

FS25_ObjectStorageVisuals - Dokumentation

Kurzanleitung für den Einbau in ein GIANTS-objectStorage.

Stand 1.0.0.0

1. Voraussetzung

FS25_ObjectStorageVisuals (OSV) erweitert GIANTS-objectStorage um zusätzliche sichtbare Darstellungen und optionale Modder-Funktionen.

Die Funktionen sind bewusst nach Lagertyp getrennt.

Lagertyp	XML-Erkennung	Erweiterungsblock
Reines Ballenlager	supportsBales="true" und supportsPallets="false"	baleStorageVisuals
Reines Palettenlager	supportsBales="false" und supportsPallets="true"	palletStacking
Gemischtes Lager	supportsBales="true" und supportsPallets="true"	keiner

Reine Ballenlager werden über baleStorageVisuals vorbereitet:

```
<objectStorage supportsBales="true" supportsPallets="false" capacity="110">
...
  <baleStorageVisuals enabled="true">
    ...
  </baleStorageVisuals>
</objectStorage>
```

Reine Palettenlager werden über palletStacking vorbereitet, wenn zusätzliche Einstellungen benötigt werden:

```
<objectStorage supportsBales="false" supportsPallets="true" capacity="250">
...
  <palletStacking dynamicCapacity="true" hardLimit="500" />
</objectStorage>
```

Gemischte Lager werden nicht für diese Erweiterungen vorbereitet. Sie bleiben beim normalen GIANTS-objectStorage-Verhalten.

Die Ausrichtung der Lagerplätze sollte grundsätzlich über die storageArea-Nodes in der i3d gelöst werden.

2. baleStorageVisuals

Eintrag innerhalb von objectStorage.

```
<baleStorageVisuals enabled="true" dynamicCapacity="true" hardLimit="500">
...
</baleStorageVisuals>
```

Parameter	Default	Funktion
enabled	false	Aktiviert Ballen-Visuals für diesen Storage.
dynamicCapacity	false	Prüft beim Einlagern den tatsächlich sichtbaren Platz in den storageAreas.
hardLimit	500	Technische Obergrenze bei aktiver dynamicCapacity.

baleStorageVisuals ist für reine Ballenlager vorgesehen.

2.1 storageArea-Gruppen für Ballen

Reine Ballenlager können ihre storageAreas optional nach Ballengruppen aufteilen. Dafür kann an einer storageArea das Attribut baleGroups gesetzt werden.

```
<objectStorage supportsBales="true" supportsPallets="false" capacity="650">
  <storageAreas>
    <storageArea startNode="storageAreaStartHay" endNode="storageAreaEndHay" maxHeight="4.6" baleGroups="hay"/>
    <storageArea startNode="storageAreaStartStraw" endNode="storageAreaEndStraw" maxHeight="4.6"
baleGroups="straw"/>
    <storageArea startNode="storageAreaStartWrapped" endNode="storageAreaEndWrapped" maxHeight="4.6"
baleGroups="wrapped"/>
    <storageArea startNode="storageAreaStartFallback" endNode="storageAreaEndFallback" maxHeight="4.6"/>
  </storageAreas>
</objectStorage>
```

Gruppe	Bedeutung
hay	trockene Heu-/Trockenfutter-Ballen
straw	Strohballen
wrapped	gewickelte Ballen

Mehrere Gruppen pro storageArea sind möglich:

```
<storageArea startNode="..." endNode="..." maxHeight="4.6" baleGroups="hay straw"/>
```

Eine storageArea ohne baleGroups dient als Rest- bzw. Fallbackfläche. Sie kann Ballen aufnehmen, die keiner passenden Gruppenfläche zugeordnet werden können.

Gewickelte Ballen haben Vorrang vor der FillType-Gruppe. Ein gewickelter Gras-, Heu-, Klee- oder Luzerneballen gehört deshalb zur Gruppe wrapped.

Ungewickeltes Gras wird keiner der Gruppen hay, straw oder wrapped zugeordnet. Solche Ballen benötigen eine ungruppierte Fallback-storageArea, wenn sie angenommen werden sollen.

Die Gruppierung ist optional. Ohne baleGroups bleibt das bisherige Verhalten erhalten.

