

Weaverシリーズにおけるデータ構造と自動化の仕組み：CSVとテンプレートの関係性解説

1. はじめに：Weaverシリーズのデータ連携コンセプト

Weaverシリーズ（LayerWeaver, MotionWeaver, RigWeaver）は、「3Dの整合性」を「2Dならではの自由な表現」へと繋ぐためのパイプラインです。

この広範なシステムにおいて、異なるアプリケーション間（Krita, MotionWeaver, Moho）で一貫性を保ち、処理を自動化するための「ハブ」として機能するのがCSVおよびJSONファイルです。設計思想として、制作者はバックエンドの技術的な詳細を意識する必要はありませんが、これらのファイルが「何を決めているのか」を理解することで、制作工程をより効率的にコントロールできるようになります。

2. 3つの基幹ファイルの役割と定義

Weaverシステムの屋台骨を支える3つのデータファイルを、その詳細な情報とユーザーのアクセス権限に焦点を当てて整理します。

ファイル名	役割	主な格納情報	ユーザーアクセス
islands.csv	パーツの幾何学カタログ	レイヤー解析（Pixel Analysis）による各パーツ（島）の名前、中心座標、関節位置（Holes）、境界領域。	自動生成 (要確認)
rigging_map.csv	ボーンの配線図	ボーンの親子関係（Hierarchy）。リグがどのように連動するかを定義するスケルトン設計図。	自動生成 (タグで制御)
templates.json	マスター・カタログ	骨格の定義、バインドのグローバル・ルール、計算順序などのシステム全体の動作規約。	システム管理 (技術者が調整)

3. templates.json：システム共通の「マスター・カタログ」

templates.json は、Weaverシリーズ全体の動作ルール（Global Rules）を司る共通基盤です。このファイルには、「3Dから2Dへの変換ルール」が定義されています。最新の仕様では、各ツール（RigWeaver, MotionWeaver, LayerWeaver）ごとに最適化された固有の設定セクションに分離されており、より精密な制御が可能となっています。

- ツール別の固有設定セクション:
 - RigWeaver固有: Moho上でのリギング挙動、仮想ボーン定義、およびレイヤー倍率からパースを逆算するための基準値（rig_perspective_base_z）などを定義。
 - MotionWeaver固有: 3Dモーション解析ルール、およびパース強度のデフォルト値（motion_perspective_default_z）などを定義。

3. **LayerWeaver固有**: Krita上でのレイヤー解析アルゴリズムやパーツ抽出ルールを定義。

- **骨格概念とキネマティック制約**: 人間型（Humanoid）や四足歩行（Quadruped）といった骨格ごとの基本構造、およびボーン名の標準規格を保持します。
- **「変換の法則（Law of Conversion）」の隠蔽**: 3DソフトとMohoでは座標軸の設計が根本から異なります（X軸とZ軸の反転など）。この変換計算はバックエンドで自動処理されており、ユーザーは空間座標の不一致を意識せず「見たまま」を制御できます。
- **自動バインド・ルールの定義**:
 1. **フレキシ・バインド（Flexi-bind）**: 胴体（Torso）や首（Neck）など、滑らかな変形が必要な部位へのルール。
 2. **レイヤー・バインド（Layer-bind）**: 手足（Limbs）や頭部（Head）など、形状をリジッドに保つべき部位への固定ルール。
- **3段階のパス解決（Path Resolution）**: システムは `templates.json` を以下の優先順位で探索します。これにより、プロジェクトごとのカスタマイズとシステムの汎用性を両立させます。
 1. PSDファイルと同階層
 2. プラグイン・フォルダ内
 3. `templates_path.txt` に記述されたカスタムパス

4. 制作者のワークフロー：レイヤー名による設定タグ

Weaverシリーズの特徴は、制作者が使い慣れたキャンバス上での「命名」が、そのままリギングの設定情報として機能する点にあります。

1. **タグ付け（Visual Scripting）**: Kritaでの作画時、レイヤー名に記号を付与します。
 - `@`: グループ化やアクセサリ（例：`@Hat`）の定義。
 - `>`: 明示的な親子関係（例：`Arm > Hand`）の記述。
2. **リギングの自動構築**: LayerWeaverがこれらのタグを読み取り、`islands.csv` と `rigging_map.csv` を自動生成します。
3. **命名＝設計**: 制作者にとっての「レイヤー整理」という日常行為が、そのまま `rigging_map.csv` の配線図を更新する行為になります。
4. **作画への没入**: ユーザーはデータの裏側を記述する手間から解放され、キャンバス上での「構成」に注力できます。

5. 役割分担：表現の領域とレスキュー・モード

Weaverシリーズは、制作者が「2D表現」に集中できるよう、技術的な役割分担を明確にしています。

- **表現の領域（2D Expression Field）**
 - `_3d.csv` は、Mohoでキャラクターを動かすための「最終出力データ」です。3Dの情報を元に生成された後は、3Dの数学的制約に縛られず、2Dとしての調整が可能です。
 - 3D側でメッシュが破綻していても、MohoのZ軸（奥行き）やベクター変形を用いて「デフォルメ表現（パースの誇張、スマイル等）」を上書きすることが推奨されます。
- **技術者の領域（System Tuning）**
 - **Mode 1（Standard）**: 描画したポーズ（Aポーズ等）を基準に骨格を組む基本モード。
 - **Mode 2（Force T-pose）**: イラストのポーズが極端すぎてメッシュが崩壊した場合の「緊急レスキュー用」。強制的に第0フレームにTポーズを挿入し、計算を安定させます。

- **パラメータの外部化**: 特殊な多脚ロボットの定義や物理挙動の調整が必要な場合、技術者が `templates.json` を数秒調整するだけで、パイプライン全体をアップデート可能です。
-

6. まとめ：演出の「指揮」への専念

CSVとテンプレートの連携は、単なる作業の短縮ではありません。それは、制作者を「リギングの設定」という技術作業から解放し、演出のコントロールに専念させるための仕組みです。

特に、**BatchWeaver** の活用はこの自動化を量産のフェーズへと押し上げます。RigWeaverで発行されたCSVは、BatchWeaverにとっての「発注書（Order Form）」となります。この発注書を元に、ひとつのBVHから「正面」「俯瞰」「煽り」といった多数のアングル違いのアニメーションを一括量産することが可能です。

「3Dの整合性」を土台にしつつ、その上で「2Dならではの表現」を追求する。 Weaverシリーズが提供する自動化の仕組みは、あなたの表現をサポートするための強力な道具となるでしょう。