Drupal Core Strategy (Draft)

December 2, 2024

Winning aspiration for Drupal Core?

Drupal Core is the platform for building customized CMS solutions.

Drupal Core is a platform that empowers both low-code users and experienced developers to create powerful, user-friendly CMS solutions. It supports a broad spectrum of projects, from no-code tools like Drupal CMS, which allow marketers and content creators to manage and publish content effortlessly, to advanced, high-code architectures designed by skilled developers, including headless CMS solutions.

Where will Drupal Core play?

Drupal Core will transition from being a general-purpose content management system to a specialized platform for building customized CMS solutions. This shift narrows core's target audience from website builders to various CMS builders, including large organizations, digital agencies, and industry-specific platform creators.

With the introduction of Drupal CMS (Drupal Starshot), Drupal Core no longer needs to serve as both a product and a framework. Drupal CMS handles the product side, allowing Drupal Core to concentrate solely on being a powerful application/platform for CMS builders.

External target audience

Drupal Core will focus on organizations and individuals building and maintaining customized versions of Drupal, including:

- **Large organizations with multiple, or very complex, Drupal websites:** These organizations range in size and complexity but seek a standardized, repeatable approach or one with significant customization.
    - Examples: Johnson & Johnson's 'Canvas', Nestle, European Commission, Australian Government with GovCMS, UK's LocalGov, Germany's deGov.

- **Digital agencies:** These agencies need a reliable starting point for delivering CMS solutions to different clients, incorporating best practices and serving as a training foundation.
    - Examples: 1xInternet, Palantir, Vardot (with VarBase), Dropsolid (with Rocketship), Smile (with Sobki).
- **Organizations or industry groups building specialized platforms:**
    - Examples: Thunder, OpenSocial, Drupal CMS.
- Any organization that already has sites built on Drupal core and are maintaining them long term, this will include Drupal CMS sites after they are launched.
- Experienced Drupal developers and site builders who are comfortable hand-picking modules and recipes to build out a site, but are not necessarily going to create their own repeatable CMS for this purpose.

Internal target audience

- Contributors to Drupal Core and versions of Drupal built on Drupal Core.

Why is this interesting?

1. **Leveraging strengths:** Drupal Core excels with CMS-builder features like field API, views, and recipes that enable the creation of reusable CMS solutions. This shift allows us to focus on and enhance these core strengths. At the same time, a focused Drupal core provides more space for contributed modules to develop and consolidate, providing functionality that Drupal core no longer tries to do.
2. **Market differentiation:** This approach sets Drupal apart in a crowded CMS market. While many platforms cater only to end-users or website builders, Drupal core focuses on CMS builders, occupying a specialized niche that aligns with its strengths and community expertise. This will also more clearly differentiate Drupal Core from Drupal CMS by ensuring that their feature set is complementary rather than competing.
3. **Expanding Drupal's reach:** By encouraging the development of industry-specific solutions, this approach opens new markets and use cases for Drupal. It enables diverse, specialized solutions, allowing Drupal to penetrate industries it might not have reached as a general-purpose CMS. Something which currently can require undoing or hiding certain core features.

Competitors

- Web Development Frameworks
    - Examples: Laravel, Next.js, Django
    - Why they compete: Organizations might choose these to build custom CMS solutions from scratch, without Drupal Core. These frameworks offer complete flexibility and control over the architecture. Teams with strong development capabilities might prefer this route to avoid any perceived constraints of Drupal's system, even if it means more development work. They may believe that a

from-scratch solution will be more tailored to their specific needs or easier to maintain long-term without Drupal's upgrade cycles.

- Headless CMS Solutions
    - Examples: Contentful, Strapi
    - Why they compete: Companies looking for flexible, API-first solutions might choose these over building with Drupal Core. Organizations prioritizing a decoupled architecture might find dedicated headless CMS solutions more straightforward to implement than configuring and maintaining Drupal Core for headless use. They may perceive these solutions as more targeted for headless use cases, or prefer the fact that they are often available as SaaS.
- Specialized CMS Products
    - Why they compete: Industry-specific solutions might be chosen over creating a custom solution with Drupal Core. These products often come with industry-specific features and workflows built-in, which could save significant development time compared to building them on Drupal Core. Organizations might choose these for faster time-to-market, industry compliance, or because they believe a specialized solution will better meet their needs without extensive customization.
- WordPress
    - Why it competes: WordPress's extensive ecosystem and "good enough" perception might deter organizations from choosing Drupal Core for custom CMS development. Its larger market share means a wider availability of plugins, themes, and developers, potentially leading to lower development and maintenance costs. Organizations might choose WordPress because they perceive it as easier to use for content editors, or because they believe they can achieve their goals with less complex development work compared to building a custom solution with Drupal Core.