Fehlende map- oder modabhängige FillTypes werden ignoriert. Dadurch bleibt die Funktion map-safe und erzeugt keine zusätzlichen Log-Warnings.

Bei aktiver dynamicCapacity werden die Gruppenflächen auch für die Annahmeprüfung genutzt. Ein Heuballen wird also nicht einfach in eine reine Strohfläche eingelagert, nur weil dort noch Platz wäre. Wenn eine passende Gruppenfläche voll ist, kann nur eine ungruppierte Fallbackfläche übernehmen. Gibt es keine passende Fläche mehr, wird die Einlagerung verweigert.

3. slotTrigger für reine Ballenlager

Für reine Ballenlager gibt es zwei Varianten: einen einzelnen slotTrigger und mehrere gruppenspezifische slotTrigger über den slotTriggers-Block.

Für gruppierte Ballenlager wird der slotTriggers-Block empfohlen.

3.1 Einzelner slotTrigger

```
<slotTrigger enabled="true" movePlayerTrigger="true" playerTriggerOffset="0 0 0" />
```

Parameter	Default	Funktion
enabled	false	Aktiviert den beweglichen SlotTrigger.
movePlayerTrigger	false	Bewegt den PlayerTrigger mit dem SlotTrigger.
playerTriggerOffset	0 0 0	Offset des PlayerTriggers relativ zum SlotTrigger in Metern.

Der passende playerTriggerOffset hängt vom jeweiligen Lager und der i3d-Struktur ab.

Bei aktivem slotTrigger wird der normale objectTrigger ersetzt und ausgeblendet. Sichtbare Marker oder Warning-Stripes sollten deshalb nicht als Child im alten objectTrigger liegen, wenn sie sichtbar bleiben sollen.

Der einzelne slotTrigger ist für klassische Lager mit einem wandernden Einlagerpunkt gedacht. movePlayerTrigger und playerTriggerOffset gelten nur für diesen Einzelblock.

Lagerzustand	Triggerposition
Lager leer	Start der ersten storageArea
Lager teilweise belegt	nächster freier Lagerplatz
Lager vollständig belegt	letzter belegter Lagerplatz

3.2 Gruppenspezifische slotTriggers

```
<baleStorageVisuals enabled="true" dynamicCapacity="true" hardLimit="650">
  <slotTriggers>
    <slotTrigger baleGroups="hay" />
    <slotTrigger baleGroups="straw" />
    <slotTrigger baleGroups="wrapped" />
  </slotTriggers>
</baleStorageVisuals>
```

Parameter	Default	Funktion
enabled	true	Aktiviert oder deaktiviert einen einzelnen Gruppen-SlotTrigger.
baleGroups	-	Beschränkt den Trigger auf eine oder mehrere Ballengruppen.

Ein Eintrag mit baleGroups="hay" nimmt Heuballen an. Ein Eintrag mit baleGroups="straw" nimmt Strohballen an. Ein Eintrag mit baleGroups="wrapped" nimmt gewickelte Ballen an.

Ein Eintrag ohne baleGroups ist ein Fallback-/Rest-Trigger, zum Beispiel für ungewickeltes Gras oder nicht erkannte Ballengruppen.

Mit enabled="false" kann ein einzelner Trigger bewusst deaktiviert werden.

Die Triggerposition wird aus den vorhandenen storageAreas berechnet. Es müssen keine separaten Trigger-Nodes im XML angegeben werden.

Gruppierte Trigger wandern zur nächsten passenden Position ihrer Gruppe. Wenn die passende Gruppenfläche voll ist und eine ungruppierte Fallback-storageArea vorhanden ist, darf der gruppierte Trigger diese Fallbackfläche nutzen.

Bei mehreren gruppenspezifischen slotTriggern gibt es nicht mehrere GIANTS-playerTrigger. Der GIANTS-playerTrigger bleibt der zentrale Bedienpunkt für das Lagermenü.

