

OCODE.JS / VELCLAW

OCode.js

Framework web full-stack cho JavaScript và TypeScript

Kiến trúc rõ ràng. Quy trình kiểm thử được. Công cụ hướng tới nhà phát triển.

PRODUCT OVERVIEW · VIETNAMESE EDITION

Tài liệu định hướng sản phẩm. Các năng lực được mô tả là mục tiêu thiết kế, trừ khi repository xác nhận đã triển khai.

Xây dựng sản phẩm web, từ một nơi

OCode.js hướng tới việc gom các thành phần thường phải ghép nối vào một quy trình phát triển thống nhất.

NGUYÊN TẮC VÀ NỘI DUNG

- Tổ chức trang theo cấu trúc thư mục dễ khám phá
- Kết hợp giao diện người dùng với logic phía máy chủ
- Định nghĩa API gắn với phần mã sử dụng chúng
- Giữ cấu hình và quy trình build minh bạch

Triết lý thiết kế

Một framework tốt nên làm cho đường đi đúng trở nên dễ dàng, còn những quyết định quan trọng vẫn nằm trong tay nhà phát triển.

NGUYÊN TẮC VÀ NỘI DUNG

- Convention trước, cấu hình khi cần
- Mã nguồn có thể đọc, kiểm thử và mở rộng
- Ưu tiên chuẩn web và hệ sinh thái JavaScript
- Không tuyên bố tính năng đã sẵn sàng nếu chưa được xác minh trong mã nguồn

Định tuyến theo hệ thống tệp

Cấu trúc thư mục có thể phản ánh cấu trúc URL để nhóm phát triển tìm được trang, layout và endpoint nhanh hơn.

NGUYÊN TẮC VÀ NỘI DUNG

- Quy ước route được tài liệu hóa rõ ràng
- Hỗ trợ layout lồng nhau theo thiết kế của framework
- Quy định riêng cho route động và route nhóm
- Kiểm tra xung đột đường dẫn trong quá trình build

Giao diện và rendering

Kiến trúc hướng tới khả năng chọn chiến lược rendering theo yêu cầu sản phẩm, thay vì ép mọi trang vào cùng một mô hình.

NGUYÊN TẮC VÀ NỘI DUNG
• Server-side rendering cho nội dung cần phản hồi theo request • Static generation cho trang có thể tạo trước • Client-side interaction cho trải nghiệm tương tác • Caching và revalidation cần có quy tắc rõ ràng, có thể kiểm thử

API và xử lý phía máy chủ

OCode.js định hướng cung cấp một nơi có tổ chức để khai báo endpoint và kết nối dịch vụ bên ngoài.

NGUYÊN TẮC VÀ NỘI DUNG

- Quy ước request/response nhất quán
- Xử lý lỗi và mã trạng thái có cấu trúc
- Tách secrets khỏi mã nguồn và bundle phía client
- Kiểm tra xác thực, phân quyền và giới hạn tần suất trước khi phát hành

Dữ liệu và tích hợp

Ứng dụng có thể kết nối cơ sở dữ liệu, dịch vụ nội bộ hoặc API bên thứ ba thông qua lớp tích hợp tách biệt.

NGUYÊN TẮC VÀ NỘI DUNG

- Đặt thông tin kết nối trong biến môi trường
- Chuẩn hóa validation ở ranh giới dữ liệu
- Thiết kế timeout, retry và xử lý lỗi có chủ đích
- Không ghi token hoặc dữ liệu nhạy cảm vào log

CSS và hệ thống giao diện

Framework cần cho phép đội ngũ dùng công cụ tạo kiểu phù hợp với sản phẩm mà không khóa chặt vào một thư viện duy nhất.

NGUYÊN TẮC VÀ NỘI DUNG

- CSS thuần hoặc CSS Modules
- Tích hợp utility CSS khi dự án lựa chọn
- Token màu, khoảng cách và kiểu chữ có thể dùng lại
- Thiết kế responsive và accessibility ngay từ đầu

Tối ưu hiệu năng

Hiệu năng là kết quả của đo lường, không chỉ là khẩu hiệu. OCode.js nên cung cấp các mặc định an toàn và công cụ giúp nhận diện điểm nghẽn.

