Configurable Products Setup

KatanaPIM for Magento 2

Setting up two-dimensional configurable products (like shoes with Color AND Size) in KatanaPIM Connect. This guide explains how to structure products in KatanaPIM with Parent and GrandParent relationships, and how to configure Magento to properly import them as configurable products with multiple variation attributes.

Understanding the Product Hierarchy

KatanaPIM uses a three-level hierarchy for products with multiple variation dimensions:

KatanaPIM Type	Example	Magento Product (GrandParent mode)
GrandParent	"Running shoes"	Configurable Product
Parent	"Running shoes - Red"	**No product created** (data only)
Child	"Running shoes - Red - 43"	Simple Product

Important: In the default GrandParent mode, Parent products appear in the KatanaPIM grid but **no Magento product is created** for them. Parents serve as data containers that hold the first variation attribute value (e.g., Color). This value is inherited by their Child products. See the Configurable Product Level section below for alternative modes where Parents become configurable products.

Visual example - Running shoes with Color and Size:

```
KatanaPIM Structure:
____________________

GrandParent: Running shoes (SKU: 000-RS)
├── Parent: Running shoes - Blue (SKU: 000-RS-B)      [Color: Blue]
│   ├── Child: Running shoes - Blue - 42 (SKU: 000-RS-B-42)  [Size: 42]
│   ├── Child: Running shoes - Blue - 43 (SKU: 000-RS-B-43)  [Size: 43]
│   └── Child: Running shoes - Blue - 44 (SKU: 000-RS-B-44)  [Size: 44]
├── Parent: Running shoes - Red (SKU: 000-RS-R)       [Color: Red]
│   ├── Child: Running shoes - Red - 42 (SKU: 000-RS-R-42)   [Size: 42]
│   ├── Child: Running shoes - Red - 43 (SKU: 000-RS-R-43)   [Size: 43]
│   └── Child: Running shoes - Red - 44 (SKU: 000-RS-R-44)   [Size: 44]
└── Parent: Running shoes - Yellow (SKU: 000-RS-Y)    [Color: Yellow]
    ├── Child: Running shoes - Yellow - 42 (SKU: 000-RS-Y-42) [Size: 42]
    ├── Child: Running shoes - Yellow - 43 (SKU: 000-RS-Y-43) [Size: 43]
    └── Child: Running shoes - Yellow - 44 (SKU: 000-RS-Y-44) [Size: 44]
```

In Magento, this becomes:

```
Magento Structure:
__________________
```

```
Configurable: Running shoes (SKU: 000-RS)
├── Simple: Running shoes - Blue - 42 (Color: Blue, Size: 42)
├── Simple: Running shoes - Blue - 43 (Color: Blue, Size: 43)
├── Simple: Running shoes - Blue - 44 (Color: Blue, Size: 44)
├── Simple: Running shoes - Red - 42 (Color: Red, Size: 42)
├── Simple: Running shoes - Red - 43 (Color: Red, Size: 43)
├── Simple: Running shoes - Red - 44 (Color: Red, Size: 44)
├── Simple: Running shoes - Yellow - 42 (Color: Yellow, Size: 42)
├── Simple: Running shoes - Yellow - 43 (Color: Yellow, Size: 43)
└── Simple: Running shoes - Yellow - 44 (Color: Yellow, Size: 44)
```

The GrandParent becomes the configurable product, and all Children become its simple product variants. **Parent products are skipped** - they only exist to organize the hierarchy and pass Color values to Children.

Configurable Product Level

By default, the GrandParent becomes the Magento configurable product. Starting with v3.9.0, you can change which hierarchy level becomes configurable.

Stores → Configuration → Katana PIM → Product Data → General → Configurable Product Level

Setting	GrandParent	Parent	Children	Use case
GrandParent (default)	Configurable product	Skipped (data only)	Simple products linked to GrandParent	Standard setup: one configurable per product family
Parent	Skipped	Configurable product	Simple products linked to Parent	Separate configurable per color/variant
Both	Configurable product	Configurable product	Simple products linked to both	Two levels of configurables

GrandParent Mode (Default)

```
Magento:
Configurable: Running shoes (SKU: 000-RS)  ← GrandParent
├── Simple: Running shoes - Blue - 42       ← Child (Color: Blue, Size: 42)
├── Simple: Running shoes - Red - 43        ← Child (Color: Red, Size: 43)
└── ...
```

One configurable with all children. Both Color and Size are configurable attributes.

Parent Mode

```
Magento:
Configurable: Running shoes - Blue (SKU: 000-RS-B)  ← Parent
├── Simple: Running shoes - Blue - 42                ← Child (Size: 42)
├── Simple: Running shoes - Blue - 43                ← Child (Size: 43)
└── Simple: Running shoes - Blue - 44                ← Child (Size: 44)

Configurable: Running shoes - Red (SKU: 000-RS-R)   ← Parent
├── Simple: Running shoes - Red - 42                 ← Child (Size: 42)
└── ...
```

Each Parent becomes its own configurable. Only Size is a configurable attribute — Color is fixed per Parent. GrandParent products are skipped.