4. palletUnderlays

Optionaler Eintrag innerhalb von baleStorageVisuals.

```
<palletUnderlays enabled="true"
  positionVariation="0.025"
  rotationVariation="1.5"
  rotationY="0" />
```

Parameter	Default	Funktion
enabled	false	Zeigt Unterlegpaletten unter belegten Ballenflächen.
positionVariation	0.025	Leichte Positionsabweichung der Unterlegpaletten.
rotationVariation	1.5	Leichte Drehabweichung der Unterlegpaletten in Grad.
rotationY	0	Feste Y-Drehung der Unterlegpaletten, zum Beispiel 90.

Wenn palletUnderlays zusammen mit dynamicCapacity genutzt wird, prüft OSV zusätzlich, ob für neu belegte Bodenflächen auch eine passende Unterlegpaletten-Fläche vorhanden ist. Dadurch werden Ballen nicht über die letzte sichtbare Unterlegpalette hinaus angenommen.

5. palletStacks

Eintrag innerhalb von baleStorageVisuals.

```
<palletStacks>
  <palletStack linkNode="stackRef01"
    translation="0 0 0"
    rotation="0 0 0"
    maxPallets="20"
    collision="true" />
</palletStacks>
```

Parameter	Default	Funktion
linkNode	-	Pflicht. Referenznode für den jeweiligen Stapel.
translation	0 0 0	Position relativ zum linkNode.
rotation	0 0 0	Rotation relativ zum linkNode in Grad.
maxPallets	auto	Maximale Palettenanzahl dieses Stapels.
collision	false	Aktiviert die Kollision für diesen Stapel.

6. Palettenlager

Die Palettenfunktionen sind für reine Palettenlager vorgesehen.

Sichtbares Palettenstapeln ist für reine Palettenlager standardmäßig aktiv, sofern die jeweilige Palette stapelbar vorbereitet ist. enabled="true" muss deshalb nicht gesetzt werden.

Beispiel mit dynamischer Palettenkapazität:

```
<objectStorage supportsBales="false" supportsPallets="true" capacity="250">
  ...
  <palletStacking dynamicCapacity="true" hardLimit="500" />
</objectStorage>
```

Beispiel ohne sichtbares Palettenstapeln, aber mit nutzbarem SlotTrigger:

```
<objectStorage supportsBales="false" supportsPallets="true" capacity="250">
  ...
  <palletStacking enabled="false" dynamicCapacity="true" hardLimit="500">
    <slotTrigger enabled="true" movePlayerTrigger="true" playerTriggerOffset="0 0 0" />
  </palletStacking>
</objectStorage>
```

Parameter	Default	Funktion
enabled	true	Schaltet mit false nur das sichtbare Palettenstapeln in diesem Lager ab. dynamicCapacity und slotTrigger bleiben separat nutzbar.
dynamicCapacity	false	Prüft beim Einlagern, ob im sichtbaren Palettenlayout noch Platz vorhanden ist.
hardLimit	500	Technische Obergrenze bei aktiver dynamischer Palettenkapazität.
stackInMultiLevelStorage	false	Erlaubt sichtbares Palettenstapeln innerhalb von Regalen oder Etagenlagern mit mehreren storageAreas auf unterschiedlichen

		Höhen.
rotatePallets	false	Globaler Schalter für Palettenrotation.
rotateEuroPallets	false	Dreht erkannte Euro-Paletten um 90 Grad.
rotateBigBagPallets	false	Dreht erkannte BigBag-Paletten um 90 Grad.
rotateSquarePallets	false	Dreht erkannte quadratische Paletten um 90 Grad, zum Beispiel IBC-/Tankpaletten.
rotateLargePallets	false	Dreht erkannte große bzw. überlange Paletten um 90 Grad.
rotateBigBags	false	Dreht reine BigBags um 90 Grad.

Die dynamische Palettenkapazität ersetzt nur die einfache Stückzahlgrenze des GIANTS-objectStorage. FillTypes, Farmberechtigungen und die sonstigen GIANTS-Prüfungen bleiben erhalten.

Gemischte Lager mit Ballen und Paletten werden nicht über palletStacking vorbereitet. Sie bleiben beim normalen GIANTS-objectStorage-Verhalten.

6.1 Palettenrotation

OSV kann Paletten für die sichtbare Darstellung optional um 90 Grad drehen. Die Rotation ist eine Modder-Option für das sichtbare Layout und die Rack-/Area-Anpassung.

