

Einleitung

In diesem Dokument werden die Funktionen zur Anpassung der Beute in SCUM beschrieben. Die Anpassung der Beute erfolgt durch die Erstellung und Änderung verschiedener JSON-Dateien. Vertrautheit mit JSON-Grundlagen ist Voraussetzung, wenn Sie also noch nie von JSON gehört haben oder einfach nur Ihr Wissen auffrischen möchten, finden Sie hier einige nützliche Links:

- <https://en.wikipedia.org/wiki/JSON>
- https://www.w3schools.com/js/js_json_intro.asp
- <https://www.json.org/>

Die meisten JSON-Dateien können über verschiedene Admin-Befehle aus dem Spiel exportiert werden, während einige von ihnen manuell erstellt werden müssen. Der Stammpfad, in dem sich die JSON-Dateien befinden, lautet:

- **<Server>\SCUM\Saved\Config\WindowsServer\Loot** für Multiplayer-Server.
- **%LocalAppData%\SCUM\Saved\Config\WindowsNoEditor\Loot** für den Einzelspieler.

JSON-Dateien werden beim Start des Spiels geladen und alle nachfolgenden Änderungen können über #ReloadLootCustomizationsAndResetSpawners Befehl erneut geladen werden , um Änderungen schnell zu testen. Der erwähnte Befehl wird später ausführlich beschrieben, ebenso wie der Prozess des Exportierens, Erstellens und Änderns von JSON-Dateien, die sich auf Beute beziehen.

Bevor wir tiefer in die Materie eintauchen und uns mit Befehlen und Beutemodifikationen befassen, werden wir einige Konzepte und Voraussetzungen behandeln, die sich für eine effektive Beuteanpassung als unerlässlich erweisen würden.

Wenn Sie sofort loslegen möchten, springen Sie zum [Abschnitt](#) Exportieren.

Seltenheit

Seltenheit wird bei der Anpassung der Beute ausgiebig eingesetzt. Er bestimmt die Wahrscheinlichkeit, dass ein Objekt aus der Menge der Objekte ausgewählt wird. Objekt ist ein abstrakter Begriff und kann alles Mögliche bedeuten. Sie können sich ein Objekt vorerst als einen Gegenstand vorstellen. Jedem Objekt im Set wird eine Seltenheit zugewiesen, die einer der folgenden Werte sein kann:

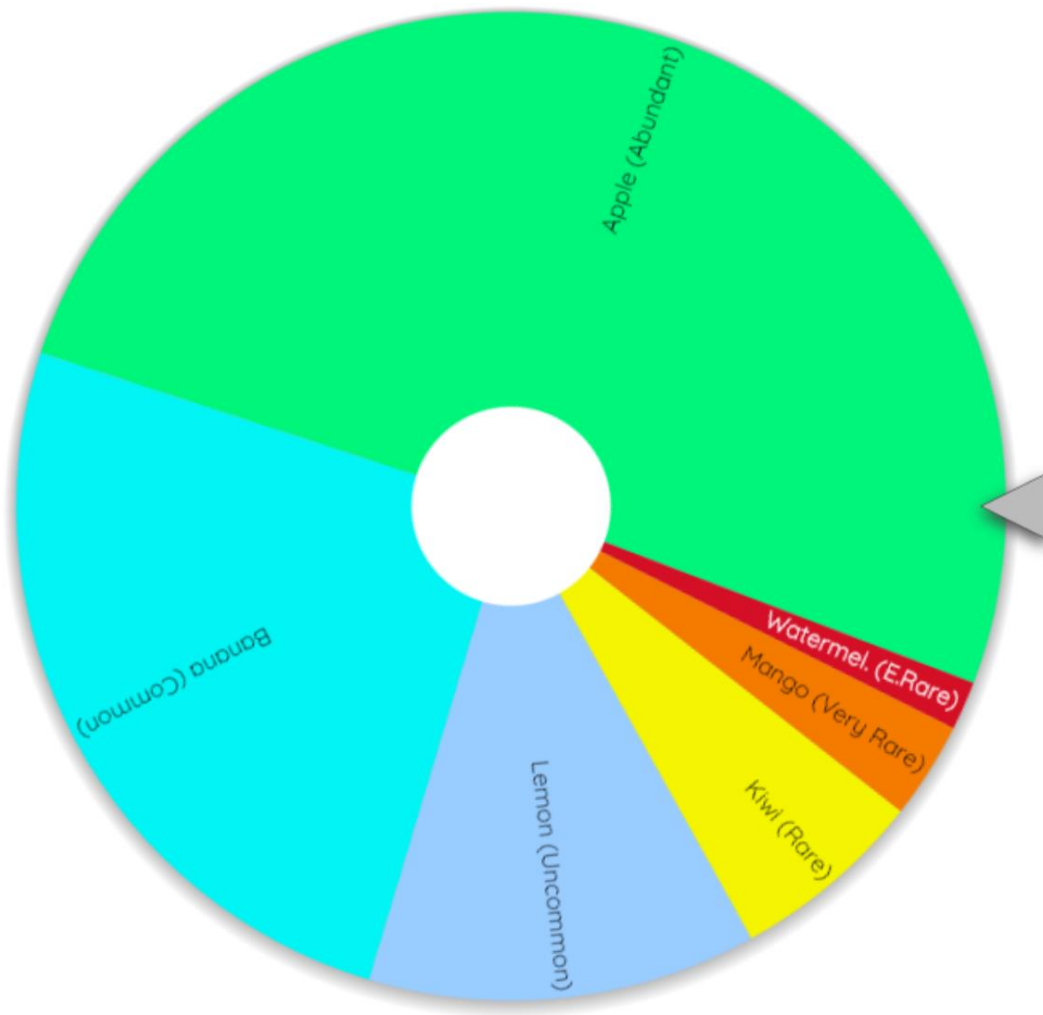
- **Häufig**: 32-mal wahrscheinlicher, ausgewählt zu werden als **Extrem Selten**
- **Häufig**: 16-mal wahrscheinlicher ausgewählt als **extrem selten**
- **Ungewöhnlich**: 8-mal wahrscheinlicher ausgewählt als **extrem selten**
- **Selten**: 4-mal wahrscheinlicher ausgewählt als **extrem selten**
- **Sehr selten**: 2-mal wahrscheinlicher ausgewählt als **extrem selten**
- **Extrem selten**: Die gleiche Wahrscheinlichkeit, ausgewählt zu werden, wie jedes andere **extrem seltene** Objekt im Set

Nehmen wir zum Beispiel an, Sie haben den folgenden Satz von Objekten (Objekte sind in diesem Beispiel Elemente):

{ Apfel (häufig), Banane (häufig), Zitrone (selten), Kiwi (selten), Mango (sehr selten), Wassermelone (extrem selten) }

Die Wahrscheinlichkeit, dass Apfel ausgewählt wird, ist 2-mal höher als bei Banane, die Wahrscheinlichkeit, ausgewählt zu werden, ist 4-mal höher als bei der Zitrone, die Wahrscheinlichkeit, ausgewählt zu werden, ist 8-mal höher als bei der Kiwi, die Wahrscheinlichkeit, ausgewählt zu werden, ist 16-mal höher als bei der Mango und die Wahrscheinlichkeit, ausgewählt zu werden, ist 32-mal höher als bei der Wassermelone. Auf der anderen Seite ist die Wahrscheinlichkeit, dass Banane ausgewählt wird, 2-mal geringer als bei Apfel, 2-mal wahrscheinlicher als bei Zitrone, 4-mal wahrscheinlicher als Kiwi, 8-mal wahrscheinlicher als Mango und 16-mal wahrscheinlicher als Wassermelone. Die Wahrscheinlichkeit einer anderen Fruchtauswahl im Vergleich zu den übrigen Früchten kann auf ähnliche Weise abgeleitet werden.

Eine andere Möglichkeit, Raritäten zu betrachten, ist ein Roulette-Rad, bei dem die Fläche, die das Objekt einnimmt, die Wahrscheinlichkeit bestimmt, dass es ausgewählt wird. Für das obige Beispiel sieht das Roulette-Rad wie folgt aus:

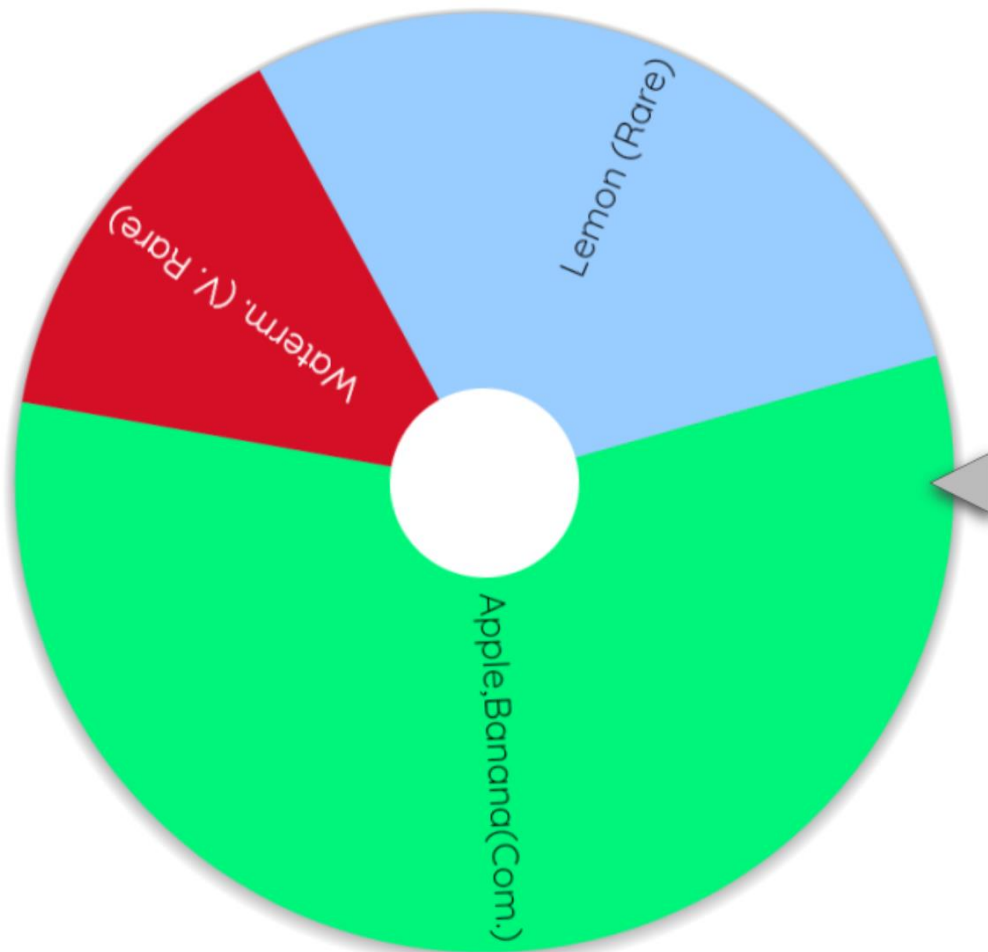


Wie Sie sehen können, nimmt Apple die meiste Fläche ein, so dass es am wahrscheinlichsten ausgewählt wird, im Gegensatz zu Wassermelone, die die geringste Fläche einnimmt und daher am wenigsten wahrscheinlich ausgewählt wird. Andere Früchte liegen dazwischen.

Schauen wir uns ein weiteres Beispiel an:

{ Apfel|Gewöhnlich, Banane|Gewöhnlich, Zitrone|Selten, Wassermelone|Sehr selten }

Diesmal "fehlen" einige Raritäten und wir haben zwei Früchte mit der gleichen Seltenheit. Objekte mit der gleichen Seltenheit werden gruppiert. Um das ausgewählte Objekt in der Gruppe zu bestimmen, wird ein weiterer Würfel geworfen, wobei die Wahrscheinlichkeit für alle Objekte mit der gleichen Seltenheit gleich hoch ist. Schauen wir uns das Roulette-Rad für das obige Beispiel an:



In diesem Fall ist die Wahrscheinlichkeit, dass Apfel oder Banane ausgewählt werden, 4-mal höher als bei Zitrone und 8-mal höher als bei Wassermelone. Im obigen Roulette-Rad werden zuerst Apfel oder Banane ausgewählt und dann werden erneut gewürfelt, um entweder Apfel oder Banane auszuwählen, wobei jeder von ihnen eine 50%ige Chance hat, ausgewählt zu werden.