Both Mode

Both GrandParent and Parent become configurable products. Children are linked to both levels. Use this when you need both a master configurable and per-variant configurables.

Smart Attribute Filtering

The module automatically skips configurable attributes that have only one distinct value among the children. For example, in Parent mode, "Running shoes - Yellow" won't get Color as a configurable attribute because all its children share the same color.

Switching Modes

After changing the configurable level:

1. Run **Reload Data from API** to re-import product data
2. Run **Sync Full** to recreate Magento products
3. Flush Magento cache

The setting has `canRestore` enabled — use the "Use Default" button to reset to GrandParent mode.

Prerequisites in KatanaPIM

Before importing, ensure your KatanaPIM products are structured correctly:

1. Product Structure

- **GrandParent products** must exist as the top-level product (the "master" product)
- **Parent products** must have a `ParentId` pointing to the GrandParent
- **Child products** must have:
  - A `ParentId` pointing to their Parent
  - A `GrandParentId` pointing to the GrandParent

2. Variation Attributes

Each level holds specific variation attribute values:

- **GrandParent**: No variation attributes (this is the base product)
- **Parent**: First variation attribute value (e.g., Color = "Red")
- **Child**: Second variation attribute value (e.g., Size = "43")

Attribute inheritance: Children inherit the Color value from their Parent. So a Child under "Running shoes - Red" automatically gets Color = "Red", and you only need to set Size = "43" on the Child itself.

This means when the Child is created in Magento, it will have both attributes:

- Color: Red (inherited from Parent)
- Size: 43 (set on Child)

3. Verify in KatanaPIM

Check your product structure in KatanaPIM:

1. Open a Child product
2. Verify it has both `ParentId` and `GrandParentId` fields populated
3. Check that variation attributes (Color, Size) have values

Step 1: Import Attributes from KatanaPIM

First, import all attributes including your variation attributes.

1. Navigate to **Katana PIM → Attributes**
2. Click **Import Attributes**
3. Wait for import to complete
4. Locate your variation attributes (e.g., "Color", "Size")

Step 2: Mark Attributes as Configurable

This is the crucial step. Attributes used for product variations MUST be marked as configurable.

1. In **Katana PIM → Attributes** grid
2. Find your first variation attribute (e.g., "Color")
3. Click the **Magento Attribute Code** cell
   - Select existing Magento attribute OR
   - Select "Create New" to auto-create
4. Click the **Is Configurable** cell
5. Set to **Yes**
6. Repeat for all variation attributes (e.g., "Size")

Example configuration:

Katana Attribute	Magento Attribute	Is Configurable
color	color	**Yes**
size	size	**Yes**
brand	brand	No
material	material	No

Important: If "Is Configurable" is not set to Yes, the module cannot create configurable product relationships.

Step 3: Create Magento Attributes

After mapping and setting "Is Configurable = Yes", create the attributes in Magento:

1. Select the mapped variation attributes (checkboxes)
2. Use mass action **Update/Create Magento Attribute**

The module automatically:

- Creates attributes with `katanapim_` prefix if they're new
- Sets "Use in Configurable Products" to Yes for attributes marked as configurable
- Sets the correct input type (Dropdown)

Step 4: Import Products

Now import the products from KatanaPIM.

1. Navigate to **Katana PIM → Products**

2. Click **Sync Full** to import all products
3. Wait for import to complete

After import, you'll see products with different "Katana Type" values:

- **GrandParent** - Your master configurable products
- **Parent** - Intermediate level (color variants in our example) - **appears in grid but no Magento product is created**
- **Child** - The actual simple products with all variations

What you'll see in the grid:

Name	SKU	Katana Type	Type (Magento)
Running shoes	000-RS	GrandParent	Configurable Product
Running shoes - Blue	000-RS-B	Parent	*(empty - no product)*
Running shoes - Red	000-RS-R	Parent	*(empty - no product)*
Running shoes - Blue - 42	000-RS-B-42	Child	Simple Product
Running shoes - Red - 43	000-RS-R-43	Child	Simple Product

Notice that Parent rows have an empty "Type" column - this is expected. The module uses Parents only for data inheritance (the Color attribute), not for creating products.