Sie ersetzt keine freie Lagerrotation und macht unpassende storageAreas nicht automatisch passend.

```
<palletStacking
  dynamicCapacity="true"
  hardLimit="60"
  rotatePallets="true"
  rotateBigBagPallets="false" />
```

Parameter	Default	Funktion
rotatePallets	false	Globaler Schalter für Palettenrotation.
rotateEuroPallets	false	Dreht erkannte Euro-Paletten.
rotateBigBagPallets	false	Dreht erkannte BigBag-Paletten.
rotateSquarePallets	false	Dreht erkannte quadratische Paletten, zum Beispiel IBC-/Tankpaletten.
rotateLargePallets	false	Dreht erkannte große bzw. überlange Paletten.
rotateBigBags	false	Dreht reine BigBags.

Wenn rotatePallets fehlt oder false ist, rotieren nur Gruppen, deren eigener Gruppenschalter explizit true ist.

Wenn rotatePallets="true" gesetzt ist, rotieren grundsätzlich alle unterstützten Palettengruppen. Einzelne Gruppen können dann über ihren eigenen Gruppenschalter wieder ausgeschlossen werden, wenn der jeweilige Gruppenschalter explizit auf false gesetzt ist.

```
<palletStacking
  rotatePallets="true"
  rotateBigBagPallets="false"
  rotateSquarePallets="true" />
```

In diesem Beispiel werden grundsätzlich Paletten gedreht. BigBag-Paletten werden aber ausdrücklich ausgeschlossen. Quadratische Paletten bleiben ausdrücklich erlaubt.

Die Rotation muss immer zur tatsächlich definierten storageArea passen. OSV verkleinert keine Objektmaße künstlich. Wenn eine gedrehte Palette nicht vollständig in die storageArea passt, sollte die entsprechende Rotation deaktiviert oder die storageArea angepasst werden.

6.2 Regale und Etagenlager

Wenn ein Lager mehrere storageAreas auf unterschiedlichen Y-Höhen nutzt, behandelt OSV das Lager als Multi-Level-Lager. Sichtbares Palettenstapeln wird in solchen Lagern standardmäßig deaktiviert, damit Paletten nicht ungewollt innerhalb einzelner Regalfächer zusätzlich übereinander gestapelt werden.

Der aktuelle Grenzwert für die automatische Multi-Level-Erkennung liegt bei 0.35 m Höhenunterschied zwischen den storageArea-Startnodes.

Soll ein reines Palettenlager bewusst als Regal mit sichtbarer Rack-Logik genutzt werden, kann dies aktiviert werden:

```
<palletStacking
  dynamicCapacity="true"
  hardLimit="60"
  stackInMultiLevelStorage="true" />
```

Bei `stackInMultiLevelStorage="true"` behandelt OSV jede `storageArea` als eigenes Regalfach bzw. Regalsegment.

`DynamicCapacity` und `Visual-Rebuild` nutzen dabei dieselbe Rack-Slot-Logik. Dadurch wird verhindert, dass das Lager „voll“ meldet, obwohl optisch noch sinnvolle Regalplätze frei sind, oder umgekehrt Paletten unsichtbar in nicht nutzbare Flächen eingelagert werden.

Gleiche Paletten werden bevorzugt auf bestehende passende Stapel bzw. Slots gelegt. Danach werden freie passende Regalplätze genutzt. Das ist besonders wichtig für gemischte Regale mit IBCs, Boxpaletten, Verbrauchsmaterial-Paletten und BigBagPallets.

6.3 Beispiel für ein Palettenregal

```
<objectStorage supportsBales="false" supportsPallets="true" capacity="60">
  <storageAreas>
    <storageArea startNode="shelf01Start" endNode="shelf01End" maxHeight="1.65" />
    <storageArea startNode="shelf02Start" endNode="shelf02End" maxHeight="1.65" />
    <storageArea startNode="shelf03Start" endNode="shelf03End" maxHeight="1.65" />
  </storageAreas>

  <palletStacking
    dynamicCapacity="true"
    hardLimit="60"
    stackInMultiLevelStorage="true"
    rotateEuroPallets="true"
    rotateBigBagPallets="false"
    rotateSquarePallets="true" />
</objectStorage>
```

In diesem Beispiel ist sichtbares Stapeln innerhalb der Regalfächer erlaubt. Euro-Paletten und quadratische Paletten dürfen gedreht werden. BigBag-Paletten bleiben ungedreht, weil sie in vielen Regalen gedreht zu tief oder zu breit für das Fach sein können.