Zu wissen, wie die Objektauswahl anhand von Seltenheiten funktioniert, ist wichtig, da sie im Rest des Dokuments für die Elementauswahl, die Knotenauswahl und die Auswahl von Untereinstellungen verwendet wird. Machen Sie sich keine Sorgen, wenn Sie nicht wissen, was Knoten und Supresets bedeuten. Sie werden in den folgenden Abschnitten erläutert.

Beutebaum-Knoten

Überblick

Bevor wir erklären, was Beutebaumknoten sind, wollen wir uns mit ihrer Motivation befassen. Nehmen wir an, wir haben eine Leiste, die die folgenden Gegenstände spawnen kann:

- Bier
- Absinth
- Pommes frites
- Haselnüsse
- 1H_KitchenKnife

Wir müssen den Gegenständen irgendwie Wahrscheinlichkeiten zuweisen, weil es keinen Sinn macht, dass z.B. Haselnüsse die gleiche Wahrscheinlichkeit haben, zu spawnen wie Bier. Zumindest nicht in einer Bar. Wir haben bereits Seltenheiten, um Wahrscheinlichkeiten zu spezifizieren, also versuchen wir das:

- Bier (reichlich)
- Absinth (sehr selten)
- Chips (Ungewöhnlich)
- Haselnüsse (selten)
- 1H_KitchenKnife (extrem selten)

Das Problem bei diesem Ansatz ist, dass diese Wahrscheinlichkeiten in einer Bar sinnvoll sind, aber woanders nicht unbedingt Sinn machen würden. Wenn wir zum Beispiel Gegenstände in einer Wohnküche spawnen müssen, haben 1H_KitchenKnife eine größere Wahrscheinlichkeit, gefunden zu werden als Haselnüsse oder Absinth. Wenn wir es mit einer sehr großen Anzahl von Gegenständen zu tun haben, bieten Seltenheiten einfach nicht genug Granularität. Eine Lösung könnte darin bestehen, Zahlen für Wahrscheinlichkeiten anstelle von Seltenheiten zu verwenden, aber dies könnte sich als umständlich für die Optimierung und Wartung erweisen.

Wir haben uns diesem Problem genähert, indem wir Gegenstände und ihre Seltenheiten in einem Baum kategorisiert haben. Jeder Baumknoten hat einen Namen und eine Seltenheit. Der übergeordnete Knoten stellt eine Reihe von Elementen dar, die einige gemeinsame Merkmale aufweisen. Der übergeordnete Knoten könnte z. B. ein Balken sein, der Elemente enthält, die in einem Balken sinnvoll sind. Darüber hinaus könnte der Bar-Knoten andere Taschen wie Getränke, Essen und Waffen enthalten, die Gegenstände enthalten würden, die für diese Taschen sinnvoll sind. Lassen Sie uns 5 der oben genannten Elemente in einen Baum einfügen:

- Stab
 - Getränke
 - Bier
 - Absinth
 - Essen
 - Pommes frites

- Haselnüsse
- Waffen
 - 1H_KitchenKnife

Der erste Schritt bestand also darin, die Elemente in einem Baum zu organisieren. Der nächste Schritt ist die Zuweisung von Raritäten. Die Seltenheit jedes Knotens ist relativ zu den Seltenheiten anderer Geschwisterknoten. Da es sich um eine Bar handelt, sollten Getränke häufiger sein als Essen und viel häufiger als Waffen. Außerdem sollte Bier häufiger als Absinth und Chips häufiger als Haselnüsse verwendet werden. Wir könnten Raritäten wie diese zuordnen:

- Stab
 - Getränke (reichlich)
 - Bier (reichlich)
 - Absinth (selten)
 - Essen (Ungewöhnlich)
 - Chips (reichlich)
 - Haselnüsse (gebräuchlich)
 - Waffen (sehr selten)
 - 1H_KitchenKnife

Beachten Sie, dass Bar- und 1H_KitchenKnife-Knoten keine Seltenheitsgrade zugewiesen sind. Leiste, weil es ein Root ist, und 1H_KitchenKnife, weil es keine Geschwister hat, also ist es die einzige Wahl innerhalb des Waffenknotens.

Die verbleibende Frage ist, wie ein Element aus einem Baum wie dem oben genannten ausgewählt wird. Die Antwort lautet von links nach rechts oder in der Baumterminologie, vom Stammknoten zum Blattknoten. Im obigen Beispiel ist der Bar-Knoten der Stammknoten, sodass die Auswahl dort beginnt. Das nächste, was Sie auswählen müssen, ist der untergeordnete Knoten von Bar. Der Barknoten hat drei untergeordnete Elemente: Getränke (reichlich), Nahrung (ungewöhnlich) und Waffen (sehr selten). Die Auswahl der untergeordneten Knoten erfolgt nach Seltenheit, wie im [Abschnitt "Seltenheit" erläutert](#) . Nehmen wir an, ein Food-Knoten ist ausgewählt. Nun wird der Prozess fortgesetzt, bis der Blattknoten erreicht ist. Der Nahrungsknoten hat zwei untergeordnete Elemente: Chips (reichlich) und Haselnüsse (häufig). Die Auswahl erfolgt nach Seltenheit und nehmen wir an, der Chips-Knoten wird ausgewählt. Der Chips-Knoten ist ein Blattknoten und die Auswahl endet dort. Der Spieler würde einen Chips-Gegenstand spawnen.

Das Gleiche gilt auch für die größten Bäume. Die Selektion beginnt mit der Wurzel, wählt die untergeordneten Elemente nach Seltenheit aus und geht weiter, bis der Blattknoten erreicht ist. Blattknoten sind immer Elemente. Knoten, die untergeordnete Knoten enthalten, sind immer Säcke mit Elementen.

Wir nennen diese Art von Baum einen Beutebaum und Knoten innerhalb eines solchen Baumes einen Beutebaum Nodes (oder kurz Nodes).

Knoten-IDs

Damit Knoten irgendwo verwendet werden können, müssen sie irgendwie referenziert werden. Nehmen wir zum Beispiel an, wir haben drei Standorte in der Leiste:

- Schublade mit Barküchenwerkzeugen
- Kühlschrank mit verschiedenen Getränken
- Stehtische, auf denen entweder Getränke oder Speisen stehen können

Wir verwenden Knoten aus dem vorherigen Abschnitt, um zuzuweisen, welche Gegenstände an welchem Ort gespawnt werden können:

- Die Schublade erhält den Waffenknoten
- Kühlschrank erhält Getränke-Knoten
- Tabellen erhalten Getränke- und Essensknoten

Im obigen Beispiel haben wir Knotennamen verwendet, um anzugeben, welche Knoten Elemente für welchen Speicherort bereitstellen. Das ist alles in Ordnung für den Baum, der so einfach ist wie im Beispiel, aber das Problem ist, dass der eigentliche Beutebaum, der im Spiel verwendet wird, riesig ist und die Knotennamen nicht eindeutig sein müssen. Zum Beispiel könnten wir einen Knoten namens Waffen sowohl in der Bar als auch in der Polizeistation haben. Wir müssen eine genauere Methode haben, um anzugeben, auf welche Knoten verwiesen wird. Hier kommen Node-IDs ins Spiel. Jeder Knoten hat eine ID, die ihn eindeutig identifiziert. Die Knoten-ID für einen bestimmten Knoten wird gebildet, indem die Namen der Vorgängerknoten diesem Knotennamen vorangestellt werden, während alle Namen durch ein Punktsymbol getrennt werden. Lassen Sie uns nun IDs anstelle von Rohnamen verwenden, um Knoten den oben genannten Standorten zuzuweisen:

- Schublade erhält Bar.Weapons-Knoten
- Kühlschrank erhält Bar.Drinks-Knoten
- Tische erhalten die Knoten "Bar.Drinks" und "Bar.Food"

Eine andere Möglichkeit, Knoten-IDs zu betrachten, besteht darin, dass es sich um Pfade im Dateisystem handelt. Stammknoten und innere Knoten (Bags) können Ordner und Blattknoten (Elemente) Dateien sein. Der einzige wesentliche Unterschied besteht darin, dass das Trennzeichen für Pfadsegmente ein Punktsymbol anstelle eines Schrägstrichs ist.

JSON-Spezifikation

Knoten werden mithilfe der JSON-Objekte angegeben. Jeder Knoten ist ein JSON-Objekt mit den folgenden beiden Attributen:

Name	Art	Beschreibung
Name	Schnur	Name des Knotens. Muss unter Geschwistern einzigartig sein.
Seltenheit	Schnur	Seltenheit des Knotens. Dies kann einer der Werte sein, die

		im Abschnitt " Seltenheit " aufgeführt sind .
--	--	---

Knoten, die andere Knoten (Behälter) enthalten, haben ein zusätzliches Attribut:

Name	Art	Beschreibung
Kinder	Array von Objekten	Knoten, die diesem Knoten untergeordnet sind.

Knoten, die Elemente (Blattknoten) darstellen, können ein zusätzliches Attribut aufweisen:

Name	Art	Beschreibung
Variationen	Array von Zeichenfolgen	Einige Elemente stellen das gleiche logische Element dar, haben aber unterschiedliche "Skins". Das Beispiel wäre das gleiche Kleidungsstück mit unterschiedlichen Farben. Elemente, bei denen es sich um Variationen der Farbe handelt, können im Variationsarray des Elementknotens angegeben werden. Jedes Mal, wenn ein Elementknoten mit Variationen ausgewählt wird, wird entweder das durch den Knoten dargestellte Element oder eine seiner Variationen mit gleichen Chancen ausgewählt.

So würde JSON für die Struktur im vorherigen Abschnitt aussehen:

```

{
  "Name": "Bar",
  "Kinder": [
    {
      "Name": "Getränke",
      "Seltenheit": "Reichlich",
      "Kinder": [
        {
          "Name": "Bier",
          "Seltenheit": "Reichlich"
        },
        {
          "Name": "Absinth",
          "Seltenheit": "Selten"
        }
      ]
    }
  ]
}

```



```

    }
  ],
  {
    "Name": "Essen",
    "Seltenheit": "Ungewöhnlich",
    "Kinder": [
      {
        "Name": "Chips",
        "Seltenheit": "Reichlich"
      },
      {
        "Name": "Haselnüsse",
        "Seltenheit": "Gewöhnlich"
      }
    ]
  },
  {
    "Name": "Waffen",
    "Seltenheit": "Sehr selten",
    "Kinder": [
      {
        "Name": "1H_KitchenKnife"
      }
    ]
  }
]
}

```

Exportieren von integrierten Knoten

Bisher haben wir einen einfachen Beutebaum verwendet, um einige grundlegende Konzepte zu veranschaulichen. Jetzt ist es an der Zeit, den eigentlichen Beutebaum zu exportieren, der vom Spiel verwendet wird. Um dies zu erreichen, werden zwei Befehle verwendet: **#ExportDefaultLootTree** und **#ExportCurrentLootTree**. **#ExportCurrentLootTree** wird im Abschnitt [Anpassung](#) erläutert .








































#ExportDefaultLootTree

Exportiert den Beutebaum in seinem Standardzustand. Der Standardzustand ist der Zustand, bevor Beuteanpassungen vorgenommen wurden, und das ist der Zustand, mit dem das Spiel

ausgeliefert wird. Nachdem du diesen Befehl ausgeführt hast, bekommst du den Beutebaum im folgenden Ordner:

- **<Server>\SCUM\Saved\Config\WindowsServer\Loot\Nodes\Default** für Multiplayer-Server.
- **%LocalAppData%\SCUM\Saved\Config\WindowsNoEditor\Loot\Nodes\Default** für den Einzelspielermodus.

