

マンガキャッチ 仕様書 v1.3 (追記: 3リング／安定化／データ確定)

A. 追跡・参加ルール (確定)

A1. 複数人時の操作対象 (確定: 方式(c))

- 同時プレイ最大3人 (リング最大3)
- 参加順ロック: 最初に検出された3人を Ring1~3 に割当て、プレイ中は原則固定
- 4人目以降は 空きリングが出るまで無視

退出・再参加 (確定値)

- 退出判定: 未検出 1.5秒でそのリング枠を解放
- 再参加: 空き枠があれば、次に検出された人を参加順で割当

これにより「最も近い」の曖昧さを排除し、展示で最も事故が少ない挙動になります。

A2. TouchDesigner → ゲーム 最終データ (確定)

- (x, y, id) 正規化・最大3点で固定
- “id” は リングID (1~3) とする (参加順ロックの結果をそのまま送る)

これによりゲーム側は「点群の追跡」から解放され、安定化の責任は TouchDesigner側に閉じます (非エンジニア運用として最も安全)。

A3. 座標系の定義 (ここで決め打ち: 確定)

ゲームが受け取る座標は 正規化 (0~1) とし、以下で固定します。

- 原点: 画面左上
- x: 左→右に増加 (0.0~1.0)
- y: 上→下に増加 (0.0~1.0)
- 範囲外はクランプ (0~1)
- リングの画面座標変換:
 - $screen_x = x * 1920$
 - $screen_y = y * 1080$

※リングの操作は当面 **x**中心 (**y**は将来拡張用・ログ用に保持)として実装して構いません。

通信仕様(TouchDesigner → ゲーム)確定

1. Transport

- **OSC over UDP**
- Host: **127.0.0.1**
- Port: **7000**

2. OSCアドレス(確定:衝突回避)

- TouchDesigner → ゲーム: **/mangacatch/players**
- (参考)HokuyoUtilの生出力は別系統で扱う(ポートも分ける)
→ 生出力とゲーム入力の衝突を防ぐため、アドレスを分離します。

3. Payload(固定長:確定)

Float配列(固定10要素)

[frame, x1, y1, id1, x2, y2, id2, x3, y3, id3]

- **frame**:経過フレーム(任意／ログ用途)
- **id1..3**:**1,2,3**(リングID)
- 未参加枠は:**x=-1, y=-1, id=0**

B. ランキング・同点処理(確定)

B4. 並び順(確定)

ランキングの比較キーは以下で固定します。

1. 合計スコア(**total_score**)降順
2. 同点なら 取得レア合計(**rarity_sum**)降順
3. それも同点なら 達成時刻(**achieved_at**)昇順(早い方が上)

※参加人数は記録しない(不要)。

B6. 点数・出現率(ここで決め打ち:確定)

「難易度は1種類でOK」の方針を守りつつ、ランキングの“レア合計”が意味を持つよう、レア度を3段階で固定します。

(後からJSON差し替えで調整可能ですが、仮開発はこの値で進められます)

レア度定義(確定)

- Common: `rarity_tier=1 / rarity_point=1`
- Rare: `rarity_tier=2 / rarity_point=3`
- SuperRare: `rarity_tier=3 / rarity_point=6`

Typeごとの初期割当(確定:仮固定)

- Type01~05: Common (weight=10 / score=100)
- Type06~08: Rare (weight=6 / score=150)
- Type09~10: SuperRare (weight=3 / score=250)

取得レア合計(`rarity_sum`)の計算(確定)

- GETした画像ごとに `rarity_point` を加算
- ランキングの2ndキーに使用
- リザルトでは 個別内訳として「Type別取得数」「Type別得点」「Type別`rarity_point`合計」を表示(あなたの指定Bに対応)

B7. 推薦文言(確定:仮扱い)

- 推薦ページの文章は すべて“(仮)”と明記し、後で差し替え前提で実装する。

B8. アセット仕様(ここで決め打ち:確定)

1) キャラクター切り抜き

- 形式:PNG

- 透過: 必須 (RGBA)
- 推奨カラープロファイル: sRGB
- 推奨サイズ: 1000×1000
- 許容: 原本サイズは任意 (大きくても可)
 - ゲーム側で 最大辺1024px相当に縮小してロード (または事前変換)
 - ※現状の素材が可変サイズのため、ここで吸収するのが現実的です

2) 書影 (作品サムネ)

- 形式: PNG
- 透過: 不要 (RGBで可)
- 縦横比: 自由 (現物が可変のため)
- 表示: UI枠に **contain** (縦横比維持で内接) で収める

3) 命名規約 (最終形: 確定)

最終的に運用しやすい命名を仕様として固定します。

- キャラ: `assets/characters/type01.png ... type10.png`
- 書影: `assets/books/type01.png ... type10.png`

ただし現状はファイル名が揃っていないため、仮開発では

- **manifest** (ID→ファイル名対応表) で吸収
- 後日、最終命名にリネームしても manifest 差し替えだけで対応とします。

今この仕様で「仮開発」に進めるか

進めます。理由は以下です。

- 3リング／途中参加／安定化の核 (参加順ロック) が確定
- TouchDesigner→ゲームのIF (座標系・フォーマット・アドレス・固定長) が確定
→ 実装が止まる要因が消えています
- 点数・レア・同点処理が確定
→ ランキング／リザルト (個別内訳) の実装が可能です
- アセットは可変サイズでも仕様で吸収し、欠品はプレースホルダーで進行可能

まだ決まっていないこと (残タスク)

開発は進められますが、最終納品までに必要な未決定は次です。

1. 不足している「あと1名」分の確定データ(作家／作品／キャラ名の最終)
2. 不足している画像2種類の本番データ(切り抜き＋書影)
3. タイトル画面のクレジット表記文言(最終稿)
4. 推薦文言の“仮”→本番差し替え(タイミングは後でOK)

必要であれば、次のステップとして、TouchDesigner側で「参加順ロックして3枠を </mangacatch/players> に送る」ための**具体的なネットワーク構成(どのノードで何をするか)**を、非エンジニア向けに手順化して提示します。これが揃うと、実装側はAntigravityに投げて一気に組み上げやすくなります。