Modder-Hinweise für Regale:

- `storageAreas` sauber pro Fach oder Ebene anlegen.
- `storageAreas` nicht unnötig überlappen lassen.
- `maxHeight` pro Ebene realistisch auf die Fachhöhe setzen.
- X/Z-Ausrichtung der Areas passend zur gewünschten Regaltiefe und Regalbreite setzen.
- Pfosten, Wände, Regalböden und andere Kollisionen nicht durch `storageAreas` schneiden lassen.
- `maxWidth` und `maxLength` am `objectStorage` können weiterhin genutzt werden, um ungeeignete Paletten über deren Basegame-Objektmaße auszuschließen.

Die Größenprüfung über `maxWidth` und `maxLength` erfolgt vor der optischen Rotation bzw. vor der Rack-Slot-Visualisierung.

IBCs haben im Basegame ungefähr 1.35 x 1.35. BigBagPallets haben im Basegame ungefähr 1.50 x 1.20.

```
<objectStorage maxWidth="1.40" maxLength="1.40">
```

Ein Regal mit diesem Filter lässt IBCs zu, sperrt BigBagPallets aber über die grundsätzliche Größenprüfung aus.

6.4 `storageArea#maxHeight` bei Palettenlagern

Vanilla nutzt `storageArea#maxHeight` bei Palettenlagern nur begrenzt, da Paletten im normalen GIANTS-`objectStorage` nicht sichtbar gestapelt werden.

OSV nutzt diesen Wert bei aktivem `palletStacking` als sichtbare Stapelgrenze. Bei Regalen mit `stackInMultiLevelStorage="true"` beschreibt `maxHeight` die maximale Stapelhöhe innerhalb des jeweiligen Regalfachs.

Der Wert sollte deshalb bewusst auf die reale Fachhöhe abgestimmt werden.

```
<storageArea startNode="shelf01Start" endNode="shelf01End" maxHeight="1.65" />
```

Wenn ein Regalfach nur eine Lage aufnehmen soll, muss maxHeight entsprechend niedrig gesetzt werden. Wenn zwei niedrige Palettenlagen möglich sein sollen, muss maxHeight entsprechend höher gesetzt werden.

6.5 Annahmefilter, optischer Footprint und Stackhöhe

OSV unterscheidet zwischen drei Dingen: Annahmefilter des objectStorage, optischem Footprint für die sichtbare Platzierung und Stackhöhe bzw. erlaubten Stapellagen.

Die grundsätzliche Annahmeprüfung nutzt weiterhin die Basegame-/Store-Größe der Palette. Dazu gehören zum Beispiel objectStorage#maxWidth, objectStorage#maxLength und objectStorage#maxHeight.

Die OSV-Stackdaten beeinflussen die sichtbare Darstellung, die genutzte Stellfläche im Visual-Rebuild, die Stackhöhe, die maximale Lagenzahl und die Rack-Slot-Belegung.

Eine Palette kann über die normale Größenprüfung des objectStorage ausgeschlossen werden, auch wenn ihre optischen Stackdaten kleiner oder anders gesetzt sind.

Bei Regalen kann es sinnvoll sein, bestimmte Palettentypen bewusst über maxWidth oder maxLength auszuschließen, wenn sie optisch nicht zum Regal passen.

7. slotTrigger für reine Palettenlager

Optionalen Eintrag innerhalb von palletStacking.

```
<objectStorage supportsBales="false" supportsPallets="true" capacity="250">
  ...
  <palletStacking dynamicCapacity="true" hardLimit="500">
    <slotTrigger enabled="true" movePlayerTrigger="true" playerTriggerOffset="0 0 0" />
  </palletStacking>
</objectStorage>
```

Parameter	Default	Funktion
enabled	false	Aktiviert den beweglichen SlotTrigger für reine Palettenlager.
movePlayerTrigger	false	Bewegt den PlayerTrigger mit dem SlotTrigger.
playerTriggerOffset	0 0 0	Offset des PlayerTriggers relativ zum SlotTrigger in Metern.