Der Standard-Beutebaum ist zu groß, um verwaltet zu werden, wenn er in eine einzelne Datei exportiert wird, daher wird er in mehrere JSON-Dateien exportiert, wobei jede Datei JSON für jedes direkt untergeordnete Element des ItemLootTreeNodes-Knotens enthält. ItemLootTreeNodes ist der Stammknoten für alle Knoten, die im Spiel verwendet werden. So sieht der Ausgabeordner aus:

Name	Date modified	Type	Size
 Airfield.json	5.10.2023. 21:49	JSON File	6 KB
 Bar.json	5.10.2023. 21:49	JSON File	16 KB
 Barn.json	5.10.2023. 21:49	JSON File	23 KB
 Bathroom.json	5.10.2023. 21:49	JSON File	16 KB
 Beach.json	5.10.2023. 21:49	JSON File	3 KB
 BrickFactory.json	5.10.2023. 21:49	JSON File	22 KB
 CarRepairShop.json	5.10.2023. 21:49	JSON File	12 KB
 CarWreck.json	5.10.2023. 21:49	JSON File	51 KB
 Castle.json	5.10.2023. 21:49	JSON File	5 KB
 Cave.json	5.10.2023. 21:49	JSON File	21 KB
 ChineseStore.json	5.10.2023. 21:49	JSON File	2 KB
 CoalMine.json	5.10.2023. 21:49	JSON File	21 KB
 DeadPuppets.json	5.10.2023. 21:49	JSON File	23 KB
 Depository.json	5.10.2023. 21:49	JSON File	4 KB
 DrinkStore.json	5.10.2023. 21:49	JSON File	8 KB
 Farm.json	5.10.2023. 21:49	JSON File	2 KB
 FireStation.json	5.10.2023. 21:49	JSON File	4 KB
 FoodStore.json	5.10.2023. 21:49	JSON File	16 KB
 Garage.json	5.10.2023. 21:49	JSON File	38 KB
 GasShop.json	5.10.2023. 21:49	JSON File	23 KB
 Hospital.json	5.10.2023. 21:49	JSON File	21 KB
 HuntingStore.json	5.10.2023. 21:49	JSON File	80 KB
 HuntingTower.json	5.10.2023. 21:49	JSON File	13 KB
 Insects.json	5.10.2023. 21:49	JSON File	3 KB
 Kitchen.json	5.10.2023. 21:49	JSON File	16 KB
 LivingRoom.json	5.10.2023. 21:49	JSON File	32 KB
 Masonry.json	5.10.2023. 21:49	JSON File	22 KB
 Military.json	5.10.2023. 21:49	JSON File	64 KB
 Observatory.json	5.10.2023. 21:49	JSON File	49 KB
 Office.json	5.10.2023. 21:49	JSON File	18 KB
 Pharmacy.json	5.10.2023. 21:49	JSON File	21 KB
 Police.json	5.10.2023. 21:49	JSON File	36 KB
 Quarry.json	5.10.2023. 21:49	JSON File	22 KB
 SaltMine.json	5.10.2023. 21:49	JSON File	21 KB
 School.json	5.10.2023. 21:49	JSON File	22 KB
 Sports.json	5.10.2023. 21:49	JSON File	4 KB
 Trash.json	5.10.2023. 21:49	JSON File	17 KB
 Workshop.json	5.10.2023. 21:49	JSON File	53 KB
 WW2.json	5.10.2023. 21:49	JSON File	20 KB

Den generierten Standard-Beutebaum könnt ihr hier einsehen.

Nachdem Sie den Standard-Beutebaum exportiert haben, haben Sie noch keine Anpassungen vorgenommen. Sie haben nur einen Ausgangspunkt, der für die Analyse und Änderung nützlich ist.

Anpassung

Dieser Abschnitt führt Sie durch die Anpassung des Beutebaums, der in einem Spiel verwendet wird. Wir zeigen Ihnen Schritt für Schritt, wie Sie ändern und testen können, welches bestimmte Objekt in einem Spiel nach der Untersuchung fallen gelassen wird. Bitte beachten Sie, dass Spielobjekte nicht unbedingt Knoten verwenden müssen, um anzugeben, welche Gegenstände sie fallen lassen. Sie könnten direkt auf Elemente verweisen und überhaupt keine Knoten verwenden. Aber da wir Knoten erklären, werden wir diesen Fall behandeln. Verschiedene Möglichkeiten, um festzulegen, welche Gegenstände von den Spielobjekten fallen gelassen werden, werden im [Abschnitt Spawner-Voreinstellungen](#) erklärt .

Wir werden nun vollständig anpassen, was der rote Papierkorb auf dem Bild unten fallen lässt.



Zu Demonstrationszwecken ist die Idee, dass der rote Müllbehälter Getränke und Lebensmittel abwirft. Getränke sollten häufiger fallen gelassen werden als Speisen. Bei der Auswahl der Getränke möchten wir entweder Wasser oder Bier fallen lassen. Bier sollte selten getrunken werden. Bei der Auswahl von Lebensmitteln möchten wir Äpfel oder Bananen mit gleichen Chancen fallen lassen.

Um das oben Gesagte zu erreichen, müssen wir aus der obigen Beschreibung einen Beutebaum-JSON erstellen. Hier ist der vollständige JSON-Beutebaum für dieses Beispiel:

```
{
  "Name": "ItemLootTreeNodes",
  "Kinder": [
    {
      "Name": "Papierkorb",
      "ChildrenMergeMode": "Ersetzen",
      "Kinder": [
        {
          "Name": "Getränke",
          "Seltenheit": "Reichlich",
          "Kinder": [
            {
              "Name": "Water_051",
              "Seltenheit": "Reichlich"
            },
            {
              "Name": "Bier",
              "Seltenheit": "Selten"
            }
          ]
        },
        {
          "Name": "Essen",
          "Seltenheit": "Gewöhnlich",
          "Kinder": [
            {
              "Name": "Apfel"
            },
            {
              "Name": "Banane"
            }
          ]
        }
      ]
    }
  ]
}
```

[Hier](#)

ist

auch

ein

[GitHub-Link.](#)

Es gibt ein JSON-Attribut oben, das zuvor nicht im Abschnitt [JSON-Spezifikation](#) erwähnt wurde, und das ist das **ChildrenMergeMode-Attribut** für einen Bag. Es weist das Beuteanpassungssystem an, wie Kinder von zwei Taschen mit derselben ID zusammengeführt werden sollen. Im Beutebaum oben ist der benutzerdefinierte Knoten ein Beutel mit der ID **ItemLootTreeNodes.Trash**. Eine Tasche mit dieser ID existiert bereits im Standard-Beutebaum und unterscheidet sich grundlegend von dem, den wir erstellen. Das Beutesystem hat zwei Optionen, wenn Taschen die gleiche ID haben:

- Ersetzen Sie die untergeordneten Elemente des Standardbereichs vollständig durch untergeordnete Elemente aus dem benutzerdefinierten Bereich.
- Führen Sie untergeordnete Elemente zusammen, sodass im resultierenden Bereich die untergeordneten Elemente mit demselben Namen aktualisiert und neue untergeordnete Elemente hinzugefügt werden.

Das erste Verhalten wird erreicht, indem das **ChildrenMergeMode-Attribut** zu in Konflikt stehenden Behältern hinzugefügt und dessen Wert auf **Replace** festgelegt wird. Das zweite Verhalten wird erreicht, indem das **ChildrenMergeMode-Attribut** entweder vollständig weggelassen oder dessen Wert auf **UpdateOrAdd** festgelegt wird. Wir haben "Ersetzen" als Zusammenführungsmodus für unseren benutzerdefinierten Behälter ausgewählt, da es uns egal ist, was sich im standardmäßigen **ItemLootTreeNodes.Trash-Behälter** befindet.

Um außerdem zu verstehen, wie das Zusammenführen funktioniert und was das Endergebnis des Zusammenführens ist, können Sie **#ExportCurrentLootTree** Befehl verwenden. Es funktioniert auf die gleiche Weise wie [#ExportDefaultLootTree](#) mit zwei wichtigen Unterscheidungen:

- Der Beutebaum wird in einen Ordner mit dem Namen "**Aktuell**" **exportiert**, im Gegensatz zu "**Standard**".
- Das Ergebnis ist ein tatsächlicher Beutebaum, der verwendet wird, um zu bewerten, welche Gegenstände gespawnt werden sollen, und es ist das Ergebnis der Zusammenführung zwischen dem Standard-Beutebaum und allen benutzerdefinierten Beutebäumen, die du erstellt hast. Wenn du keine Anpassungen des Beutebaums hast, ist der aktuelle Beutebaum identisch mit dem Standard-Beutebaum.

Jetzt haben wir also einen benutzerdefinierten Beutebaum und müssen den roten Müllbehälter dazu bringen, Wasserflaschen, Bier, Äpfel und Bananen fallen zu lassen. Gehen Sie dazu folgendermaßen vor:

1. Starten Sie das Spiel. Wir empfehlen den Einzelspielermodus zum Experimentieren mit Nodes.
2. Teleportiere deinen Charakter mit dem folgenden Befehl: **#Teleport -493223 -423063 611**. Roter Mülleimer sollte in Ihrer Nähe sein.
3. Exportieren Sie den Standard-Beutebaum mit dem [Befehl #ExportDefaultLootTree](#). Das Spiel sagt dir, wohin der Baum exportiert wurde. Im Einzelspielermodus sollte dies sein: **%LocalAppData%\SCUM\Saved\Config\WindowsNoEditor\Loot\Nodes\Default**
4. Der rote Papierkorbcontainer löscht standardmäßig Elemente aus dem **ItemLootTreeNodes.Trash-Behälter**. Später erfährst du, [wie](#) du bestimmst, welches

Spielobjekt was fallen lässt, ob es Knoten, feste Gegenstände oder eine andere Möglichkeit verwendet, um das Beuteergebnis zu bestimmen. Denken Sie vorerst daran, dass der rote Papierkorb jedes Element aus dem **ItemLootTreeNodes.Trash-Beutel** fallen lassen kann. Öffnen Sie die Datei "**Trash.json**", und überprüfen Sie sie. Wir werden diese Datei nicht verwenden, sondern sie nur durchgehen, um ein Gefühl dafür zu bekommen, wie die Knoten strukturiert sind und welche Elemente aus dem roten Papierkorb entfernt werden können.

5. Da wir die Knoten modifizieren und testen werden, was wir erhalten haben, wäre es nützlich, dass der rote Papierkorb jedes Mal Gegenstände fallen lässt, wenn er durchsucht wird. Wir wollen keine Zeit damit verschwenden, auf die Abklingzeit des Spawners zu warten, vor allem, wenn das Spielobjekt nichts fallen lässt. Als Erstes erhöhen wir die Wahrscheinlichkeit, dass Spielobjekte Gegenstände fallen lassen. Laden Sie [die Datei GeneralZonesModifiers.json](#) herunter, und legen Sie sie im **Ordner %LocalAppData%\SCUM\Saved\Config\WindowsNoEditor\Loot** ab. Diese Datei führt dazu, dass alle Spawnpunkte auf der Karte eine erhöhte Wahrscheinlichkeit haben, Gegenstände fallen zu lassen. Die Wahrscheinlichkeit wird um das 100-fache erhöht, so dass im Grunde genommen alle untersuchenden Spawnpunkte eine 100%ige Chance haben, etwas fallen zu lassen. [Im Abschnitt GeneralZonesModifiers.json](#) wird dies ausführlich erläutert. Nehmen Sie einfach die allgemeinen Zonenmodifikatoren so, wie sie sind.
6. Da du nun eine neue Anpassung (allgemeine Zonenmodifikatoren) im **Beuteverzeichnis** hinzugefügt hast, musst du dem Spiel sagen, dass es sie neu laden soll. Führe [#ReloadLootCustomizationsAndResetSpawners](#) Befehl aus, um alle Beuteanpassungen neu zu laden. Nach dem Ausführen des Befehls erhöht das Spiel die Wahrscheinlichkeit, dass Untersuchungs-Spawnpunkte Gegenstände fallen lassen, um das 100-fache und setzt auch die Abklingzeiten aller Untersuchungs-Spawnpunkte zurück.
7. Versuchen Sie, den roten Papierkorb zu durchsuchen und dann den Befehl [#ReloadLootCustomizationsAndResetSpawners](#) wiederholt nacheinander auszuführen. Wenn du das tust, siehst du, dass du immer wieder etwas aus dem roten Mülleimer suchen und herausholen kannst. Auf diese Weise können Sie schnell testen, was abgelegt wird.
8. Erstellen Sie den **Ordner Override** im Ordner **%LocalAppData%\SCUM\Saved\Config\WindowsNoEditor\Loot**. Das Spiel liest alle Anpassungen des Beutebaums aus dem **Override-Ordner**.
9. Erstellen Sie die Datei **MyTrash.json** im Ordner **Override**, und kopieren Sie den benutzerdefinierten JSON-Code **ItemLootTreeNodes.Trash**, der zuvor erstellt wurde, in die Datei. Alternativ können [Sie diese](#) Datei auch in Ihren **Override-Ordner** herunterladen.
10. Verwende [#ReloadLootCustomizationsAndResetSpawners](#), um Anpassungen des Beutebaums anzuwenden und zu testen.
11. Experimentiere mit verschiedenen Knotenstrukturen, Seltenheiten und Gegenständen.
12. Vergiss nicht, die Datei **GeneralZonesModifiers.json** zu löschen, wenn du fertig bist, es sei denn, du möchtest, dass alles mit 100%iger Wahrscheinlichkeit in einer regulären

Einzelpielersitzung

gespawnt

wird.

