

# Developers win faster with true composability

Developer





Developers are the architects, builders, and problem-solvers behind the technology stacks that power our modern world. However, those technical professionals are prone to run into obstacles—in particular tedious, slow, and outdated tools—that hinder their ability to efficiently deliver high-quality output.

### Heavier workloads

Even though developers are responsible for coordinating, building, and integrating software components, custom integrations are labor-intensive and time-consuming, requiring APIs to resolve dependencies and intricate interactions. The manual process involved, buried in a plethora of IT tickets, drastically impedes workflows.

### Loss of agility and control

With a legacy platform, developers must rely on the vendor's roadmap or third-party services to integrate new functionality or build new features, lagging behind rapidly-changing trends and business needs.

### More manual work, fewer value-add projects

Developers spend a load of time and effort adapting custom code, handling dependencies, and ensuring smooth interactions—so much so they are hard pressed to debug issues or optimize performance. As a result, productivity and innovation suffer.



## Connectivity is the key to business success

In today's age of digital transformation, **connectivity is the crucial enabler** for brands to win and retain customers. As consumer behaviors evolve, so do the technologies with which developers can plan sprints effectively, focus on value-add features, and innovate with the tools they love.

72%

Or nearly three-quarters of customer interactions are now digital.

58%

Fifty-eight percent of all consumers state that they're more likely to browse products online than in a store.

77%

Seventy-seven percent of organizations with a headless architecture assert that headless enables faster implementation of changes to storefronts.

70%

Seventy percent of developers will leverage low-code and no-code solutions by 2025.

Even though well aware that technology and customer buying habits are intertwined, digital teams are bogged down by technologies that are at odds with customer needs and preferences. Rigid platforms effectively become digital time capsules, freezing developers in place and preventing them from keeping up with industry shifts and customer-driven trends.



# Brands are pivoting to composable

Organizations are battling for consumer attention and market share in an economy that rewards speed and penalizes inaction. Omnichannel, once on the leading edge of connecting to customer journeys, is now paramount for survival.

Research shows that since omnichannel strategies drive roughly [80 percent](#) of in-store customer visits, they can raise orders by nearly [500 percent](#) compared to the single-channel approach. To stay competitive, brands are going composable to modernize outdated systems, overcome operational bottlenecks, provide relief to overburdened teams, and, ultimately, pave a path to omnichannel success.

## The steady shift to composability is being driven by three factors:

- **Composable solutions beat traditional ones in the race for digital transformation.** The black-box architecture on which traditional content-management systems (CMSes) are based is difficult to upgrade and scale. Also, depending on the vendor's roadmap, legacy CMS features in an all-in-one approach don't always extend across the platform.
- **Composable enables agility and flexibility.** Decoupled or headless API-first services—CMSes, commerce, digital asset management (DAM)—are the new favorites for building omnichannel experiences. By going composable, rather than being locked in a single-vendor, expensive contract, brands can efficiently deliver personalized content with best-of-need tools.
- **Composable separates winners from losers.** Gartner reports that [60 percent](#) of organizations are switching to composability. Brands that have already done so will outpace competition by [80 percent](#) as flexibility, adaptability, and scalability become crucial for innovation and omnichannel delivery.

## What does this mean for a developer?

A composable architecture enables developers to efficiently design and build cutting-edge features for cross-channel digital experiences because its technologies simplify the integration process and yield more flexibility and adaptability. With composable, developers—

- Gain the freedom to use front-end frameworks that work best for their unique requirements.
- Adapt tools to quickly respond to market changes and strategic priorities.
- Reduce time spent on building a new stack and maintaining legacy systems while allocating bandwidth for value-added development tasks.

# What is composable?

Composable architectures comprise vendor solutions that work together and communicate with one another through APIs. By implementing a composable stack, you can select the solutions—a customer-data platform, a marketing-automation platform, and systems for managing content and digital assets—that best meet your needs from the vendors you prefer.

## Key advantages

- **Flexibility**  
You choose the tools that best serve your needs.
- **Best-of-need approach**  
Instead of settling for a monolithic system's prepackaged capabilities, you adopt the solutions of your choice.
- **Experimentation**  
You test new tools, vendors, and frameworks in an environment in which it's easy and intuitive to integrate or update technologies.
- **Scalability**  
Your digital stack can grow as the business expands.
- **Multivendor environment**  
Free from the constraints of a single system, you—not the vendor—are in control of the tools that power your customer experience.

A composable architecture might look something like this:





# Going composable is hard

Despite the many promises of composable, digital teams that adopt composable architectures still face numerous challenges.

Chief among those concerns are the following:

## Slow and costly implementation

Connecting composable systems is time-consuming and costly. Developers must learn new APIs to build and maintain custom glue code, let alone that personalization is difficult to apply and manage.

