Acquia®

API-First Drupal: On the Road to Publish Once, Access Everywhere

Whether you call it headless or decoupled, using API-first Drupal as a central data service can power your entire application ecosystem.

Table of Contents

Introduction

Drupal, one of the largest open-source projects in the world, is the digital platform that propels global enterprises, governments, higher education institutions, and NGOs. The Drupal community provides organizations with every functionality, module, and distribution necessary to deliver exceptional digital experiences. Today, these experiences need to be omnichannel, interact with numerous applications, and require API-first architecture.

What does it mean to be API-first? It refers to the interaction between a central web service and a variety of applications, which allows the two systems to exchange data over a network. This exchange isn't limited to websites but extends to mobile applications, wearables, and Internet of Things devices. Using API-first Drupal means decoupling the front end and using another technology for the theming layer and presentation.

At Acquia, our team sees an enormous variety of enterprise Drupal use. Some situations are best addressed by decoupling the Drupal brain from its theming layer and using Drupal as a data repository to support a robust technical network. Others have needed the exciting new capabilities of JavaScript Model-View-Controler (MVC) frameworks, such as AngularJS or Ember, but wanted the back-end capabilities of Drupal.

Although responsive design can address the proliferation of different device form factors among consumers of web content, some situations cry out for native mobile apps. And finally, consider the ultimate goal for the Internet: publish once, access everywhere. In addition to a full Drupal site, some teams have taken data from Drupal to power television broadcasts, native mobile web apps, signs, and numerous other presentation devices.

No matter the technical obstacle, Drupal can provide the solution. Read the following case studies to learn how decoupled Drupal can fit into your digital environment.

How Does it Work?

So how do you go about getting data out of Drupal without using the theming layer? Actually, this arrangement is nothing new. Although Drupal has functioned as a services layer in various applications for years, evolving internet trends have recently brought a lot of attention to this configuration, along with new names such as headless, decoupled, and API-first. Developers have found more use cases for Drupal as a Web service due to the rise of mobile apps and JavaScript MVC frameworks.

Drupal's back end is very tightly coupled to its presentation layer. If you take that out, you can lose a lot, including forms, control over layout, the ability to preview content, and other aspects of Drupal that users have come to expect. Other risks include introducing an additional point of failure, loss of in-place and in-context editing, and sacrificing the improved performance capabilities of Drupal 8. First things first, before adopting an decoupled Drupal approach, all organizations must be sure that the benefits outweigh the losses.

As a developing technology, web services come in many flavors, but the one standard that seems to be winning over most of the web is the RESTful API. Representational State Transfer (REST) uses the standard HTTP protocol to enable communication between devices connected to a network. Devices are not limited to computers and may include smartphones, banking systems, televisions, and IoT devices, among other things. Its widespread acceptance across the web makes it the leading API approach of choice for Web services. Nonetheless, there are other non-RESTful approaches which Drupal is enabled for as well, such as GraphQL.

Several contributed modules enable you to add web services to a Drupal installation without writing code. Two notable examples are the Services module, which has been around a long time, and the RESTful Web Services module. Using these modules, a developer can configure a server that will allow the Drupal installation to push or allow data to be pulled as needed using the REST API. Whether the action is push or pull, Drupal is acting as a services layer. Users can add content using Drupal's content management, user, and permission systems, but the information is sent outside the Drupal context.

Drupal 8 has also introduced new capabilities such as an out-of-the-box REST API that is built in core, and enables operators to interact with content entities like nodes, users, taxonomy terms, and comments. Four Drupal 8 REST modules have been developed in core to provide better off-the-shelf functionality and to guarantee that Drupal is API-first.

Powering a Multitude of Devices

Web services can send data to any digital device, not just to computers and smartphones. Coupled with Drupal's ability to integrate easily with other systems, API-First Drupal can serve as a powerful brain for complex systems. An API-first approach positions Drupal at the center of many technical ecosystems and simplifies the distribution of content between numerous applications.

Presentation Devices on Major Cruise Line

Cruise ships strive to give their passengers the most enjoyable experience possible. Passengers depend on a daily newsletter for information about events, security, and how to find locations on the ship. For one major cruise line, the choice to move this newsletter online to improve guest experience was an obvious one.

The company had used Drupal for its shipboard intranet for years. The same advantages that made Drupal the CMS of choice for that project also held true due to flexibility, dependability, and a passionate open-source community. The IT team was already fluent in Drupal and knew how to find drupal.org contributed projects for most functionality needs.