Dies sollte vorerst alles sein, was Sie über Knoten wissen müssen. Später erfährst du, wie du Knoten aus benutzerdefinierten [Spawner-Voreinstellungen](#) referenzierst.

Häufige Fallstricke

Es kommt häufig vor, dass ein kleiner Syntaxfehler in der JSON-Datei auftritt und man sich fragt, warum die Beuteanpassung danach nicht funktioniert. Wenn die Beuteanpassungen nicht wie erwartet funktionieren, empfehlen wir, das Spielprotokoll auf Fehler zu überprüfen. Wenn wir z. B. einen Fehler in der Datei MyTrash.json gemacht und das Komma an der falschen Stelle eingefügt haben, enthält das Ausgabeprotokoll etwa Folgendes:

```
LogItemSpawningDataRegistry: Fehler: Fehler beim Parsen  
von
```

```
"E:/Projects/SCUM/Main/SCUM/Saved/Config/Windows/Loot/Nodes/Override/MyTrash.json".
```

Wir sind uns einig, dass es nicht sehr aussagekräftig ist, aber zumindest wissen Sie, in welcher Datei Sie nach einem Fehler suchen müssen.

Neben Syntaxfehlern enthalten Beute-bezogene Spielprotokolle weitere nützliche Informationen zu den Beuteanpassungen. Beutebezogene Kategorien sind LogItemSpawning, LogItemLootTree und LogItemSpawningDataRegistry, so dass Sie in der Lage sein sollten, Spielprotokolle mit diesen Ausdrücken zu filtern.

Exportierend

Wir haben bereits alles unten Geschriebene exportiert. Du findest auch alle Sektoren und ein paar Städte, so dass du das Modifizieren im Einzelspielermodus üben kannst, **ohne im Mehrspielermodus** nach Orten exportieren zu müssen. Sie können es hier herunterladen: <https://github.com/CheeseKingu/Loot-Modifiers/blob/main/Loot.7z>

#ExportDefaultItemSpawningParameters

Dieser Befehl erstellt eine .json-Datei mit Parametern für alle Gegenstände, die vom Beutesystem gespawnt werden können. Das Beispiel für Gegenstände, die nicht enthalten sind, sind hergestellte Gegenstände.

- Nachdem Sie #ExportDefaultItemSpawningParameters **in** Multiplayer **einggegeben** haben, exportieren Sie "**Parameters.json**", die sich hier befindet:
Server\SCUM\Saved\Config\WindowsServer\Loot\Items\Default
- Nachdem Sie #ExportDefaultItemSpawningParameters **Singleplayer** **eingegeben** haben, exportieren Sie "**Parameters.json**", die sich hier befindet:
%LocalAppData%\SCUM\Saved\Config\WindowsNoEditor\Loot\Items\Default

- Wir nehmen **M82A1** als Beispiel, um zu erklären, was die Linien bedeuten:

```
{
  "Id": "Weapon_M82A1",
  "IsDisabledForSpawning": false,
  "AllowedLocations": [
    "Coastal",
    "Continental",
    "Mountain"
  ],
  "CooldownPerSquadMemberMin": 0,
  "CooldownPerSquadMemberMax": 0,
  "Variations": [
    "Weapon_M82A1_Black",
    "Weapon_M82A1_Desert",
    "Weapon_M82A1_Snow"
  ],
  "ShouldOverrideInitialAndRandomUsage": false,
  "InitialUsageOverride": 0,
  "RandomUsageOverrideUsage": 0
},
```

- **"IsDisabledForSpawning": false** - das Element kann gespawnt werden, wenn Sie **"true"** setzen, wird das Element nicht spawnen.
- **"AllowedLocations":**
Küste - Gegenstände, die nur an der Meeresküste der Karte spawnen.
Kontinental - Gegenstände, die nur auf dem kontinentalen Gebiet der Karte spawnen.
Berg - Gegenstände, die nur im Bergbereich der Karte spawnen.
- **"CooldownPerSquadMemberMin": 0** - **wenn der Gegenstand** von jemandem aus deinem Trupp geplündert wurde, kann der gesamte Trupp den Gegenstand nicht für die hier festgelegte Mindestmenge (in Stunden) finden.
- **"CooldownPerSquadMemberMax": 0** - **wenn der Gegenstand** von jemandem aus deinem Trupp geplündert wurde, kann der gesamte Trupp den Gegenstand für die hier festgelegte maximale Menge (in Stunden) nicht finden.
- Wenn es auf diese Weise eingestellt ist, **"CooldownPerSquadMemberMin": 1** und **"CooldownPerSquadMemberMax": 4**, **beträgt die Abklingzeit für einen bestimmten Gegenstand 1 bis 4 Stunden.**
- **"Variationen": ["Weapon_M82A1_Black", "Weapon_M82A1_Desert", "Weapon_M82A1_Snow"]** - Wie Sie sehen können, sind dies alle Skin-Variationen für M82A1, die hier platziert sind. Wenn du **Weapon_M82A1** in den **Einstellungen** für **Knoten** oder **Gegenstände** in einem beliebigen Spawner-Preset einfügst, kann es auch seine Variationen spawnen.
- **"ShouldOverrideInitialAndRandomUsage": false** - es werden nicht die Verwendungseinstellungen aus der Spawner-Voreinstellung verwendet, sondern **"InitialUsageOverride"** und **"RandomUsageOverrideUsage"** in dieser Datei.
- **"InitialUsageOverride": 0** - der Wert ist in Prozent, wenn Sie ihn auf 80 setzen, werden 80% der maximalen Anzahl von Verwendungen entfernt, die ein Element hat.
- **"RandomUsageOverrideUsage": 0** - der Wert ist in Prozent, wenn Sie ihn auf 20 setzen, werden 0-20% vom Maximalbetrag entfernt.

- Zum Beispiel: Unsere **Gasflasche** hat 40 Verwendungszwecke. Wir haben **"InitialUsageOverride": 80** und **"RandomUsageOverrideUsage": 20** festgelegt. Wenn **"RandomUsageOverrideUsage": 20** ausgewählt wurde, sagen wir 10, dann werden insgesamt 90 % von 40 Verwendungen entfernt. In diesem Fall **spawnt unser Gaskanister mit 4/40 Einsätzen.**

Überschreiben von Parameters.json

- Um Ihre **"Parameters.json"** zu ändern, müssen Sie den **Ordner "Override"** erstellen
 1. Für den **Mehrspielermodus** gehen Sie zu **Server\SCUM\Saved\Config\WindowsServer\Loot\Items** und erstellen Sie den **Ordner "Override"**.
 2. Für den **Einzelspielermodus** erstellen Sie es hier:
%LocalAppData%\SCUM\Saved\Config\WindowsNoEditor\Loot\Items
 3. Sie können so viele JSON-Dateien erstellen, wie Sie möchten, und sie im Ordner **"Überschreiben"** nach Belieben benennen, dies dient ausschließlich organisatorischen Zwecken.
 4. Sie können eine beliebige Anzahl von Elementen in den von Ihnen erstellten JSON-Dateien platzieren, es liegt an Ihnen, wie Sie sie organisieren möchten. **Beispiel** für einen solchen Ordner: <https://github.com/CheeseKingu/Loot-Modifiers/blob/main/Items.7z>
 5. In diesem Ordner können Sie sehen, dass wir die Standardeinstellung und die Überschreibung haben, die wir erstellt haben.
 6. Im Override-Ordner haben wir 2 JSON-Dateien erstellt, die wir nach unseren Wünschen benannt haben. In den JSON-Dateien haben wir ein paar Elemente

überschrieben.

```
Xjson x
1 {
2   "Parameters": [
3     {
4       "Id": "Banana",
5       "IsDisabledForSpawning": false,
6       "AllowedLocations": [
7         "Coastal",
8         "Continental",
9         "Mountain"
10      ],
11      "CooldownPerSquadMemberMin": 0,
12      "CooldownPerSquadMemberMax": 0,
13      "Variations": [],
14      "ShouldOverrideInitialAndRandomUsage": false,
15      "InitialUsageOverride": 0,
16      "RandomUsageOverrideUsage": 0
17    },
18    {
19      "Id": "Apple",
20      "IsDisabledForSpawning": false,
21      "AllowedLocations": [
22        "Coastal",
23        "Continental",
24        "Mountain"
25      ],
26      "CooldownPerSquadMemberMin": 0,
27      "CooldownPerSquadMemberMax": 0,
28      "Variations": [],
29      "ShouldOverrideInitialAndRandomUsage": false,
30      "InitialUsageOverride": 0,
31      "RandomUsageOverrideUsage": 0
32    }
33  ]
34 }
```

#ExportDefaultItemSpawnerPresets:

Dieser Befehl exportiert alle Standard-Spawner-Voreinstellungen, die wir im Spiel verwenden.

- Um loszulegen, geben Sie **#ExportDefaultItemSpawnerPresets** in Multiplayer ein, **um den** Ordner "Loot" in Ihrer Konfiguration zu erstellen, der sich hier befindet:
Server\SCUM\Saved\Config\WindowsServer\Loot\Spawners\Presets\Default
- Wenn du im Singleplayer **#ExportDefaultItemSpawnerPresets**, wird es hier gespeichert:
%LocalAppData%\SCUM\Saved\Config\WindowsNoEditor\Loot\Spawners\Presets\Default
- Im Standardordner findest du alle exportierten Spawner-Presets. Sie können keine neuen erstellen, Sie können nur die vorhandenen ändern.
- Voreinstellungen, die "**Untersuchen**" im Namen haben, werden Objekten zugewiesen, die du "**suchen**" musst, damit sie Beute fallen lassen, z. B. **wird Gebäude-Küche-Examine_Cabinet** Küchenschränken zugewiesen, die du in Häusern suchst.
- Presets, die "**World**" im Namen haben, werden Spawnpunkten zugewiesen, die Gegenstände in der Nähe spawnen. Zum Beispiel **wird Buildings-Garage-Residential-World_Shelf** Regalen in Garagen in Wohngebieten zugeordnet und wirft Gegenstände wie **Gasflaschen** oder **Autobatterien** ab.

#ExportItemSpawnerPresetsInZone

Dieser Befehl exportiert alle Spawner-Voreinstellungen, die in einer bestimmten Zone verwendet werden.

- Positionen können nur mit einer Rechteckform auf der Karte oder mithilfe des Sektornamens angegeben werden.
- Dieser Befehl generiert keine Landschafts-, Farming-, Charakter- und Fracht-Drop-Spawner-Voreinstellungen, diese müssen manuell hinzugefügt werden, wie in den Schritten weiter unten erklärt.
- Alle Zonen sollten im **Mehrspielermodus** exportiert werden, im **Einzelspielermodus** gibt es eine bestimmte Distanz um deinen Gefangenen herum, so dass du nicht effektiv exportieren kannst.

