

HDIC Viewer 2023: 古辞書総合情報ポータルサイトの構築 に向かって

劉 冠偉 (東京大学史料編纂所)
2023年1月21日
シンポジウム「古辞書データ共有と拡張」

HDIC Viewerとは

- 最初は平安時代漢字字書総合データベース(HDIC)を公開・検索するためのインターフェースとして開発
- HDIC以外の古辞書データベースを追加
 - 夢梅本和玉篇
 - 本草和名
 - 大字典和訓
 - 一切経音義



HDIC Viewerの現状

- ページ数
 - 48
- API数
 - 40
- テーブル数
 - 20
- レコード数
 - 24万(古辞書関連データ)

今までの開発

- モデリング
 - ほとんどは表データのまま
- 設計
 - テキストDB
 - 検索のみ
 - 管理はRDBMSに直接操作
 - 画像DB
 - ローカル画像ファイル
- 実装
 - スタック(Stack)
 - 開発に利用するプログラミング言語、フレームワーク、ライブラリなど
 - PHP、MySQL、Laravel、Vue
 - HNG単字検索も同様に開発

課題・問題点

- 長年にわたって
 - 2015～現在
- 問題:
 - モデリング
 - DBそれぞれに異なる
 - 設計
 - テキストデータ検索のみ
 - 管理インタフェースがない
 - 実装
 - 開発パターンの不統一
 - テンプレート(バックエンドのみ)
 - フロントエンド・バックエンド分離
 - PHP
 - セキュリティー
- これ以上の拡張は難しい

スタック変更の試し

- “辞書語彙データベース”の開発
 - 藤本灯科研 ([21H00529](#))
 - URL: <https://kojisho.netlify.app/>
 - 管理システム: <https://portal.kojisho.com/admin/login/?next=/admin/>
 - Django (Python)
 - Quasar (Vue3, JavaScript)
- 良いところ
 - 参考事例が多い
 - 管理サイトが内蔵
- 悪いところ
 - 重い(どっちも)
 - 開発言語が不統一
 - Quasarが不人気

辞書語彙データベース

- フロントエンド・バックエンドの分離
 - 当然、WEB APIも提供
- 管理サイトによってデータの一括更新
 - エクセルファイルなど

開発上の不足

- モデリング
 - Django内蔵ORM
 - カスタマイズしにくい
- 設計
 - フロント側は検索のみ
- 実装
 - Python...
 - デメリットはJavaScriptではない
 - もちろん速度、タイプチェック、インデントなども嫌いけど

ポータル网站的な存在は？

- HDIC Viewer、HNG単字検索、辞書語彙データベースの開発から分かったこと
 - (研究資源として)元データの形式は様々
 - 画像ファイル、ページURL、APIなどなど
 - 完璧なモデルは(僕が作れ)ない
 - 使いやすい、作りやすい、残りやすい
 - どちらの二つしかできない
 - 最初の設計によって拡張性が決まる
 - 作りながら考えよう...をやめましょう → HDIC Viewer
 - まずは日本古辞書関連データを集めてほしい
 - ポータルサイト

なんの情報の集合？

- 漢字情報ポータルサイトではない
 - 形・音・義以外
- 古辞書の情報のポータルサイト
 - 複数の古辞書
 - さらに複数の出典
 - 画像とテキスト
 - 広い年代・形式
- ↑すぐには無理
 - データマイニング ← モデリング ← 理論

設計(機能)

- テキストデータ管理
 - スキーマ
 - DBのCRUD
 - インポート・エクスポート
- 画像データ管理
 - 画像ファイル名と項目ID
 - 外部IIIFのURL
- 字形データ管理
 - GlyphWikiダンプデータの導入
 - 字形データ(Kage)の管理
- ユーザー管理
 - ユーザーデータ
 - 通知システム

設計(モジュール)

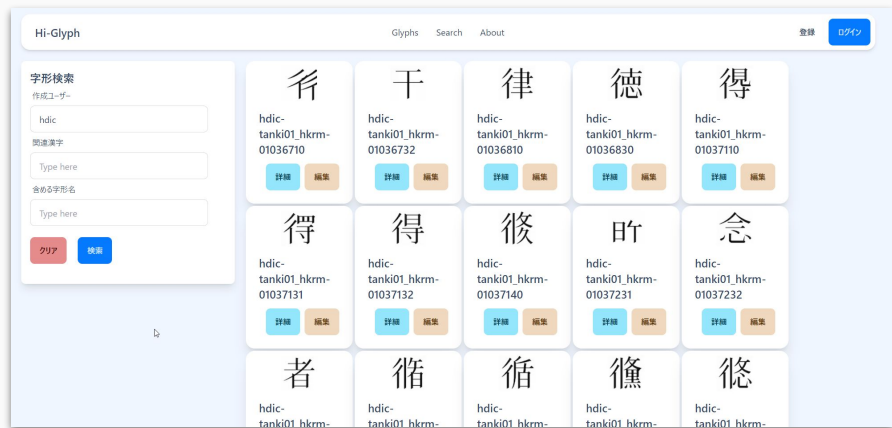
- データの管理
 - インポート・エクスポート
 - 自動化: GitHub、Dropbox; マルチ形式: プレインテキスト(TSV、CSV、JSON)、Excel
 - バージョニング; ソフトドロップ
- 認証
 - JWT; ロール管理
- WEB API
 - Restful
- データモデル
 - マスター
 - 各自のテーブル
 - 構造化データの管理
- 画像管理
 - IIIFクライアント(外部)
 - JPEG・PNG画像ファイル(内部)
- 字体・字形管理
 - 字形データ: Kage形式