The daily newsletter developed into a full-fledged onboard passenger app, customized for each passenger's interests. Guests can find dynamic event information, details about ports visited, current weather and heading, menus, and stateroom account details. Guests can also use free instant messaging to stay in touch with each other. The app has fundamentally changed the way a passenger can find information on the ship, providing a unique experience for every cruise passenger. With Drupal, the possibilities are endless, and many additional services are in the pipeline.

In addition, the team realized that using Drupal as a services layer, as they did for the smartphone version of the app, could be extended. Drupal powers the free video-on-demand service as well as digital signage, sending content to 120+ screens around the ship. The Drupal brain controls every passenger facing screen on the ship.

The company began with its newest ship and has now implemented the system on two others. The system is so robust that it can be deployed to any ship within a month's time. Passenger response exceeded expectations, and the team has rolled it out to half the fleet. The company plans to install the system on the entire fleet by the end of 2015.

Leveraging other Front-end Technologies

Rendering websites is a dance between the client (in most cases the web browser) and the server. In the past, most of the work was done on the server, with simple interactions handled in the client by jQuery. This process works extremely well for many websites.

Recent developments on the web have created demand for more flexibility in the client. In some cases, data needs to be updated for millions of users in millions of places, and accessing that data through a central server is inefficient. In other cases, an application interacts with the user and manipulates data the user enters, leading to long lag times if presentation is processed on the server. Other situations demand an interactive presentation that is more application than website. These are just three examples; the list goes on.

With more logic executed in the client, JavaScript libraries have grown large and difficult to maintain. JavaScript frameworks, on the other hand, promise increased productivity and maintainable code. Many exist, some with variations on the MVC model. Some of the most popular include Ember, React, and AngularJS.. Drupal's approach focuses on the server and may come up short for an application with heavy frontend needs. In this case, Drupal can function as a services layer to allow content created in the Drupal CMS to be presented through a JavaScript framework.

Nutrition Education Company

A nutrition education company, steeped in Drupal, moved all their training online in 2009. Very attuned to the student experience, they quickly realized that to be most effective, the system should be rebuilt.

Significant effort went into fully understanding student needs and experience prior to developing any code for the new system. Being committed Drupalists, there was no doubt that Drupal would provide the back end: registration, user management, and content creation. But the experience the developers envisioned required a dynamic system and an application that was more than a website. This experience included an application that tuned to each individual user, and changed in real time with each interaction with the system. They needed a client-side framework for the front end and chose AngularJS.

The new system functions as a Learning Management System with the simplicity of modern web design. Students can use the device of their choice: desktop, tablet, or smartphone, all powered by the Drupal back end. The system gives ready access to all class materials, including video and audio files and saves student progress with such precision that it knows where the student left off in each media file.

Students sees progress in each activity, updated as it happens, including the completion percentage of a current media file or quiz, overall class completion and the overall progress toward graduation. Students can easily study and search course assets, track progress, and interact with other students, at the time and with the device they choose.

Integrating with Multiple Systems

Drupal is the content management system (CMS) of choice for the enterprise. For companies with a developed web presence, this often means converting complex web properties to Drupal websites. In some cases this requires organizations to introduce Drupal to the back end in order to support existing technical systems.

Major Athletic Apparel Company

A major purveyor of athletic apparel needed to streamline their web operations. The company used different e-commerce platforms in different countries and information-only websites for countries in which they were not licensed to sell. They wanted one platform that could create a consistent user experience across all websites.

Acquia Commerce offers such a platform. Drupal is well-known for its ability to integrate with other systems. Acquia has leveraged that quality to provide a Drupal installation that can sit on top of any e-commerce system, providing all the advantages of Drupal integrated with the existing e-commerce platform. This integration sounded like what the company was looking for, but such an overhaul would require extensive changes. They wanted to try Acquia Commerce without disrupting the current US website's user experience.

This meant adapting Acquia Commerce to function in the middle of a Demandware installation, pulling information from the Demandware back end, allowing content manipulation using Drupal, and then pushing the data to a Demandware front end. Drupal functioned as a content management services layer. This transitional phase was quite successful, and the company plans to move ahead will a full implementation of Acquia Commerce using Drupal as a front end for all web properties.

Progressively Decoupled Drupal

Decoupled Drupal provides the flexibility needed to support every aspect of complex technical ecosystems. However, when the entire front end is controlled by a JavaScript framework, technical teams cannot take advantage of the Drupal capabilities many developers have come to know and love. This strategy of fully decoupling negates Drupal capabilities like in-place editing, contextualized interfaces, and even protection against additional points of server failure.