Exportieren nach Sektornamen:

- Wenn Sie **#ExportItemSpawnerPresetsInZone A2** im **Mehrspielermodus** eingeben. Dadurch wird der folgende Ordner erstellt:
Server\SCUM\Saved\Config\WindowsServer\Loot\Spawners\Presets\Override\A2

- Wenn du **#ExportItemSpawnerPresetsInZone A2** im **Einzelspielermodus** eingibst. Dadurch wird der folgende Ordner erstellt:
%LocalAppData%\SCUM\Saved\Config\WindowsNoEditor\Loot\Spawners\Presets\Override\A2
- Im Ordner "A2" findest du alle Spawner-Presets, die im A2-Sektor verwendet werden.
- Am Ende des Ordners finden Sie die "Zones.json":

Street-Residential-World_Rabbitary.json	10/2/2023 11:59 AM	JSON File
Water-Shipwrecks-Examine_Crate.json	10/2/2023 11:59 AM	JSON File
Water-Shipwrecks-Examine_Crate_Green...	10/2/2023 11:59 AM	JSON File
Water-Shipwrecks-Examine_Crate_Green...	10/2/2023 11:59 AM	JSON File
Zones.json	10/2/2023 11:59 AM	JSON File

- Diese Datei enthält die Positionen "TopLeft" und "BottomRight" des Rechtecks, das in diesem Fall der A2-Sektor ist:

```

1  {
2    "Zones": [
3      {
4        "TopLeft": "X=9600.000 Y=-295198.375",
5        "BottomRight": "X=-295198.375 Y=-599996.750"
6      }
7    ]
8  }

```

- Die "Zones"-Datei bestimmt die Ortsgrenze für alle Spawner-Presets im angegebenen Ordner; das wird im nächsten Absatz mehr Sinn machen.

Exportieren nach Rechteckpositionen:

- Öffnen Sie ein beliebiges Admin-Hilfsprogramm oder öffnen Sie https://scum-map.com/en/interactive_map

- Suchen Sie einen Ort, den Sie ändern möchten, z. B. werden wir Tisno ändern:



- Wenn wir `#ExportItemSpawnerPresetsInZone A2` eingeben würden, **würden wir die Spawner-Presets für den gesamten A2 erhalten und wir wollen hier nur Tisno angeben.**
- Das für diesen Befehl verwendete Format ist `#ExportItemSpawnerPresetsInZone X Y X Y Name`.
- Wir würden also das X Y der oberen linken Position eingeben, dann das X Y der unteren rechten Position und der Name ist vollständig für organisatorische Zwecke. Der angegebene Name ist der Name des Ordners, der erstellt wird.
- Klicken Sie mit der rechten Maustaste auf die obere linke Position des Rechtecks und drücken Sie TP-Position kopieren. Sie erhalten diesen Wert: `#Teleport -15613.1299 -472342.2348 0 (X= -15613.1299, Y= -472342.2348)`.
- Speichern Sie die Position irgendwo und kopieren Sie dann die untere rechte Position des Rechtecks, Sie erhalten diesen Wert: `#Teleport -83853.2402 -516124.1469 0 (X= -83853.2402, Y= -516124.1469)`.
- Um den Befehl jetzt zu verwenden, müssen wir das X-Y von der ersten und der zweiten Position nehmen und in den Befehl einfügen. Wir werden auch Tisno für organisatorische Zwecke eingeben.
- Geben Sie also `#ExportItemSpawnerPresetsInZone -15613.1299 -472342.2348 -83853.2402 -516124.1469 Tisno` ein. Es hat 130 Voreinstellungen für den angegebenen Ort exportiert

Exported 130 spawner preset(s).

- Wir haben nun den Ordner "Tisno" erstellt, **der sich hier befindet:**
Server\SCUM\Saved\Config\WindowsServer\Loot\Spawners\Presets\Override\Tisno.
- In diesem Ordner finden wir alle Spawner-Voreinstellungen, die an diesem Ort verwendet werden, und wir können sehen, dass der Befehl die entsprechende Datei `"Zones.json"` erstellt hat .

Zones.json

- Die "**Zones.json**" legt die Grenzen für die Spawner-Presets in einem bestimmten Ordner fest.
- Wenn in einem bestimmten Ordner keine Zones-Datei vorhanden ist, werden die Zones aus dem übergeordneten Ordner verwendet (die untergeordnete Zones-Datei überschreibt also die Zones-Datei der übergeordneten Ordner).
- Wenn im gesamten Ordner "Override" **keine Zonendatei enthalten ist**, wird der globale Speicherort verwendet (die Voreinstellungen werden auf der gesamten Karte verwendet).

1. Wir werden das Beispiel dessen verwenden, was wir bereits exportiert haben, und wir werden ein weiteres kleines Rechteck im Hafen von Tisno exportieren (wir haben den A2-Sektor und Tisno selbst exportiert).
2. Wir werden also die gleichen Schritte zum Exportieren des Standorts verwenden, indem wir die Position oben links und unten rechts des Ports selbst verwenden.
3. Position oben links: **#Teleport -52954.4182 -493783.9695 0**
4. Position unten rechts: **#Teleport -77080.9405 -510602.7629 0**
5. Jetzt setzen wir das X Y X Y und den TisnoPort in den Befehl selbst:
#ExportItemSpawnerPresetsInZone -52954.4182 -493783.9695 -77080.9405 -510602.7629 TisnoPort

6. So sieht unsere Situation auf der Karte jetzt aus:



7. Jetzt haben wir also die **"Zones"-Datei in unserem A2-Ordner, wir haben sie auch in unserem "Tisno"-Ordner** und wir haben sie in unserem **"TisnoPort"-Ordner**.
8. Die Standorte sind von den kleinsten bis zu den größten priorisiert.
9. In diesem Fall wird also unsere Konfiguration für A2 für alles in A2 gesetzt, mit Ausnahme von Tisno und ohne TisnoPort.
10. Die Konfiguration, die im Tisno-Ordner durchgeführt wird, priorisiert die Spawner-Voreinstellungen gegenüber den Spawner-Voreinstellungen im **Ordner "A2"**.
11. Das Gleiche gilt für TisnoPort, es wird die Spawner-Presets im Inneren gegenüber den "Tisno"-Spawner-Presets priorisieren .

Kleinere Standorte modifizieren

- Sie können Haus für Haus ändern, wenn Sie möchten, aber **bitte beachten Sie, dass** die Speicherorte in Admin-Hilfsprogrammen möglicherweise nicht korrekt sind.
- Wenn du Haus für Haus modifizieren möchtest, solltest du Orte im Spiel nehmen, die du am besten mit zwei Spielern machst (einer oben links, einer unten rechts).
- Administratoren sollten **#Location eingeben** , wenn sie sich an den gewünschten Orten befinden.
- Nachdem du die Orte im Spiel übernommen hast, solltest du sie mit den generierten X-Y-X-Y-Positionen an den Admin-Befehl übertragen.

Überlappende Zonen

- Wenn du eine Situation wie unten gezeigt hast, hat das rote Rechteck eine eigene Beutevoreinstellung.
- Der obere rechte Teil des roten Rechtecks wird jedoch durch das grüne Rechteck überschrieben, da die festgelegte Position kleiner ist.
- Daher werden alle Spawner-Voreinstellungen für das überlappende Rechteck aus dem grünen Rechteckordner priorisiert.
- Beispiel: Wenn Sie festlegen, dass Steine ein Katana in das rote Rechteck fallen lassen und wenn Sie festlegen, dass Steine im grünen Rechteck ein MK18 fallen lassen, wird jeder Stein im überlappenden Rechteck ein

MK18.



GeneralZonesModifiers.json

- Dies ist etwas, das wir verwenden können, wenn wir einen bestimmten rechteckigen Bereich auf der Karte oder den gesamten Sektor auswählen möchten, um die Beute global auf dem ausgewählten Gebiet zu ändern.
- Sie müssen die **Datei "GeneralZonesModifiers.json"** manuell im Stammordner **"Loot"** erstellen, der sich hier befindet:
Server\SCUM\Saved\Config\WindowsServer\Loot
- Beispiel für die Verwendung, Kopieren und Einfügen Link:
<https://github.com/CheeseKingu/Loot-Modifiers/blob/main/GeneralZoneModifiers>

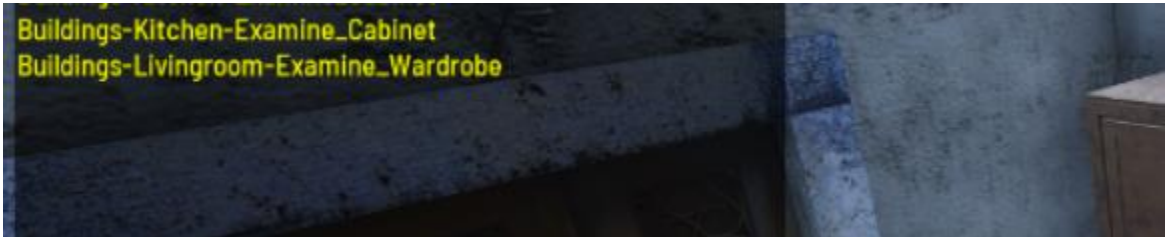
```
GeneralZoneModifiers.json x
1 {
2   "Modifiers": [
3     {
4       "Zones": [
5         {
6           "Name": "FishFactory",
7           "TopLeft": "X=-327211.8142 Y=-346843.5702",
8           "BottomRight": "X=-363868.8911 Y=-383500.647"
9         }
10      ],
11      "SpawnerProbabilityMultiplier": 100,
12      "ExamineSpawnerProbabilityMultiplier": 100,
13      "ExamineSpawnerQuantityMultiplier": 5
14    },
15    {
16      "Zones": [
17        {
18          "Sector": "A1"
19        }
20      ],
21      "ExamineSpawnerQuantityMultiplier": 5
22    }
23  ]
24 }
```

1. Wir haben die Datei "**GeneralZonesModifiers.json**" erstellt und die **Position "TopLeft"** und "**Bottomright**" der Fischfabrik hinzugefügt.
2. Die **Zeile "Name"** dient ausschließlich organisatorischen Zwecken.
3. Dadurch wird die Beute, die wir unten in diesem bestimmten Gebiet angegeben haben, global erhöht.
4. Du kannst auch die Beute des gesamten Sektors global erhöhen, wie im Bild oben gezeigt.
5. Diese multiplizieren sich im Grunde mit Ihren "**ServerSettings.ini**" - Beuteeinstellungen für das von Ihnen ausgewählte Gebiet und auch mit der Menge, die Sie eingegeben haben.
6. Daher haben wir die **ExamineSpawnerQuantity** für den gesamten A1-Sektor auf 5 gesetzt, wenn sie in Ihrer "**ServerSettings.ini**" auf **0,5** gesetzt ist. Er wird auf 2,5 ($5 * 0,5 = 2,5$) festgelegt.
7. Wir haben auch die gesamte **FishFactory** auf die Werte eingestellt, die auf dem Bild zu sehen sind.

#SetShouldPrintExamineSpawnerPresets

Dieser Befehl gibt aus, welche Spawner-Voreinstellung du von welchem Objekt plünderst.

- Um herauszufinden, welche Untersuchen-Voreinstellung welchem Objekt im Spiel zugewiesen ist, müssen Sie **#SetShouldPrintExamineSpawnerPresets true eingeben**. Nachdem du ein Objekt gesucht hast, wird das zugewiesene Spawner-Preset in deinem Chat und deinem Log ausgegeben.
- Sie können das Protokoll mit diesen Zeilen filtern, um alles zu finden, was Sie mit dieser Zeile gesucht haben: **Spawner-Voreinstellung untersuchen**.
- Beispiel für die Zeilen, die in **SCUM.log** und **Loot.log** gedruckt werden:
LogItemSpawning: Spawner-Voreinstellung untersuchen: Buildings-Office-Police-Examine_File_Cabinet.



Einzelspieler-Modus

Um im Singleplayer **zu testen**, werden alle Suchvorgänge, die Sie durchgeführt haben, in Ihrem **"SCUM.log" ausgegeben**, das sich im folgenden Ordner befindet:
C:\Users\%username%\AppData\Local\SCUM\Saved\Logs

Mehrspielermodus

Um im **Multiplayer** zu testen, werden alle Suchvorgänge, die Sie durchgeführt haben, in Ihrem **"Loot.log"** ausgegeben, das sich in der Datei **Server\SCUM\Saved\SaveFiles\Logs\Loot.log** befindet.

#ReloadLootCustomizationsAndResetSpawners

Lädt alle Beuteanpassungen, die du in der **Konfiguration/Beute vorgenommen hast, neu**. Außerdem werden Spawnpunkte untersucht, sodass sie würfeln, welche Gegenstände gespawnt werden sollen, ohne auf eine Abklingzeit zu warten (wenn sie kürzlich durchsucht wurden). Spawnpunkte in der Umgebung werden nicht zurückgesetzt, also müsst ihr warten, bis die Abklingzeit abgelaufen ist, um sie zu testen. Alternativ kannst du auch im

Einzelspielermodus testen und nach jeder Beutemodifikation einen neuen Charakter erstellen.