【これから】開発用のスタック

- 開発言語
 - JavaScript (TypeScript)
- データ
 - ORM: Prisma
- フレームワーク
 - バックエンド: NestJS
 - フロントエンド: NextJS (候補: Nuxt、最近React系が好きになった)
- デプロイメント
 - NodeJS環境
 - PM2で管理

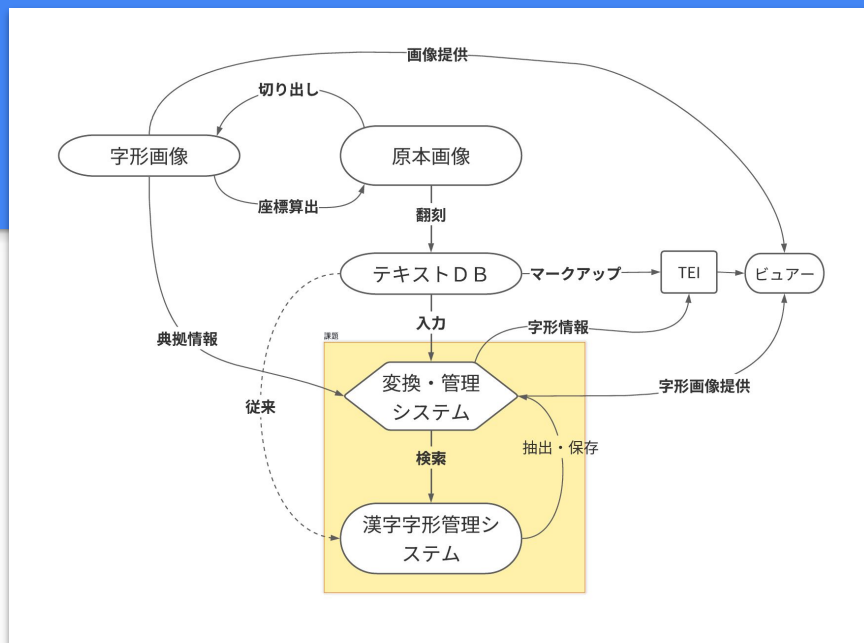
実装の一例：字形データ管理（モジュール）

- 独立の字形管理システム
- 機能
 - GlyphWikiダンプデータの導入
 - 字形の作成・編集
 - GlyphWikiと同じエディタを利用
 - 字形画像生成
 - Kage → SVG;PNG
 - 字形の検索
 - Kageデータのパーサー
 -



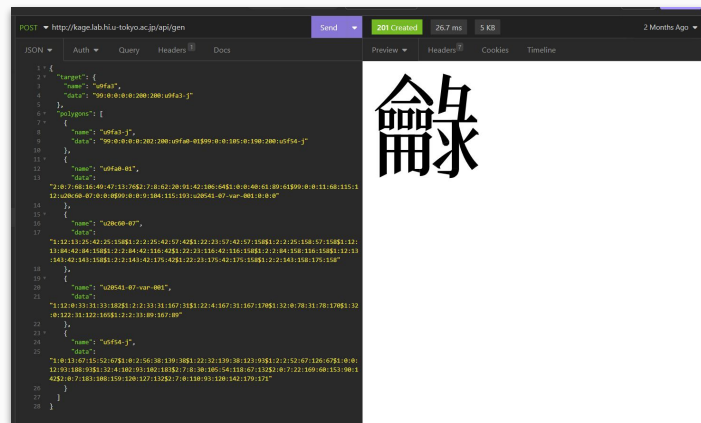
字形管理の必要

- 独自のサービス
 - GlyphWikiに依頼せず
- 著作権の明白
 - CC-BY予定
- 実験台
 - 字体字形サービスのテスト
 - GlyphWikiを汚染しない
 - 「hdic」で検索したらやばい量が...