NGUYÊN TẮC VÀ NỘI DUNG

- Theo dõi kích thước bundle và tài nguyên tải
- Tối ưu ảnh và font theo chiến lược đã cấu hình
- Giảm JavaScript gửi tới trình duyệt khi có thể
- Đánh giá Core Web Vitals trên trang thực tế

Công cụ phát triển

Trải nghiệm phát triển cần phản hồi nhanh, thông báo lỗi hữu ích và quy trình lặp lại được trên máy cá nhân lẫn CI.

NGUYÊN TẮC VÀ NỘI DUNG

- Thông báo lỗi có ngữ cảnh và vị trí tệp
- Lệnh dev/build/test nhất quán
- TypeScript strict là mục tiêu khuyến nghị
- Tương thích runtime Node.js; hỗ trợ Bun chỉ khi được kiểm thử

Bảo mật theo mặc định

Bảo mật phải nằm trong quy trình thiết kế và phát hành, không phải bước bổ sung sau cùng.

NGUYÊN TẮC VÀ NỘI DUNG

- Không commit .env hoặc khóa API
- Validate input và encode output đúng ngữ cảnh
- Bảo vệ endpoint bằng xác thực và phân quyền phù hợp
- Pin GitHub Actions bằng commit SHA khi chính sách repository yêu cầu
- Quét dependency và chạy kiểm thử trong CI

Kiểm thử và chất lượng

Một thay đổi chỉ nên được coi là hoàn tất khi có bằng chứng kiểm tra tương ứng.

NGUYÊN TẮC VÀ NỘI DUNG

- Unit test cho hàm và module độc lập
- Integration test cho route và lớp dữ liệu
- Smoke test cho luồng quan trọng
- Lint, typecheck, test và build trong CI
- Không triển khai production khi chưa có phê duyệt

Lộ trình phát triển

Các năng lực dưới đây là định hướng sản phẩm. Trạng thái thực tế cần được xác nhận từ repository, test và tài liệu phát hành.

NGUYÊN TẮC VÀ NỘI DUNG

- Giai đoạn 1 — xác lập quy ước và CLI tối thiểu
- Giai đoạn 2 — router, layout và xử lý request
- Giai đoạn 3 — rendering và tích hợp dữ liệu
- Giai đoạn 4 — kiểm thử, tài liệu và ví dụ thực tế
- Giai đoạn 5 — ổn định API, bảo mật và phát hành phiên bản

Bắt đầu với một dự án

Tên lệnh dưới đây là ví dụ đề xuất, chưa phải cam kết rằng CLI đã được phát hành. Chỉ dùng lệnh cài đặt sau khi package và phiên bản đã được xác minh.

NGUYÊN TẮC VÀ NỘI DUNG

- Kiểm tra README và hướng dẫn cài đặt trong repository
- Cài phiên bản đã phát hành từ nguồn đáng tin cậy
- Tạo một route mẫu và chạy dev server
- Chạy test/build trước khi đưa lên môi trường staging

Ví dụ quy trình khởi tạo (đề xuất, chưa xác nhận CLI phát hành):

```
npm create ocodejs@latest my-app
```

Chỉ sử dụng khi package chính thức đã tồn tại và lệnh được ghi nhận trong tài liệu dự án.

Hệ sinh thái Velclaw

Velclaw là lớp nhận diện và hệ sinh thái định hướng công cụ phát triển. OCode.js là framework; hai tên có vai trò riêng, không gộp thành một sản phẩm không phân biệt.

NGUYÊN TẮC VÀ NỘI DUNG

- OCode.js — framework và quy ước lập trình
- Velclaw — hệ sinh thái, công cụ và trải nghiệm nhà phát triển
- Tích hợp chỉ được công bố khi có hướng dẫn và kiểm thử
- Liên kết dự án chính thức: GitHub organization/repository của Velclaw

Sẵn sàng xây dựng

OCode.js by Velclaw

NGUYÊN TẮC VÀ NỘI DUNG

- Rõ ràng trong kiến trúc
- Có thể kiểm thử trong quy trình
- An toàn trong cấu hình
- Mở rộng theo nhu cầu sản phẩm
- Bản giới thiệu định hướng — kiểm tra repository để biết tính năng nào đã triển khai thực tế

OCode.js by Velclaw — xây dựng có chủ đích, xác minh bằng mã nguồn.