Dieser Befehl ist sehr nützlich, wenn Sie die von Ihnen vorgenommenen Anpassungen schnell testen möchten. Ein gängiger Workflow könnte sein:

1. Nehmen Sie die Beuteanpassung Ihrer Wahl vor (vergessen Sie nicht, geänderte Dateien zu speichern).
2. Führen Sie **#ReloadLootCustomizationsAndResetSpawners** Befehl aus.
3. Testen Sie die Anpassung der Beute, indem Sie die Container durchsuchen, die von den Änderungen in Schritt 1 betroffen sind.

Spawner-Voreinstellungen

Spawn-Voreinstellungen bestimmen, welche Beute du in der Welt oder bei der Suche nach Objekten findest.

Jedem gesuchten Objekt ist eine Spawnpunkt-Voreinstellung zugewiesen, ebenso jedem durchsuchbaren Objekt.

Hinweise zur Spawner-Voreinstellung:

- Voreinstellungen, die "**Untersuchen**" im Namen haben, werden Objekten zugewiesen, die du "**suchen**" musst, damit sie Beute fallen lassen, z. B. **wird Gebäude-Küche-Examine_Cabinet** Küchenschränken zugewiesen, die du in Häusern suchst.
- Presets, die "**World**" im Namen haben, werden Spawnpunkten zugewiesen, die Gegenstände in der Nähe spawnen. Zum Beispiel: Gaskanister in Regalen, Kleidung in Waschmaschinen usw.
- Nur die Änderungen, die im **Ordner "Override"** vorgenommen werden, ändern die Änderungen der Spawner-Voreinstellungen, alles andere, was Sie extrahiert haben, sind deren Standardwerte.
- Wenn Sie Spawner-Voreinstellungen nur zum **Ordner "Override"** hinzufügen, werden diese Änderungen global, da Sie mit "**Zones.json**" **keinen Speicherort angegeben haben**.
- Gegenstände, die bereits auf der Karte vorhanden sind, werden nicht ersetzt, sobald die Spawnpunktvoreinstellung geändert wurde.
- Nachdem Sie eine bestimmte Spawner-Voreinstellung geändert haben, müssen Sie warten, bis der Gegenstand abgelaufen ist und der neue spawnnt, oder Sie können **#ReloadLootCustomizationsAndResetSpawners verwenden** , um durchsuchbare Objekte neu zu laden. Dies wirkt sich nicht auf Spawns in der Umgebung oder "**Welt**" aus.

- Der beste Weg wäre, alles zu extrahieren, was du im **Multiplayer** brauchst, und dann den gesamten Loot-Ordner in den **Singleplayer** zu übertragen, indem du ihn hier einfügst:
C:\Users\%username%\AppData\Local\SCUM\Saved\Config\WindowsNoEditor
- Im **Einzelspielermodus** kannst du einfach einen neuen Charakter erstellen und deine Spawns in der Umgebung werden ebenfalls neu geladen.

Spawner-Preset-Einstellungen

- Wir werden **Buildings-Factory-Quarry-World_Production_Line** als Beispiel verwenden, um zu beschreiben, worauf sich die Spawner-Preset-Einstellungen beziehen:

```
Buildings-Factory-Quarry-World_Production_Line.json x
1  {
2    "Nodes": [
3      {
4        "Rarity": "Uncommon",
5        "Ids": [
6          "ItemLootTreeNodes.Quarry.Crafting",
7          "ItemLootTreeNodes.Quarry.Tools.Heavy",
8          "ItemLootTreeNodes.Quarry.WorkClothes.Torso",
9          "ItemLootTreeNodes.Quarry.WorkClothes.Legs",
10         "ItemLootTreeNodes.Quarry.WorkClothes.Feet",
11         "ItemLootTreeNodes.Quarry.Other.Ammo",
12         "ItemLootTreeNodes.Quarry.Other.Explosives",
13         "ItemLootTreeNodes.Quarry.Other.Flares",
14         "ItemLootTreeNodes.Quarry.Other.Money",
15         "ItemLootTreeNodes.Quarry.Other.Backpacks.Belt_Pistol_Holster",
16         "ItemLootTreeNodes.Quarry.Other.Backpacks.Hiking.Other",
17         "ItemLootTreeNodes.Quarry.Other.Backpacks.Plain",
18         "ItemLootTreeNodes.Quarry.Other.Backpacks.School",
19         "ItemLootTreeNodes.Quarry.Other.Backpacks.WaistBags"
20       ]
21     }
22   ],
23   "Probability": 25,
24   "QuantityMin": 1,
25   "QuantityMax": 1,
26   "AllowDuplicates": false,
27   "ShouldFilterItemsByZone": true,
28   "InitialDamage": 5,
29   "RandomDamage": 35,
30   "InitialUsage": 5,
31   "RandomUsage": 35
32 }
```

Wahrscheinlichkeit

- **"Wahrscheinlichkeit": 25**, - 25% für den Abfall des Elements multipliziert mit dem, was Sie in Ihren **ServerSettings.ini** und Zonenmodifikatoren haben.
- Wenn Sie eine **100%ige Drop-Chance wünschen**, können Sie einfach **die gesamte** Zeile "Wahrscheinlichkeit" löschen
- **Seien Sie vorsichtig**, wenn Sie **Wahrscheinlichkeit": 100** setzen, es wird immer noch mit den **ServerSettings.ini und** Zonenmodifikatoren multipliziert, und Ihre Drop-Chance wird nicht 100% betragen

Menge

- **"QuantityMin": 1**, - Mindestanzahl an Gegenständen, die aus der Spawner-Voreinstellung fallen gelassen werden.
- **"QuantityMax": 1**, - maximale Anzahl von Gegenständen, die aus der Spawner-Voreinstellung fallen gelassen werden.
- Wenn du "QuantityMin": 4 und "QuantityMax": **9** eingibst, **wählt das Spawner-Preset eine Zahl von 4 bis 9 und lässt so viele Gegenstände fallen.**

Duplikate

- **"AllowDuplicates": false**, - es wird nie zwei gleiche Elemente geben
- Selbst wenn die Spawner-Voreinstellung 7 Gegenstände enthält, aber 3 dieser Gegenstände doppelt vorhanden sind, erhalten Sie nur 4 Gegenstände.
- **"AllowDuplicates": true** - die Spawner-Voreinstellung kann denselben Gegenstand mehrmals spawnen.

Schaden

- **"InitialDamage": 5**, - Der Gegenstand sollte mit 100 % Haltbarkeit spawnen, aber die Spawner-Voreinstellung fügt **diesem Gegenstand 5 % Haltbarkeitsschaden** zu, so dass du den Gegenstand auf 95 % bekommst
- **"RandomDamage": 35**, - wählt eine Zahl von 0 bis 35 und verursacht so viel % Schaden der maximalen Haltbarkeit.
- In diesem Fall haben wir **"InitialDamage": 5 und "RandomDamage": 35 hat 25 ausgewählt, unser Gegenstand spawnnt mit 70% Haltbarkeit (100 - 5 - 25 = 70)**

Benutzungen

- **"InitialUsage": 5**, - entfernt 5% der maximalen Verwendungen für einen Gegenstand, wenn der Gegenstand 20 Verwendungen hat, wird 1 Verwendung entfernt.
- **"RandomUsage": 35**, - wählt eine Zahl von 0 bis 35 und teilt so viel % der maximalen Haltbarkeit aus.

- In diesem Fall haben wir **"InitialUsage": 5** und **"RandomUsage": 35** hat **15** ausgewählt, unser Gegenstand wird bei **16/20** Einsätzen spawnen ($20 - 20\% = 20 - 4 = 16$).

ShouldFilterItemsByZone

- Dies bezieht sich auf die **Zonenpositionen "Coastal", "Continental"** und **"Mountain"**, die in der **Datei "Parameters.json"** festgelegt sind.
- **"ShouldFilterItemsByZone": true**, - bedeutet, dass die in der Parameters.json festgelegten Zonenpositionen verwendet werden, wenn Sie es auf **false** setzen, kann es überall spawnen. (Ein Beispiel: Du hast 10 Gegenstände in deinem **Node** und die Spawnpunkt-Voreinstellung ist auf ein Objekt gesetzt, das sich im **kontinentalen Bereich** befindet. Jeder Gegenstand, der **nicht** Continental in seinen **AllowedLocations** hat, wird dort nicht spawnen).
- Sie können die **Datei "Parameters.json"** mit dem folgenden Befehl **exportieren**: **#ExportDefaultItemSpawningParameters** sie befindet sich hier: **Server\SCUM\Saved\Config\WindowsServer\Loot\Items\Default**
- **Achtung**, wenn Sie die Datei "Parameters.json" bearbeiten möchten, **müssen Sie Elemente zu Dateien hinzufügen, die** im Ordner "Override" erstellt wurden, **der im Abschnitt "Override Parameters.json" erläutert wird.** Der Speicherort lautet: **Server\SCUM\Saved\Config\WindowsServer\Loot\Items\Override**
- **Küste** - Gegenstände, die nur an der Meeresküste der Karte spawnen.
- **Kontinental** - Gegenstände, die nur auf dem kontinentalen Gebiet der Karte spawnen.
- **Berg** - Gegenstände, die nur im Bergbereich der Karte spawnen.
- Zum Beispiel wirst du nie eine Weihnachtsmannjacke in der Nähe des Meeres finden, da sie keine Küstenjacke enthält:

```
"Id": "Santa_Jacket",
"IsDisabledForSpawning": false,
"AllowedLocations": [
    "Continental",
    "Mountain"
],
"CooldownPerSquadMemberMin": 0,
"CooldownPerSquadMemberMax": 0,
"Variations": [],
"ShouldOverrideInitialAndRandomUsage": false,
"InitialUsageOverride": 0,
"RandomUsageOverrideUsage": 0
```

- Du wirst auch nie eine taktische Jacke auf dem **Kontinent** oder **an der Küste** finden, da sie nur **Berg** enthält:

```

{
  "Id": "Tactical_Jacket_01_04",
  "IsDisabledForSpawning": false,
  "AllowedLocations": [
    "Mountain"
  ],
  "CooldownPerSquadMemberMin": 0,
  "CooldownPerSquadMemberMax": 0,
  "Variations": [

```

Organisieren von Spawner-Voreinstellungen:

- Wir werden die Spawner-Voreinstellungen ändern, die wir bereits in den obigen Schritten extrahiert haben.
1. Jetzt haben wir alle Spawner-Presets extrahiert, die in A2, Tisno und TisnoPort verwendet werden.
 2. Da in unserem **Ordner "Überschreiben"** diese drei extrahiert sind, werden wir sie zur besseren Organisation nach Priorität in Ordner einordnen, da sie sich alle im A2-Sektor befinden.

Name	Date modified	Type	Size
A2	10/2/2023 12:27 PM	File folder	
Tisno	10/2/2023 12:23 PM	File folder	
TisnoPort	10/2/2023 2:41 PM	File folder	

3. Da "TisnoPort" die "Tisno"-Spawner-Presets und "Tisno" die "A2"-Spawner-Presets überschreibt, **werden wir sie in diese Reihenfolge bringen, wie in der Zones.json erklärt.**
4. **Server\SCUM\Saved\Config\WindowsServer\Loot\Spawners\Presets\Override\A2\Tisno\TisnoPort.**

> Server > SCUM > Saved > Config > WindowsServer > Loot > Spawners > Presets > Override > A2 > Tisno > TisnoPort

Spawner-Voreinstellungen ändern:

- Es gibt 4 verschiedene Möglichkeiten, wie du die Beute innerhalb eines bestimmten Spawner-Presets ändern kannst: FixedItems, Items, Nodes und Subpresets.
- Wir erklären Ihnen Schritt für Schritt, wie Sie diese verwenden und was sie bedeuten und wie sie verwendet werden.

FixedItems (Behobene Elemente)

FixedItems wird verwendet, wenn alle Elemente in der Liste erzeugt werden sollen.