## Bottlenecks

Due to the technical complexity and developer-centric API approach of composable, marketers need developer help while building, updating, and launching digital experiences—a process that creates friction between teams and negatively impacts the scalability and agility of projects.

## Complexity in development

Composability's decoupled approach means that no two architectures are the same. Developers who move on to other horizons invariably take with them valuable knowledge of the system, and new hires face a steep learning curve.

## What are the problems that negatively impact developers?

- **Slow user adoption.** Though modularity is central to a composable approach, multiservice stacks can impede and complicate onboarding for new users.
- **Delayed sprints.** As marketing tickets mount, developers are inundated with tasks that deprive them of adequate time to focus on enhancing site performance.
- **Clunky glue code.** Developers must write complex yet low-value code to connect composable systems to make them work together.
- **Stifled innovation.** Due to the challenges in updating the stack, developers are stuck with software that's been overtaken in the market and that doesn't support key features.

**To be truly composable**, brands need a layer of technology to visually orchestrate components. That way, developers are freed from having to custom-code composable elements, and marketers can dedicate themselves to building and maintaining experiences in a visual workspace.

[Learn more in a demo](#)



# A technology stack that empowers digital teams

A major benefit of composable architecture is its use of best-of-need products. However, composable tools lack prebuilt connectors through which to work together and a visual interface that affords business users ownership and control over digital experiences.

To help developers focus on building features instead of getting buried in marketing requests, composable workspaces must perform two tasks:

1. Eliminate the tech debt, high costs, and lengthy timelines associated with building and maintaining custom integration code.
2. Balance the needs of developers and marketers so that they can collaborate in parallel while prototyping and adopting new technologies.

**The solution?** A visual workspace that empowers digital teams to quickly blend content, data, and technologies from anywhere to deliver winning multichannel experiences.

A new product category, coined by Gartner as digital experience composition, contains a visual experience layer, a win-win for organizations. Developers can then readily integrate composable solutions for digital teams to quickly build omnichannel experiences all by themselves and avoid being left in the backlog of IT tickets.



# What is digital experience composition?

Missing from the composable equation are prebuilt integrations for headless tools— content management systems, commerce, personalization—that shave weeks or even months from the integration process. Running alongside headless systems, a digital experience composition platform (DXCP) features a visual workspace that seamlessly combines composable, disparate technologies into a cutting-edge and highly adaptable stack with three components:

- **Experience builder**

Digital teams use no-code tools to build and manage high-performance digital experiences without developer involvement.

- **API integration**

Prebuilt connectors for most major composable services lend more flexibility and agility to technology stacks. Developers can focus on innovating instead of coding and maintaining integration logic for those services.

- **Front-end orchestration**

Developers work with the front-end components that best meet their needs. Digital teams can update and publish experiences faster without involving technical teams.

## How does DXCP help developers?

- **Greater flexibility and agility.** Developers are free to work with the tools they like and know best instead of proprietary software.
- **Faster stack implementation.** Building a stack without manual glue code or expensive replatforming translates to efficiency and sound economy.
- **Lower attrition and higher efficiency.** A consistent interface flattens the learning curve for novice content authors, and live preview from all sources reduces errors while improving efficiency.
- **Speed to market and boosted performance.** Freed from experience-creation and delivery tasks, developers can tackle development-oriented projects that deliver more compelling customer experiences and add business value.
- **Stronger collaboration across teams.** With no bottlenecks or team dependencies, workflows become smooth and frictionless.

[Learn more in a demo](#)





# Benefits of Uniform DXCP's visual workspace

Developers are keen to adopt headless and composable tools but often face objections from business teams that are forced to relinquish control to IT. Moreover, since content has no real home in composable architectures, usability and personalization become problematic.

**With Uniform DXCP's workspace, developers gain major advantages:**

## Agility

With composable multivendor tools, technical teams need not tackle publication tasks, and business teams can escape IT backlogs.

## Compatibility

Business teams can quickly build digital experiences, and developers work faster in a composable environment with the tools of their choice.

## Sustainability

Maintaining a long-term technology stack is much more streamlined because of the elimination of integration needs.

## Affordability

Adding, changing, and removing solutions in the team workflow is simple and straightforward without incurring the time, effort, and costs associated with replatforming.

## Availability

Thanks to the elimination of integration needs and the availability of a low-code or no-code environment for business teams, technologies become more feasible for small and mid-size organizations that lack extensive technical resources.

With uniform, developers can—

- Orchestrate composable tools.
- Personalize digital experiences.
- Enable advanced API connections.
- Create omnichannel experiences.
- Run A/B tests in a low-code or no-code environment.

[Learn more in a demo](#)