Summary of competitor Advantages

- Web Frameworks: Provide better developer tools, flexibility and documentation
- Headless CMS: Offer better API-first capabilities and documentation for decoupled architectures
- Specialized CMS: Have robust industry-specific out-of-the-box experience
- Wordpress: perceived as quicker to get started with and cheaper to run.

How will we win?

To succeed in our strategic shift towards becoming the platform of choice for CMS builders, we need to focus on two key aspects:

1. Attracting CMS builders to use Drupal Core as their platform of choice.

2. Ensuring that the CMS solutions created with Drupal Core are appealing to their target users, including both no-code site builders and developers.

To win, we will focus on the following 4 strategic tracks:

1. **Tools for CMS builders:** As explained above, this allows us to differentiate ourselves in the market by focusing on our strengths in building customized CMS solutions.
    - Include Project Browser with support for Recipes
    - Modernize content modeling capabilities
    - Make it easier for people to build third-party integrations
2. **Improved ease of use and reduced costs of ownership:** Ease of use and cost reduction are essential for attracting CMS builders and staying competitive. While Drupal once had unique features that justified a higher cost, many of these features have become standard across platforms. To remain competitive, in absence of clear differentiators, we must focus on simplifying the user experience and lowering costs of ownership.
    - Streamline the Drupal upgrade process through automated version updates
    - Improve the content editor experience with a modern, user-friendly interface
    - Provide comprehensive, clear documentation focused on specific user needs
    - Standardize development environments to reduce setup complexity
3. **Accelerated innovation and modernization:** In the fast-paced world of web development, staying ahead of industry trends and continuously innovating is essential for maintaining relevance and meeting evolving user needs. By prioritizing innovation and modernization, Drupal can remain a leader in the CMS space, provide cutting-edge solutions, and attract top talent who want to work with the latest technology.
    - Modernize the frontend with component-based design
    - Simplify the PHP development experience by improving internal consistency and alignment with modern practices and Symfony
    - Implement faster development processes with increased automation and experimentation
    - Establish channels for faster feedback loops between developers, site builders, and end-users
4. **Improved developer and contributor experience:** A strong and engaged community is essential for the long-term success and growth of an open-source platform like Drupal. By investing in the developer and contributor experience, Drupal can foster innovation, attract top talent, and ensure the platform's sustainability.
    - Streamline contribution processes, governance and decision-making
    - Improve documentation, onboarding and mentoring programs for new contributors
    - Develop a faster process for integrating community contributions into core

What capabilities must be in place?

(The "Capabilities" section is still a work in progress and requires further discussion among the Core Committers. While we are not seeking feedback on it at this stage, we've included it here to provide additional context. The "Management" section has been omitted entirely for now, but we have a draft of that as well.)

Development excellence

- Modern technical foundation
    - Well-maintained, modern PHP codebase aligned with current best practices for the backend
    - Modern JavaScript for the front-end, being prototyped by Experience Builder
    - Comprehensive testing infrastructure that ensures reliability
    - Strong security team and processes that maintain Drupal's security reputation
    - Reduce technical debt in Core
    - Define future state of our technology stack; how to do new things in modern way
- Platform building tools
    - Powerful content modeling system (Field API, entity system)
    - Robust API framework for building customized solutions
    - Recipe system for packaging and sharing CMS configurations
    - Ability to effectively integrate with third-party services
- Developer tooling excellence
    - Modern development workflow and tooling
    - Efficient local development environment
    - Strong debugging and profiling capabilities
- Innovation capability
    - Balance between innovation and stability
    - Ability to ship experimental features quickly
    - Efficient contribution review and approval systems

Strong user experience focus

- Ability to create and maintain intuitive, accessible administrative interfaces
- Expertise in modern UX patterns for content management
- Systems for gathering and acting on user feedback
- Expertise in conducting usability research with CMS builders

Documentation excellence

- Ability to produce and maintain clear, comprehensive end user documentation
- Ability to provide clear technical architecture documentation
- Capability to maintain documentation as a project strength and differentiator

Strong cost reduction focus

- Automated update processes that work across different hosting environments
- Ability to maintain predictable security release cycles
- Capability to provide simple, reliable upgrade paths
- Focus on keeping hosting requirements efficient

Contributor development program

- Ability to attract new contributors
- Ability to facilitate effective cross-project collaboration
- Capability to develop specialized skills (frontend, UX, etc.)
- Mentoring program for developing technical leaders
- Capability to sustain contributor engagement through effective recognition

Community governance excellence

- Ability to maintain and communicate clear strategic direction
- Ability to coordinate focused working groups
- Ability to maintain effective organizational partnerships, such as with the Drupal Association
- Capability to coordinate between Drupal Core and distributions such as Drupal Starshot/CMS
- Clear processes for cross-project collaboration