Der Paletten-SlotTrigger nutzt die sichtbare Palettenplatzierung des Lagers.

Lagerzustand	Triggerposition
Lager leer	Start der ersten storageArea
Lager teilweise belegt	nächster freier Palettenplatz
Lager vollständig belegt	letzter belegter Palettenplatz

Der passende playerTriggerOffset hängt vom jeweiligen Lager und der i3d-Struktur ab.

Der Paletten-slotTrigger kann auch genutzt werden, wenn das sichtbare Palettenstapeln für ein Lager deaktiviert wurde, zum Beispiel bei Regalen mit eigenen storageArea-Ebenen.

Bei Lagern, deren Lagerbereich betreten werden soll, dürfen blockierende Kollisionen im Lagerbereich nicht im Weg stehen.

8. objectStorageStacking für Paletten aus externen Mods

Dieser Block gehört in die XML der jeweiligen Palette aus einem externen Mod. Er wird nicht im objectStorage des Lagers eingetragen.

Einfaches Beispiel in der Paletten-XML:

```
<objectStorageStacking enabled="true" />
```

Erweitertes Beispiel in der Paletten-XML:

```
<objectStorageStacking
  enabled="true"
  stackHeight="0.82"
  stackWidth="1.24"
  stackLength="0.84"
  maxStackLayers="2"
  stackFullOnly="true"
  fullFillLevel="1000" />
```

Parameter	Default	Funktion
enabled	false	Erlaubt sichtbare Stapelung dieser Palette im ObjectStorage.
stackHeight	StoreData	Vertikaler Stapelabstand in Metern.
stackWidth	StoreData	Breite der genutzten Visual-Stellfläche in Metern.
stackLength	StoreData	Länge der genutzten Visual-Stellfläche in Metern.
maxStackLayers	unbegrenzt	Maximale sichtbare Stapellagen.
stackFullOnly	false	Stapelt nur volle Paletten.
fullFillLevel	automatisch	Referenzwert für volle Paletten.
palletGroup	automatisch	Überschreibt die Palettengruppe für Rotation und Footprint-Zuordnung.

stackHeight ist ein Meterwert. maxStackLayers ist die Anzahl der erlaubten Lagen.

stackFullOnly="false" erlaubt Stapel aus teilgefüllten Paletten gleicher Füllmenge. Je nach Modell kann das optisch kollidieren.

palletGroup beeinflusst nur die Gruppenzuordnung für Rotation und Footprint-Logik. Der Eintrag aktiviert nicht automatisch sichtbares Stacking.

Erlaubte palletGroup-Werte
euroPallet
bigBagPallet
squarePallet
largePallet
bigBag

Beispiel für reine Gruppenzuordnung ohne aktiviertes Stacking:

```
<objectStorageStacking palletGroup="squarePallet" />
```

Soll eine Palette zusätzlich sichtbar stapelbar sein, muss enabled="true" gesetzt und die Palette entsprechend vorbereitet werden:

```
<objectStorageStacking
  enabled="true"
  palletGroup="euroPallet"
  stackHeight="0.82"
  maxStackLayers="2" />
```

Dieser Eintrag ist für Modder gedacht, die eigene Paletten aus ihrem Mod für die sichtbare Stapelung im ObjectStorage vorbereiten möchten.

Nur echte GIANTS-objectStorage-Paletten werden unterstützt. Keine Dekoobjekte, Placeables oder Sonderobjekte.

9. Verbrauchsmaterial-Paletten und Varianten

OSV korrigiert die Client-Darstellung bestimmter Basegame-Verbrauchsmaterial-Paletten im ObjectStorage.

Betroffene Basegame-Palette	Beschreibung
baleTwinePallet	Bindegarn-Palette
baleNetPallet	Rundballennetz-Palette
raniWrapPallet	Wickelfolie-Palette

Im Multiplayer werden Bindegarn, Ballennetz und Wickelfolie im ObjectStorage auf Clients wieder sichtbar dargestellt, statt nur als leere Palette zu erscheinen.