An increasingly popular solution to mitigate the risks of fully decoupling Drupal is a progressively decoupled strategy. Unlike a fully decoupled architecture, progressively decoupled implementations insert a JavaScript front end into a Drupal site. JavaScript frameworks will continue to consume the REST API; however, certain areas of content can be controlled and rendered out by Drupal while others can still take advantage of client-side rendering.

Progressively decoupling preserves the front-end experience of JavaScript while technical teams can continue to take advantage of valuable Drupal capabilities.

Major Weather Company Uses Progressively Decoupled Panels

A nationwide weather company needed a digital ecosystem that could withstand the unpredictable. This company provides forecasts for over 2.9 million locations, curates hundreds of dynamic weather maps, and sees an average of 50 million pages views a day. Variability is a constant as site traffic can dramatically spike during major weather events. Before moving to Drupal, this weather company's digital properties relied on 144 different origin servers, powered by three data servers. Their solution was to progressively decoupled Drupal and to architect a new Presentation Framework to deliver interactive experiences on a Drupal rendered page.

One of the largest considerations for this weather channel was determining how to accommodate for diverse performance and caching requirements. On an average page there are very different caching and time load needs across each content section. With a progressively decoupled strategy, the weather channel developed a new presentation framework that breaks pages into sections. Each individual section is called a component, which lives in its own subdirectory. A JSON file declares metadata about the component to Drupal.

Drupal then ingests these directories into panel panes. These panes are exportable and reusable, and can be created by front-end developers with minimal involvement from the back end. With progressively decoupled panes, this weather channel can specify regional content, push uniform reports, and personalize content that is not cacheable and needs to be rendered on the client side. A progressively decoupled Drupal strategy enabled the large weather company to fulfill all of their sites' diverse technical needs. JavaScript developers could continue to work in JavaScript, and editorial teams had an intuitive way to create pages without extensive development involvement.

Controlling All Aspects of Media

Drupal as a services layer can serve as a central repository to send video and data to the many outlets available in the current media marketplace. This of course includes native iOS and Android apps, as described in other examples. But it can also include sending data to live broadcast television, posting to YouTube, and accessing future outlets as they become available.

College Athletic Conference Television Network

A college athletic conference owned several live television channels and all the media associated with them. They managed their hundreds of thousands of media assets through a patchwork of content management systems, which offered a slow and awkward arrangement that did not allow them to provide the level of access and user experience they wanted for their fans. They chose to build a new hub for their operations using Drupal. Not only did Drupal integrate with every member school and streamline operations, but it also provided a bridge between athletics and the top-rated IT departments of member schools, who were already running tens of thousands of Drupal instances.

Drupal proved an excellent choice. With Drupal, the team could organize all assets and their metadata and make them available to fans, both video on demand and live events. In addition, the team, with the help of a top-rated Drupal shop, developed a scheduling application for the over 3000 events scheduled by the conference every year. Each school gained easy access to the application, their data integrated into the website for the entire conference without access to the full platform.

Key to the success of the project was the ability for Drupal to output RESTful APIs. Not only did Drupal serve as the repository for all athletic assets and data from member schools, it also sent those assets to native iOS and Android apps. Fans could now access current and past games from any device.

Through the services layer available with Drupal, the organization could better control the media experience. For example, they wrote their own APIs to push live video to YouTube at the conclusion of an event. Drupal also pushed data to the conference's linear TV platform and distributed it to all major cable and satellite television companies.

The system comes very close to achieving the tantalizing goal of publish once, access everywhere, and Drupal's ability to provide data to native mobile apps, member schools, YouTube, and live television is a big part of that. The project structure is API-driven and modular, scalable, and provides flexible architecture that allows for new functionality and sites with no limit on innovation and growth. Through Drupal, the team will keep the conference's fans engaged and delighted now and for years to come.

Conclusion

Using Drupal as a services layer is a practice that is approaching maturity, and is supporting the digital ecosystems of some of the biggest brands in the world today. While there are many diverse applications of API-first architectures, approaching an API-first solution many seem intimidating. At Acquia, we can help you pull all the pieces together, and connect you with the experts who have experience in your particular situation.

Drupal is a very complete system that can handle almost anything you need done. If you suspect that your needs go beyond Drupal’s typical offerings, have a conversation with a team member at Acquia. You might find that Drupal can handle your project on its own. If not, Acquia can help find whatever solution best fits your needs.