- FixedItems ignoriert alle Spawner-Voreinstellungseinstellungen mit Ausnahme von: InitialDamage, RandomDamage, InitialUsage, RandomUsage und PostSpawnActions.
- Kopieren und Einfügen eines Spawner-Presets mit "**FixedItems**":
<https://github.com/CheeseKingu/Loot-Modifiers/blob/main/FixedItems>

1. Öffnen Sie eine der Spawner-Voreinstellungen, die wir bereits im **Ordner "Override"** **extrahiert haben, wir haben die** Datei Street-Residential-Examine_Trash_Container_Red.json **im Ordner "TisnoPort"** ausgewählt, der sich hier befindet:
Server\SCUM\Saved\Config\WindowsServer\Loot\Spawners\Preset\Override\A2Tisno\TisnoPort

```
Street-Residential-Examine_Trash_Container_Red.json x
1 {
2   "Nodes": [
3     {
4       "Rarity": "Uncommon",
5       "Ids": [
6         "ItemLootTreeNodes.Trash"
7       ]
8     }
9   ],
10  "Probability": 20,
11  "QuantityMin": 1,
12  "QuantityMax": 1,
13  "AllowDuplicates": false,
14  "ShouldFilterItemsByZone": true,
15  "InitialDamage": 50,
16  "RandomDamage": 40,
17  "InitialUsage": 50,
18  "RandomUsage": 40,
19  "PostSpawnActions": [
20    "SetAmmoAmount_SmallStash",
21    "SetCashAmount_SmallStash",
22    "SetClothesDirtiness_DirtyClothes"
23  ]
24 }
```

2. Diese Spawner-Voreinstellung hat einen bestimmten Knoten im Inneren, der eine Reihe von Dingen fallen lassen kann, aber wir wollen einen bestimmten Gegenstand oder mehrere Gegenstände.
3. Wir nehmen den Code aus <https://github.com/CheeseKingu/Loot-Modifiers/blob/main/FixedItems> und ersetzen ihn durch den Code, den wir in der **Street-Residential-Examine_Trash_Container_Red.json** haben

```
Street-Residential-Examine_Trash_Container_Red.json x
1 {
2   "FixedItems": [
3     "Weapon_M82A1",
4     "Weapon_MK18",
5     "2H_Katana"
6   ],
7   "QuantityMin": 1,
8   "QuantityMax": 2,
9   "AllowDuplicates": true,
10  "ShouldFilterItemsByZone": false,
11  "InitialDamage": 0,
12  "RandomDamage": 0,
13  "InitialUsage": 0,
14  "RandomUsage": 0
15 }
```

4. Wir haben dem Spawner-Preset drei Gegenstände hinzugefügt. Um dies sofort zu testen, können Sie diesen Befehl verwenden:

[#ReloadLootCustomizationsAndResetSpawners](#). Wenn wir nun den roten Container im TisnoPort plündern, erhaltet ihr die folgenden Gegenstände:



Artikel

- Wenn Sie eine Reihe von Gegenständen hinzufügen möchten, müssen Sie ihnen auch eine Seltenheit zuweisen
- Kopieren und Einfügen eines Presets mit **"Items"**:
<https://github.com/CheeseKing/Loot-Modifiers/blob/main/Items>
- Spawner-Voreinstellungen mit "Gegenständen" **hängen von ihrer Seltenheit, Wahrscheinlichkeit, MengeMin und QuantityMax innerhalb der Spawner-Voreinstellung selbst ab**

1. Wir nehmen ein Beispiel für die Spawner-Voreinstellung aus dem obigen Link.
2. Wir werden es zu einer der Spawner-Voreinstellungen hinzufügen, die wir bereits in den obigen Schritten extrahiert haben.
3. Gehen Sie zu **Server\SCUM\Saved\Config\WindowsServer\Loot\Spawners\Presets\Override\A2\Tisno** und wählen Sie eine beliebige Spawner-Voreinstellung aus.
4. Wir entscheiden uns für **Landscape-Foliage-Examine_Aloe_Vera**.
5. Kopiert den Code aus dem obigen Link und ersetzt den Code in der **Landschafts-Laub-Examine_Aloe_Vera**.-Jetzt wird jede Aloe Vera, die wir im Tisno-Gebiet plündern, diese Spawner-Voreinstellung haben:

```

Landscape-Foliage-Examine_Aloe_Vera.json
1  {
2      "Items": [
3          {
4              "Rarity": "Abundant",
5              "Id": "Weapon_M82A1"
6          },
7          {
8              "Rarity": "Common",
9              "Id": "1H_Hatchet"
10         },
11         {
12             "Rarity": "Uncommon",
13             "Id": "2H_Katana"
14         }
15     ],
16     "Probability": 20,
17     "QuantityMin": 1,
18     "QuantityMax": 2,
19     "AllowDuplicates": true,
20     "ShouldFilterItemsByZone": false,
21     "InitialDamage": 0,
22     "RandomDamage": 0,
23     "InitialUsage": 0,
24     "RandomUsage": 0
25 }

```

6. Nachdem du Aloe Vera im ausgewählten Gebiet durchsucht hast, hast du zunächst die Chance, Beute zu erhalten, die durch die Wahrscheinlichkeit definiert ist.
7. Wenn Sie einen Artikel erhalten, hängt die Menge davon ab, was Sie in die "QuantityMin" und "QuantityMax" eingeben, in diesem Fall erhalten Sie 1 bis 2 Artikel.

8. Gegenstände mit geringerer Seltenheit haben eine höhere Drop-Chance, wie in der Seltenheit erklärt.



Subpresets

- Wenn du mehrere Presets mit jeder Art von Spawn-Preset innerhalb eines bestimmten Spawn-Presets haben möchtest, ist es am bequemsten, Subpresets zu verwenden. In unserem Fall verwenden wir **FixedItems** in unseren Subpresets.
- Wenn Subpresets keine Seltenheit zugewiesen ist, sind sie standardmäßig ungewöhnlich. Wenn Sie ihnen Seltenheiten zuweisen möchten, um die Chancen zu erhöhen oder zu verringern, sollten Sie dies wie unten beschrieben tun.
- Link zum Kopieren und Einfügen eines Presets mit **"Subpresets"**:
<https://github.com/CheeseKingu/Loot-Modifiers/blob/main/Subpresets>

1. Wir werden **Buildings-Armory-TV_Bunker-Examine_Weapon_Locker_Lockpick_Tier_4** als Beispiel verwenden

```
Buildings-Armory-TV_Bunker-Examine_Weapon_Locker_Lockpick_Tier_4.json
1  {
2      "Subpresets": [
3          {
4              "Rarity": "Uncommon",
5              "Id": "Special_Packages-Vault-Examine_RPK_Vault_Pack"
6          },
7          {
8              "Rarity": "Uncommon",
9              "Id": "Special_Packages-Vault-Examine_RPK_Ammo_Vault_Pack"
10         },
11         {
12             "Rarity": "Uncommon",
13             "Id": "Special_Packages-Vault-Examine_GrenadeLauncher_Vault_Pack"
14         },
15         {
16             "Rarity": "Uncommon",
17             "Id": "Special_Packages-Vault-Examine_GrenadeLauncher_Ammo_Vault_Pack"
18         },
19         {
20             "Rarity": "Uncommon",
21             "Id": "Special_Packages-Vault-Examine_Explosives_1_Vault_Pack"
22         },
23         {
24             "Rarity": "Uncommon",
25             "Id": "Special_Packages-Vault-Examine_Explosives_2_Vault_Pack"
26         },
27         {
28             "Rarity": "Uncommon",
29             "Id": "Special_Packages-Vault-Examine_Explosives_3_Vault_Pack"
30         },
31         {
32             "Rarity": "Uncommon",
33             "Id": "Special_Packages-Vault-Examine_Explosives_4_Vault_Pack"
34         },
35         {
36             "Rarity": "Uncommon",
37             "Id": "Special_Packages-Vault-Examine_Explosives_5_Vault_Pack"
38         }
39     ]
40 }
```

2. Wir verwenden hier **"Subpresets"**, weil wir mehrere Presets von **"FixedItems"** spawnen lassen wollen

3. Sie haben alle die gleiche Seltenheit, also haben sie alle die gleiche Chance zu spawnen.
4. Wenn wir die "Subpresets" **ändern oder** sehen wollen, müssen wir sie in Server\SCUM\Saved\Config\WindowsServer\Loot\Spawners\Presets\Default **finden**
5. Wir nehmen **Special_Packages-Vault-Examine_RPK_Vault_Pack** und öffnen sie im Ordner "Default"

```

1  {
2    "FixedItems": [
3      "Weapon_RPK-74",
4      "ScopeRail_AK47",
5      "WeaponSights_V3_Holographic"
6    ],
7    "InitialDamage": 0,
8    "RandomDamage": 0,
9    "InitialUsage": 0,
10   "RandomUsage": 0
11 }

```

6. Wenn es dieses "Subpreset" **spawnen soll**, erhalten wir die **darin enthaltenen "FixedItems"**.
7. Um zu ändern, was sie enthalten, können Sie einfach eine Untervorgabe öffnen und die Elemente ändern.

Knoten

- Link zur vollständigen Verwendung von **Knotenpunkten** : [Beutebaumknoten](https://github.com/CheeseKing/Loot-Modifiers/blob/main/Nodes)
- Wir stellen euch zwei Möglichkeiten vor, wie ihr sie in euren Spawner-Voreinstellungen verwenden könnt.
- Kopieren und Einfügen des Links von **Nodes** für Ihre Spawner-Voreinstellung: <https://github.com/CheeseKing/Loot-Modifiers/blob/main/Nodes>
- Mehrere Knoten, getrennt nach Seltenheit: <https://github.com/CheeseKing/Loot-Modifiers/blob/main/MultipleNodes>

Gruppierung nach Seltenheit

- Du kannst Knoten nach Seltenheit gruppieren, wenn deine Spawner-Voreinstellung rollt, um zu bestimmen, welchen Knoten du bekommst, kannst du eine Reihe von ihnen zur

gleichen Seltenheit hinzufügen, zum Beispiel:

```
{
  "Nodes": [
    {
      "Rarity": "Uncommon",
      "Ids": [
        "ItemLootTreeNodes.Workshop.Tools",
        "ItemLootTreeNodes.Workshop.WorkClothes.Feet",
        "ItemLootTreeNodes.Workshop.WorkClothes.Torso",
        "ItemLootTreeNodes.Workshop.WorkClothes.Legs",
        "ItemLootTreeNodes.Workshop.Crafting",
        "ItemLootTreeNodes.Workshop.LightsFire"
      ]
    },
    {
      "Rarity": "Rare",
      "Ids": [
        "ItemLootTreeNodes.Trash"
      ]
    }
  ],
}
```

- Du hast eine viel größere Chance, die Knoten in der Seltenheitsstufe **"Ungewöhnlich"** zu erhalten, als die Knoten, die mit **der Seltenheit "Selten"** getrennt sind.
- Wenn Sie eine Anzahl von Knoten in einer Seltenheit haben, haben die Knoten darin immer noch ihre eigene Seltenheit, die Seltenheit, die wir in der Gruppierung angegeben haben, bestimmt nur die Seltenheit der gruppierten Knoten.
- In diesem Fall haben wir:
-Workshop.Tools, das ist "Uncommon",-Workshop.WorkClothes, das ist "Common"-Workshop.Crafting, das ist "Common"

```
  "Name": "tools",
  "Rarity": "Uncommon",
  "Children": [
    -Workshop.LightsFire
    "Name": "WorkClothes",
    "Rarity": "Common",
    "PostSpawnActions": [
      "SetClothesDirtiness_DirtyClothes"
    ],
    "Name": "Crafting",
    "Rarity": "Common",
    "Children": [
      {
```


, das ist "VeryRare"

```
"Name": "LightsFire",  
"Rarity": "VeryRare",  
"Children": [
```

- Nachdem unser Spawner-Preset diese Gruppe von Knoten ausgewählt hat, wird es immer noch die Seltenheiten der genannten Knoten im Inneren durchgehen, so dass je nach Seltenheit die Chancen auf bestimmte Gegenstände unterschiedlich sind.
- Um bestimmten Knoten gewünschte Seltenheiten zuzuordnen, sehen Sie sich bitte das folgende Beispiel an.