Step 5: Create Magento Products

Create products in the correct order:

Automatic Creation

If **Auto create new products** is enabled in Automation settings:

- Products are created automatically during sync
- The module handles the correct creation order
- Parent products are automatically skipped (no Magento product needed)

Manual Creation

If creating manually:

1. **First**: Select all **Child** products → **Update/Create Magento Products**
2. **Second**: Select all **GrandParent** products → **Update/Create Magento Products**

3. **Skip**: Parent products - do not create them (they're data containers only)

Why this order? The configurable product (GrandParent) needs its simple products (Children) to exist first so it can link them as variants.

Tip: You can filter by "Katana Type" to easily select all Children or all GrandParents.

Verifying the Setup

After creation, verify in Magento:

Check Configurable Product

1. Go to **Catalog → Products**
2. Find your GrandParent product (e.g., "Running Shoe")
3. Verify **Type** is "Configurable"
4. Edit the product
5. Go to **Configurations** section
6. Verify all child products are linked
7. Verify variation attributes (Color, Size) are shown

Check on Frontend

1. Navigate to the product page
2. Verify both variation dropdowns appear (e.g., Color and Size)
3. Test selecting different combinations
4. Verify correct SKU/price updates per selection

Common Configurations

Two Variations: Color + Size (Apparel/Footwear)

Attribute	Type	Is Configurable
color	Dropdown/Swatch	Yes
size	Dropdown/Swatch	Yes

Two Variations: Material + Length (Hardware)

Attribute	Type	Is Configurable
material	Dropdown	Yes
length	Dropdown	Yes

Two Variations: Flavor + Package Size (Food)

Attribute	Type	Is Configurable
flavor	Dropdown	Yes
package_size	Dropdown	Yes

Troubleshooting

Configurable product not linking children

Symptoms: GrandParent creates as configurable but has no variants

Solutions:

1. Verify Children were created first (check Magento Product ID in grid)
2. Check "Is Configurable" is Yes for variation attributes in the Attributes grid
3. Re-run sync: Select GrandParent → **Update/Create Magento Products (force)**

If still not working:

1. **Recreate the Magento attributes:**
   - Go to **Katana PIM → Attributes**
   - Select the variation attributes
   - Use mass action **Update/Create Magento Attribute**
   - This ensures attributes are created with correct settings
2. **Check debug logs for errors:**
   - Enable debug mode: **Stores → Configuration → Katana PIM → General → Debug Mode** → Yes
   - Re-run the sync
   - Check logs at `var/log/katanapim/debug.log` and `var/log/katanapim/error.log`
   - Look for attribute or configurable product related errors

3. **Verify Magento attribute settings manually:**
    - Go to **Stores → Attributes → Product**
    - Find your variation attributes (e.g., `color`, `size`)
    - Confirm **Catalog Input Type for Store Owner** is "Dropdown" or "Visual Swatch"
    - Confirm **Use in Configurable Products** is set to "Yes"
    - If you had to change these, re-sync the GrandParent product

Products created as Simple instead of Configurable

Symptoms: GrandParent product is Simple, not Configurable

Solutions:

1. Check product has correct Katana Type = "GrandParent" in grid
2. Verify GrandParentId is set on Child products in KatanaPIM
3. Delete the incorrectly created Magento product
4. Fix the KatanaPIM structure
5. Re-sync and recreate

Missing variation attribute in dropdown

Symptoms: Only one variation shows on frontend (e.g., Color but not Size)

Solutions:

1. Verify both attributes are marked "Is Configurable = Yes"
2. Check Magento attribute settings (must be Dropdown type)
3. Ensure Child products have values for both attributes
4. Re-sync the GrandParent product

Children showing wrong GrandParent relationship

Symptoms: Products linked to wrong configurable parent

Solutions:

1. Check `GrandParent ID` column in Products grid (enable if hidden)
2. Verify KatanaPIM data has correct GrandParentId values
3. Delete and re-import affected products

Products Grid Columns

Enable these hidden columns for debugging configurable products:

Column	What it shows
Parent ID	KatanaPIM Parent ID (first level parent)
GrandParent ID	KatanaPIM GrandParent ID (top level)
Katana Type	Simple, Child, Parent, or GrandParent
Type	Resulting Magento product type

To enable: Click **Columns** button in grid toolbar → check the columns you need.

Need More Help?

Documentation:

- All Help Articles - Complete documentation overview

Support:

- Contact Support - Get help from our team

For a complete overview of features and configuration options, see the KatanaPIM extension on magmodules.eu

All articles for KatanaPIM

Installation

1	Installation using Composer (recommended)
2	Install through FTP and SSH

Configuration

1	Quick Start Guide
2	Configuration
3	Categories Grid
4	Attribute Mapping
5	**Configurable Products Setup (current)**
6	Automation

Troubleshooting

1	Troubleshooting

Grids

1	Products Grid
2	Attributes Grid
3	Assets Grid
4	Sync Log Grid

Background

1	CLI Commands

2	Cron Jobs
3	Product to Website Matching
4	Understanding the Sync Process