Kage Engine Online

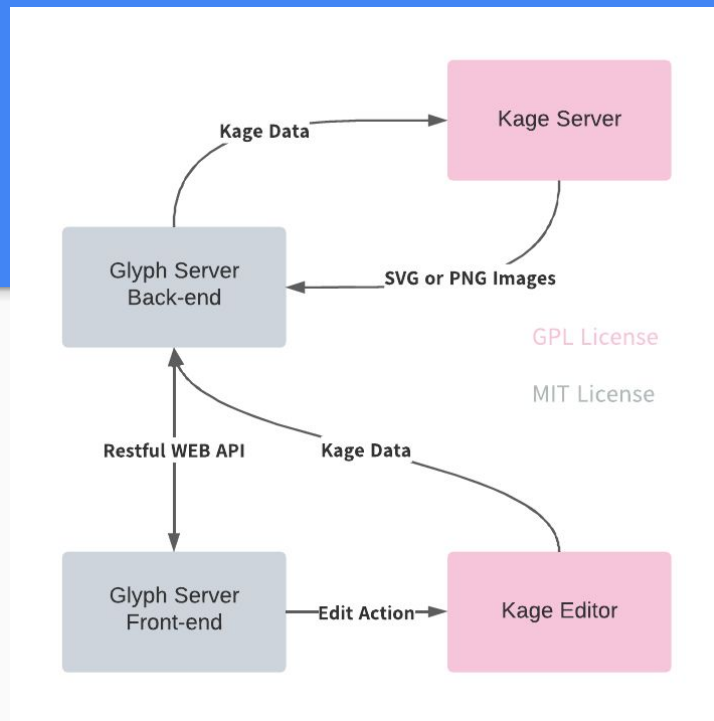
- Kage-engineのライセンス
 - GPL
 - 感染性あり
 - 関係ある部分を独立なサービスとして開発
 - その部分のコードだけはGPLライセンス
- Kage-Server 開発版
 - ソース: <https://github.com/toyjack/kage-server>
 - URL: <http://kage.lab.hi.u-tokyo.ac.jp/api/gen>
 - 伝説の「KAGE/cgi サーバ仕様」を再現したい
 - <http://kamichi.jp/spec.html>
 - ずっと停止...
 - 結局簡易版だけを作った



The screenshot shows a REST client interface with a POST request to `http://kage.lab.hi.u-tokyo.ac.jp/api/gen`. The response is a JSON array of objects, each containing 'name' and 'data' fields. The first object has 'name': 'defax' and 'data': '99-0-0-0-200-200-uf63-j'. The second object has 'name': 'uf63-j' and 'data': '99-0-0-0-202-200-uf6b-01599-0-0-105-0-100-200-uf54-j'. The third object has 'name': 'uf6ab-01' and 'data': '92-0-7-108-30-0-0-13-17051-7-0-02-20-01-02-100-040-0-0-0-0-01-00-01000-0-11-00-155-1-12-000-00-07-0-0-0090-0-0-10-104-115-101-00950-07-var-001-0-0-0'. The fourth object has 'name': '020c00-07' and 'data': '78-123-25-42-25-15081-2-2-25-02-57-4281-22-23-57-02-57-15081-2-2-25-150-57-15081-121-13-100-42-04-15081-12-7-04-04-1300-408-22-13-130-42-1300-15081-7-2-04-150-130-15081-12-13-145-42-140-15081-12-2-140-0-3-70-0281-22-0-1-170-04-1-170-15081-2-2-140-130-170-15081'. The fifth object has 'name': '020541-07-var-001' and 'data': '4-132-0-13-13-18251-2-2-13-13-167-3151-12-4-167-31-167-17051-32-0-70-31-70-17051-32-0-10-122-30-122-16681-2-2-31-00-167-00'. The sixth object has 'name': 'uf54-j' and 'data': '78-0-0-1-071-0-2-55-30-30-3081-22-12-130-30-822-0561-2-2-152-07-130-0761-0-0-12-0-180-0951-32-4-180-0-180-1850-7-0-30-180-54-110-07-1324-0-7-22-160-00-153-00-1-4281-0-7-183-180-150-120-127-1320-7-0-130-0-130-152-170-175'. The right side of the browser shows a large Japanese character '録'.

実装

- ライセンスが異なるのため
 - 4つのサービスに分け
 - ピンクはGPL
 - グレーはMIT



Glyph Server

- URL
 - バックエンドサーバー
 - <https://glyph.lab.hi.u-tokyo.ac.jp/api/glyph/sandbox/png>
 - ソースコード
 - <https://github.com/toyjack/hi-glyph-server>
 - フロントエンド
 - 開発中(不安定過ぎ ...)
 - ソースコード
 - <https://github.com/toyjack/hi-glyph-front>
 - プロトタイプ
 - <https://hi-glyph-front.vercel.app/>
 - (よくフリーズしてしまう)

2023年度の展開

- 優先順位
 - ③データの管理
 - ⑤認証
 - ④WEB API
 - ②データモデル
 - ⑥画像管理
 - ①字体・字形管理

ご清聴ありがとうございました。