Trennung nach Seltenheit

- Sie können Knoten nach Seltenheit trennen, wie wir oben gezeigt haben, aber hier ist ein Beispiel für eine größere Anzahl von Knoten:

```
"Nodes": [  
  {  
    "Rarity": "Uncommon",  
    "Ids": [ "ItemLootTreeNodes.Workshop.Tools" ]  
  },  
  {  
    "Rarity": "Common",  
    "Ids": [ "ItemLootTreeNodes.Workshop.WorkClothes.Feet" ]  
  },  
  {  
    "Rarity": "Rare",  
    "Ids": [ "ItemLootTreeNodes.Workshop.WorkClothes.Torso" ]  
  },  
  {  
    "Rarity": "VeryRare",  
    "Ids": [ "ItemLootTreeNodes.Workshop.WorkClothes.Legs" ]  
  },  
  {  
    "Rarity": "Uncommon",  
    "Ids": [ "ItemLootTreeNodes.Workshop.Crafting" ]  
  },  
  {  
    "Rarity": "Uncommon",  
    "Ids": [ "ItemLootTreeNodes.Workshop.LightsFire" ]  
  }  
]
```

- Das Spawner-Preset würfelt, um zu bestimmen, welcher Knoten aus dieser Liste ausgewählt wird, je niedriger die Seltenheit, desto höher die Chancen, sie zu erhalten.
- Nachdem die Spawner-Voreinstellung ausgewählt hat, welchen Knoten du hier erhältst, kannst du nur Gegenstände erhalten, die in dem erwähnten Knoten enthalten sind.

Kombinierend

- Sie können Spawner-Vorgaben mit mehreren Methoden ändern.
- Das beste Beispiel dafür ist das, was wir bereits im Spiel haben, wir werden **Landscape-Examine_GroundRocks_Snow.json** öffnen, das sich hier befindet:
\Loot\Spawners\Presets\Default



```
1 {
2   "Items": [
3     {
4       "Rarity": "Abundant",
5       "Id": "Stone_Small"
6     },
7     {
8       "Rarity": "Common",
9       "Id": "Stone"
10    },
11    {
12      "Rarity": "Uncommon",
13      "Id": "Snowballs_01"
14    },
15    {
16      "Rarity": "VeryRare",
17      "Id": "Snowballs_02"
18    }
19  ],
20  "FixedItems": [
21    "Stone_Small",
22    "Snowballs_01"
23  ],
24  "QuantityMin": 1,
25  "QuantityMax": 2,
26  "AllowDuplicates": true,
27  "ShouldFilterItemsByZone": false,
28  "InitialDamage": 0,
29  "RandomDamage": 0,
30  "InitialUsage": 0,
31  "RandomUsage": 0
32 }
```

- Diese Spawner-Voreinstellung besagt, dass Sie immer "**Stone_Small**", "Snowballs_01" **erhalten**, wie in den "**FixedItems**" **angegeben**. Sie erhalten auch 1 bis 2 Artikel aus dem Abschnitt "**Artikel**", wie in "QuantityMin" **und** "QuantityMax" **angegeben** :
- Sie können beliebig viele Methoden zu jedem Spawner-Preset hinzufügen, am bequemsten ist es, "**FixedItems**" mit "**Items**"/"**Subpresets**"/**Nodes** zu kombinieren.

Landschafts-, Marionetten-, Fracht- und Farming-Drops:

- Wenn du Spawn-Voreinstellungen für bestimmte Orte exportierst, erhältst du keine Spawn-Voreinstellungen für **Marionetten-**, Fracht-, Landwirtschafts- und **Wachposten und auch keine Landschafts-Spawner-Voreinstellungen** wie Steine
 - Um diese Spawner-Voreinstellungen zu ändern, musst du sie aus dem Ordner `\Loot\Spawners\Presets\Default` entfernen und an den Ort hinzufügen, den du extrahiert hast (Wenn du sie zum Override hinzufügst, werden sie sich global ändern)
 - **Achtung**, es gibt viele dieser Spawner-Voreinstellungen. Ein Beispiel: Es gibt **53 Marionetten-Spawner-Voreinstellungen**, wenn Sie möchten, dass Ihre Marionetten nichts oder einen bestimmten Gegenstand fallen lassen. Sie müssen sie alle ändern
1. Wechseln Sie zu diesem Speicherort:
Server\SCUM\Saved\Config\WindowsServer\Loot\Spawners\Presets\Default
 2. Wir nehmen hier ein paar Voreinstellungen:
Landschaft-Examine_GroundRocks
Special_Packages-Cargo_Drops-Examine_ASVal_CargoDropCharacter-Puppen-
Militär-Examine_SK_Military_Zombie_03Farming-Examine_Cabbage
 3. Um sie zu ändern, müssen Sie das gesamte Preset nehmen und in den gewünschten Bereich kopieren und einfügen, in diesem Fall haben wir bereits den **A2-Sektor**
Server\SCUM\Saved\Config\WindowsServer\Loot\Spawners\Presets\Override\A2
extrahiert
 4. Wir werden diese im A2-Sektorordner ablegen und den Code, den wir in "FixedItems" verlinkt haben, für jede dieser Voreinstellungen verwenden
<https://github.com/CheeseKingu/Loot-Modifiers/blob/main/FixedItems>
 5. Jetzt, da wir sie durch die gewünschten "**FixedItems**" **ersetzt haben**, wird jeder einzelne Stein, den wir im A2-Sektor suchen, einen M82A1, einen MK18 und ein Katana fallen lassen
 6. Wenn Sie nach einem Military_Zombie_03 suchen, werden auch diese Elemente gelöscht
 7. Wenn Sie die **ASVal_Cargodrop** Voreinstellung finden, erhalten Sie auch diese 3 Elemente
 8. Jeder **Kohl**, den du mit dem Farmen anbaust, lässt diese 3 Gegenstände in ihrem Reifestadium fallen, so dass du im Grunde jeden Gegenstand, den du hineinlegst, anbauen kannst
 9. Du kannst jede Art von Beutemodifikator für diese Spawner-Voreinstellungen verwenden, wie oben in den Spawner-Voreinstellungen [modifizieren](#)

Aktionen nach dem Spawnen

- Post-Spawn-Aktionen werden in der Zeile "**PostSpawnActions**" **hinzugefügt**, wenn wir die Gegenstände, die fallen gelassen werden, ändern möchten
- Link eines Spawner-Presets mit mehreren "**PostSpawnActions**" :
<https://github.com/CheeseKingu/Loot-Modifiers/blob/main/PostSpawnActions>

Liste der Post-Spawn-Aktionen:

- **AbandonedBunkerKeycard** : Wenn es sich bei dem Gegenstand um eine Schlüsselkarte handelt, weisen Sie zu, dass er den nächstgelegenen Bunker öffnen kann.
- **SetAmmoAmount_BigStash - Wenn es sich** bei dem Gegenstand um Munition handelt, wird die Munitionsanzahl auf 50-100% der Kapazität des Kalibers gesetzt (Beispiel: cal_22 maximale Anzahl 20 ist, ist es 10-20/20).
- **SetAmmoAmount_SmallStash** - Wenn es sich bei dem Gegenstand um Munition handelt, wird die Munitionsanzahl auf 0-35% Kapazität des Kalibers gesetzt (Beispiel: cal_22 maximale Anzahl 20 beträgt, ist sie 0-7/20).
- **SetCashAmount_BigStash** - Wenn es sich bei dem Gegenstand um Bargeld handelt, wird der Wert auf 200-500 festgelegt.
- **SetCashAmount_MediumStash** - Wenn es sich bei dem Artikel um Bargeld handelt, wird der Wert auf 50-200 festgelegt.
- **SetCashAmount_SmallStash** - Wenn es sich bei dem Artikel um Bargeld handelt, wird der Wert auf 1-100 festgelegt.
- **SetClothesDirtiness_DeadPuppets** - wenn es sich bei dem Artikel um Kleidung handelt, wird der Verschmutzungsgrad auf 93-96% eingestellt
- **SetClothesDirtiness_DirtyClothes** - wenn es sich um Kleidung handelt, wird der Verschmutzungsgrad auf 60-85% eingestellt
- **SetClothesDirtiness_ResidentialClothes** - Wenn es sich bei dem Artikel um Kleidung handelt, wird die Verschmutzung auf 0-20% eingestellt
- **SetUsage_Max - Alle Gegenstände mit Verwendungen** spawnen mit 0 Verwendungen, z. B. Wasserflasche mit 0/5 Verwendungen, Haustierflasche mit **0/20** usw.

Beispiel für Post-Spawn-Aktionen:

1. Wir nehmen **Street-Residential-Examine_Car_Wreck_Scrap** als Beispiel:

```
Street-Residential-Examine_Crate_Tarp_Red.json
1  {
2      "Nodes": [
3          {
4              "Rarity": "Uncommon",
5              "Ids": [
6                  "ItemLootTreeNodes.Barn.Tools",
7                  "ItemLootTreeNodes.Barn.Crafting",
8                  "ItemLootTreeNodes.Barn.LightsFire",
9                  "ItemLootTreeNodes.Barn.WorkClothes"
10             ]
11         }
12     ],
13     "Probability": 15,
14     "QuantityMin": 1,
15     "QuantityMax": 2,
16     "AllowDuplicates": false,
17     "ShouldFilterItemsByZone": true,
18     "InitialDamage": 0,
19     "RandomDamage": 50,
20     "InitialUsage": 0,
21     "RandomUsage": 50,
22     "PostSpawnActions": [
23         "SetClothesDirtiness_ResidentialClothes"
24     ]
25 }
```

2. Diese Spawner-Voreinstellung besagt, dass, wenn der Gegenstand, der gedroppt wird, Kleidung ist, die Verschmutzung des Gegenstands auf 0-20% Verschmutzung gesetzt wird, wie oben erklärt.

3. Ein weiteres Beispiel für die Verwendung mehrerer **"PostSpawnActions"** in diesem Spawner-Preset: **Buildings-Garage-Residential-Examine_Wardrobe_Locker**:

```
Buildings-Garage-Residential-Examine_Wardrobe_Locker.json x
1 {
2   "Nodes": [
3     {
4       "Rarity": "Uncommon",
5       "Ids": [
6         "ItemLootTreeNodes.Garage.WorkClothes",
7         "ItemLootTreeNodes.Garage.Other.Ammo",
8         "ItemLootTreeNodes.Garage.Other.Backpacks.WaistBags",
9         "ItemLootTreeNodes.Garage.Other.Backpacks.School",
10        "ItemLootTreeNodes.Garage.Other.Backpacks.Plain"
11      ]
12    }
13  ],
14  "Probability": 20,
15  "QuantityMin": 1,
16  "QuantityMax": 1,
17  "AllowDuplicates": false,
18  "ShouldFilterItemsByZone": true,
19  "InitialDamage": 10,
20  "RandomDamage": 30,
21  "InitialUsage": 1,
22  "RandomUsage": 30,
23  "PostSpawnActions": [
24    "SetAmmoAmount_SmallStash",
25    "SetCashAmount_SmallStash",
26    "SetClothesDirtiness_DirtyClothes"
27  ]
28 }
```

4. **SetAmmoAmount_SmallStash** - Wenn es sich bei dem Gegenstand um Munition handelt, stellen Sie die Munitionsanzahl auf 0-35% Kapazität des Kalibers ein.
5. **SetCashAmount_SmallStash** - Wenn es sich bei dem Artikel um Bargeld handelt, legen Sie den Wert auf 1-100 fest.
6. **SetClothesDirtiness_DirtyClothes** - wenn es sich um Kleidung handelt, wird der Verschmutzungsgrad auf 60-85% eingestellt

Verzichtserklärung

Links von Drittanbietern in diesem Dokument können Sie zu Websites von Drittanbietern führen, die nicht mit uns verbunden sind. Wir sind nicht dafür verantwortlich, den Inhalt oder die Richtigkeit zu prüfen oder zu bewerten, und wir übernehmen keine Garantie und übernehmen keine Haftung oder Verantwortung für Materialien oder Websites Dritter oder für andere Materialien, Produkte oder Dienstleistungen Dritter.

Wir haften nicht für Schäden oder Schäden im Zusammenhang mit dem Kauf oder der Nutzung von Waren, Dienstleistungen, Ressourcen, Inhalten oder anderen Transaktionen, die in Verbindung mit Websites Dritter getätigt werden. Bitte lesen Sie die Richtlinien und Praktiken des Drittanbieters sorgfältig durch und stellen Sie sicher, dass Sie diese verstehen, bevor Sie eine Transaktion durchführen. Beschwerden, Ansprüche, Bedenken oder Fragen zu Produkten von Drittanbietern sollten an den Drittanbieter gerichtet werden.