",
            "expressions": [
                {
                    "key": "_parent",
                    "values": [
                        "5448e5284bdc2dcb718b4567"
                    ]
                },
                {
                    "key": "ArmorType",
                    "values": [
                        "None"
                    ]
                }
            ]
        },
        "multiply": {
            "Weight": 0.5
        }
    },
```

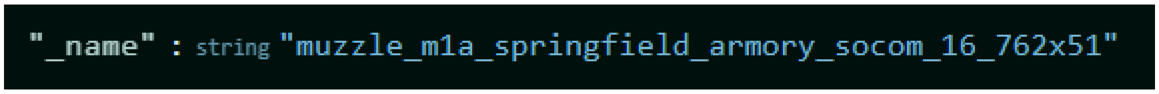
Recursive property access

I

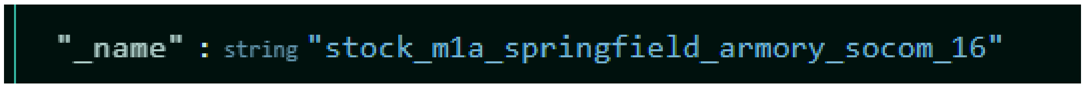
Manual Overwrite

S

A selector which tweaks the SOCOM 16 build of M1A. Lets say you are satisfied with M1A’s stats in a meta build, but you wish to make the SOCOM build more viable. Perhaps a good way to filter items is to check if “_name” contains “socom_16”.



```
"_name" : string "muzzle_m1a_springfield_armory_socom_16_762x51"
```



```
"_name" : string "stock_m1a_springfield_armory_socom_16"
```

S

S

S