魚図鑑アプリ仕様書

1. 概要

- 魚のグループ、撮影場所、魚種、動画を一覧・絞り込み・再生できる Web アプリ
- クライアント側での高速なフィルタリング（グループ、場所、カナ）と動画連続再生機能を両立

2. 現状と課題

- GAS の doGet に逐次エンドポイントを追加しており、その都度動作変更に伴うバグやデグレが発生
- CORS 問題によりブラウザ環境での fetch が失敗し、ローカルテストが困難
- 再生機能は動作しているが、フィルタとアイコン切り替えの制御ルールが一貫せず、UX が不安定

3. 機能要件

1. **初期データ取得**
2. **魚一覧取得**: `listFishes` で全魚 (約400件) を一度だけ取得し、初期状態でユーザーが全魚から選択できるようにする。
3. **初期グループ動画取得**: クライアント側で乱数 (0–56) によって最初の GroupID を決定し、`listVideos?groupId=<ID>` で当該 Group の動画情報のみを取得。
4. **場所リスト**: 増減しない前提でクライアント内に静的配列として定義。
5. **グループID範囲**: 0–56 を静的配列または乱数範囲としてクライアント内で管理。
6. **フィルタ動作と優先順位**
7. **魚選択 (最優先)**
    - fishSelect が行われた場合、他のフィルタ（仲間・場所）は無視し、指定魚のみを再生して停止。
8. **魚未選択の場合**
    - **仲間 & 場所 共未選択**
    - ランダムに選ばれたGroupの動画を再生。連続再生ON時は終了後に再度ランダムGroupを再生。
    - **仲間のみ選択**
    - 指定Group内の全魚の動画を最初から順に再生（場所フィルタ不適用）。
    - **場所のみ選択**
    - 再生中Groupは維持し、Group内の魚アイコンを表示。選択場所で撮影された魚のみカラー表示、その他は白黒表示。
    - **仲間と場所 同時選択 (AND処理)**
    - 指定Group内かつ選択場所で撮影された魚のみを対象に、その魚の動画を最初から順に再生。
9. **UI フィルタ**
10. ドロップダウン (`#groupSelect` , `#placeSelect`) とカナボタンによる絞り込み
11. 初回データ取得後はクライアント内データで高速絞り込み
12. **動画再生機能**

13. YouTube IFrame API (`onYouTubeIframeAPIReady`, `onPlayerStateChange`)
14. 初回は「▶ 再生」ボタン押下で開始

15. 連続再生ON/OFF 切替

16. **パフォーマンス/可用性**

17. 初期ロード時のフェッチは魚一覧 (約400件) + 初期Group動画 (数件) のみ
18. CORS 対策済みの `script.googleusercontent.com` URL を利用
19. 以降はクライアント内データのみで完結

4. API 仕様. API 仕様

action	パラメータ	戻り値
listGroups	なし	Array of
listPlaces	なし	Array of place strings
listFishes	groupId?, kana?	Array of
listVideos	なし	Array of
getRandomGroup	なし	Object for random Group data

5. クライアント構成

- データ取得: fetch() + JSON（script.googleusercontent.com URL）
- フィルタ処理: groupSelect, kanaButtons, placeSelect のイベントでローカルデータを絞り込み
- 動画制御: onYouTubeIframeAPIReady, onPlayerStateChange

6. テストとデプロイ

- ローカルテスト: Live Server の HTTPS モード or CORS フリー URL で実行
- 本番環境: script.googleusercontent.com ドメインの Web アプリ URL を利用
- GAS デプロイ: バージョン管理し最新エンドポイントを反映

上記仕様をご確認の上、追加・修正があればお知らせください。

7. 起動から再生ボタン押下までのプログラムフロー

1. **ページロード**

2. ブラウザが `index.html` を読み込み、HTML/CSS/JavaScript がロードされる

3. **初期化処理 (`DOMContentLoaded` 内)**

4. **魚一覧取得 (`initFishes()`)**
   - クライアントから `fetch(BASE_API_URL + '?action=listFishes')` を送信
   - GAS の `doGet(e)` で `action=listFishes` を受け、`listFishes()` が `Fishes` シートから全魚（約400件）を読み取り JSON で返却
   - クライアントは応答を `allFishes` に格納し、`#fishSelect` ドロップダウンおよびアイコンリストを描画

5. **初期グループ決定**
    - `initialGroup = Math.floor(Math.random() * 57)` によって 0–56 の乱数を生成
6. **初期動画取得 (`initFirstGroup()`)**
    - `fetch(BASE_API_URL + '?action=listVideos&groupId=' + initialGroup)` を実行
    - GAS の `doGet(e)` で `action=listVideos` と `groupId` を受け、`listVideos(groupId)` が指定グループの動画情報数件を `Videos` シートから取得し JSON で返却
    - クライアントは応答を `currentVideos` に格納し、魚ヘッダーの動画総数表示 (`#videoCount`) などに反映
7. **YouTube IFrame API 読み込み & プレイヤー生成**
8. `<script src="https://www.youtube.com/iframe_api"></script>` が実行され、API ロード完了後に `onYouTubeIframeAPIReady()` が呼ばれる
9. `onYouTubeIframeAPIReady()` 内で `YT.Player` を生成し、`playerVars` に `{ rel:0, origin: window.location.origin }` を設定
10. この時点では `videoId` を指定せずロードせず、再生ボタン押下を待機
11. **再生ボタン押下**
12. ユーザーが「▶ 再生」ボタンをクリックすると、イベントハンドラが発火
13. `currentVideoIndex = 0` をセットし、`ytPlayer.loadVideoById(currentVideos[0].url || currentVideos[0].VideoID)` を呼び出し
14. プレイヤーが最初の動画を読み込み再生を開始
15. **以降の再生制御**
16. 再生中／再生終了後のロジックは `onPlayerStateChange` で管理し、連続再生やグループ切替などの動作を制御