Konfigurierte Verbrauchsmaterial-Paletten werden nicht mehr nur nach ihrer Grundpalette zusammengeworfen. Varianten, Marken oder Farben bleiben für die sichtbare Darstellung erhalten, soweit sie aus der Paletten-/Consumable-Konfiguration auflösbar sind.

Die Darstellung bleibt nach Einlagern, Savegame-Reload und Multiplayer-Synchronisation erhalten.

Der Auslagerungsdialog unterscheidet konfigurierte Palettenvarianten besser, wenn ein echter Variantentext auflösbar ist. Bei mehreren Varianten derselben Palette werden dadurch nicht mehr nur identische generische Einträge angezeigt, sondern erkennbare Varianten-/Markeninformationen ergänzt.

9.1 FS25_moreWrapColors

OSV unterstützt FS25_moreWrapColors automatisch.

MoreWrapColors-Wickelfolienpaletten werden im ObjectStorage visuell korrekt dargestellt. Die zusätzlichen Farben funktionieren im Singleplayer, im Selfhost/Host-MP und auf Dedicated-Server-Clients.

Berücksichtigt werden volle MoreWrapColors-Wickelfolienpaletten, halbe MoreWrapColors-Wickelfolienpaletten und zusätzliche Wickelfolienfarben aus der Consumable-Konfiguration.

10. Kompatibilität / Crossplay

Die OSV-Einträge sind optionale Erweiterungen.

Ohne OSV-Script bleibt das normale GIANTS-objectStorage nutzbar. Nur die zusätzlichen Visual-, SlotTrigger- und Kapazitätsfunktionen fehlen.

Bereits eingelagerte Objekte werden durch OSV nicht verändert, gelöscht oder migriert. OSV beeinflusst die Annahme neuer Objekte und die sichtbare Darstellung, nicht den gespeicherten Bestand.

Hinweis zu bestehenden Spielständen

dynamicCapacity verändert keine bereits eingelagerten Objekte.

Wenn dynamicCapacity nachträglich aktiviert wird oder sich storageAreas, hardLimit-Werte oder sichtbare Lagerflächen ändern, bleiben vorhandene Objekte im Lagerbestand erhalten. Ein Lager kann dadurch vorübergehend mehr Objekte enthalten, als nach der neuen sichtbaren Kapazitätsprüfung neu eingelagert werden könnten.

Auslagern bleibt möglich. Neue Objekte können erst wieder eingelagert werden, sobald die dynamische Kapazitätsprüfung freien sichtbaren Lagerplatz erkennt.

Die OSV-XML-Blöcke ändern für sich genommen nicht die Crossplay-Fähigkeit. Entscheidend sind Scripts und Assets des jeweiligen Lagers.

11. FAQ / Typische Einbaufehler

Warum kann ich nichts mehr einlagern, obwohl die normale Kapazität noch nicht erreicht ist?

Bei aktiver dynamicCapacity zählt nicht nur die technische objectStorage#capacity. OSV prüft zusätzlich, ob für das nächste Objekt noch sichtbarer Platz in den definierten storageAreas vorhanden ist. Wenn die sichtbare Fläche voll ist, wird die Einlagerung blockiert, auch wenn die technische Kapazität noch höher ist.

Was ist der Unterschied zwischen capacity, dynamicCapacity und hardLimit?

capacity ist die normale GIANTS-Stückzahlgrenze des objectStorage. dynamicCapacity ist die zusätzliche Prüfung des sichtbaren Platzes in den storageAreas. hardLimit ist eine zusätzliche technische Obergrenze für Lager mit aktiver dynamicCapacity.

Warum bleibt ein Lager nach einem Update überfüllt?

Bereits eingelagerte Objekte werden nicht gelöscht oder gekürzt. Wenn ein Lager nach einem Update mehr Objekte enthält, als die neue dynamische Prüfung zulassen würde, bleibt dieser Bestand erhalten. Auslagern funktioniert weiterhin. Neue Einlagerung ist erst wieder möglich, wenn wieder genug sichtbarer Lagerplatz frei ist.

Warum stapeln Paletten in meinem Regal nicht sichtbar übereinander?

Sichtbares Palettenstapeln wird in Multi-Level-Lagern standardmäßig deaktiviert. Dadurch werden Regale oder Etagenlager nicht versehentlich innerhalb jedes Fachs zusätzlich gestapelt. Wenn Stapeln im Regal gewünscht ist, muss `stackInMultiLevelStorage="true"` gesetzt werden.

Warum wird eine Palette trotz aktivierter Rotation nicht eingelagert?

Die gedrehte Palette muss mit ihren echten Objektmaßen vollständig in die definierte storageArea passen. OSV verkleinert keine Paletten künstlich. Wenn eine Palette gedreht zu breit oder zu tief für ein Fach ist, wird sie nicht passend gerechnet.

Warum stehen Paletten in Pfosten, Wänden oder Regalteilen?

OSV kennt nur die definierten storageAreas, nicht die komplette Geometrie des Gebäudes. Die storageAreas müssen so gesetzt werden, dass sie nicht durch Pfosten, Wände, Regalböden oder andere Kollisionen laufen. Bei Regalen sollte jedes Fach eine eigene, exakt begrenzte storageArea bekommen.

Warum wird neben einer gestapelten Palette noch eine andere Palette in dasselbe Fach gelegt?

Wenn `stackInMultiLevelStorage="true"` aktiv ist, nutzt OSV den freien Restplatz innerhalb eines Regalfachs weiter, solange das nächste Objekt vollständig in die storageArea passt. Das Fach wird nicht pauschal für eine Objektgruppe reserviert.

Warum stehen manche Palettentypen nebeneinander und andere nicht?

OSV nutzt die echten Objektmaße und die definierte storageArea. Unterschiedliche Palettentypen können unterschiedliche Breite, Länge und Höhe haben. Was bei Euro-Paletten passt, muss bei BigBag-Paletten oder IBC-Paletten nicht automatisch passen.

Warum werden mehrere Wickelfolienfarben getrennt angezeigt?

OSV berücksichtigt die Palettenkonfigurationen bei der internen Gruppierung und bei der visuellen Darstellung. Dadurch bleiben Varianten, Farben oder Marken erhalten, soweit sie aus der Konfiguration auflösbar sind.

Warum passt eine Palette trotz Stackdaten nicht ins Regal?

Die grundsätzliche Annahmeprüfung nutzt weiterhin die Basegame-/Store-Maße des Objekts. OSV-Stackdaten steuern die sichtbare Platzierung, ersetzen aber nicht die normalen `maxWidth`-/`maxLength`-Prüfungen des `objectStorage`.

Kann ich gemischte Lager für Ballen und Paletten mit diesen Erweiterungen nutzen?

Nein. Die OSV-Erweiterungen `baleStorageVisuals` und `palletStacking` sind für reine Ballenlager oder reine Palettenlager vorgesehen. Gemischte Lager bleiben beim normalen GIANTS-objectStorage-Verhalten.

Was sollte ich als Modder in meine eigene Modbeschreibung aufnehmen?

Wenn ein Lager `dynamicCapacity` nutzt, sollte in der Modbeschreibung kurz erwähnt werden, dass die sichtbare Lagerfläche zusätzlich zur normalen Stückzahlkapazität begrenzt. Bestehende Bestände bleiben erhalten. Neue Einlagerung kann aber blockiert sein, bis wieder sichtbarer Lagerplatz frei ist.

12. Hinweistext für Modder-Modbeschreibungen

Dieser Text kann in die Beschreibung eines Lager-Mods übernommen werden, wenn das Lager `dynamicCapacity` nutzt:

Dieses Lager nutzt eine dynamische Kapazitätsprüfung. Bereits eingelagerte Objekte aus bestehenden Spielständen bleiben erhalten. Wenn das Lager durch ein Update rechnerisch oder optisch überfüllt ist, können Objekte weiterhin ausgelagert werden. Neue Einlagerung ist erst wieder möglich, sobald ausreichend sichtbarer Lagerplatz frei ist.