Adobe

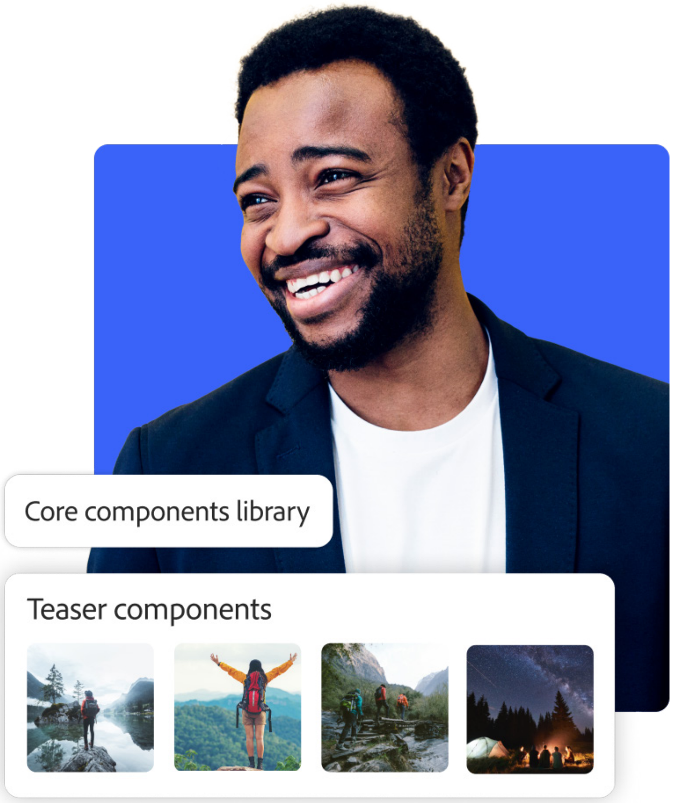
Core Components playbook

Mastering core components.

A complete guide to working with authors and developers.

Crafting a website that satisfies organizational requirements can be a daunting task, whether you're working from scratch or migrating existing builds into a new codebase. Fortunately, proper planning with Adobe Experience Manager can ease that challenge at all stages of your site's lifecyle. In this playbook, we'll dive into key considerations and insightful questions to ask your authors and developers. This will help you collaborate more effectively and drive success together.

This is what you need to think about to be a rock star Experience Manager user, and in a lot of ways, building a site is a lot like being in a rock band.



Key		
🔴 Questions for authors	🔵 Questions for developers	🩵 Questions for both

What kind of band do you have? What's the status of your Experience Manager instance?

Solo songwriter: You're building your first site with Experience Manager.

Step	Instruction	Track
1	● Determine Your Structures: ● Site structure 🔵 Will you be adding additional segments or regions later? 🔵 Do you need to think about scale now? 🔴 Will content be shared? 🔴 What pages will be primary navigation vs. secondary navigation? 🔵 Consider SEO implications. 🔴 Do you need permissions set for control of specific content? ● Assets and AM structure 🔴 Will you have shared content? 🔴 Will that content be specific to segments or regions? 🔴 Do you need permissions set for control of specific content? ● Taxonomy and structure 🔴 What are your categories or topics? 🔴 How do you want to bucket or categorize content? 🔴 Are any tags unique per piece of content? 🔵 How do you want to search for or index content? ● Template structure 🔵 Will you need flexible templates? 🔵 Will you need fixed templates? 🔵 Will you need a combination of the two (pre-baked flexible templates)? ● How are you authoring content and what do you need? 🔵 Are core components sufficient? 🔵 Do you need customization of components? 🔵 Do you have outside data? 🔵 API? 🔵 Can you use content fragments? 🔵 Do you need a custom content fragment model? 🔵 Can experience fragments help you customize?	Author and developer

Duo: You're migrating multiple sites to an existing Experience Manager codebase.

Step	Instruction	Track
2	• Are you built to scale? • Do you need localization? • Do you need user group controls? • Do you need translations? • Can you use content fragments and experience fragments for variations of languages? • Do you need taxonomy translations? • Do you have shared content in multiple regions? • Will you use Blueprint? • Will you use live copies?	Author and developer

Band: You're building out multiple sites on multiple Experience Manager codebases.

3	• Repeat steps one and two for Songwriter and Duo for new codebases.	Author and developer

Legendary band: You're consolidating sites to one codebase or consolidating and eliminating technical debt.

4	• What is working and what isn't working? • Can you still be scalable while consolidating down? • Can you learn from previous mistakes on one codebase and eliminate them on the new one? • Can you optimize the authoring UX? • Can you optimize how things are built as well as performance? • What can be cut?	Author and developer

Scalability.

Consider scalability early and often, or you might need to record multiple takes.

Step	Instruction	Track
5	● Ask yourself these questions: ● What is your minimum viable product day one? ● What do you need for the foreseeable future from a build standpoint? ● What do you need to scale long term against long term goals?	Author

Component structure.

Card Component - One chord (component) multiple keys.

6	● Can you group component functionality by component type? ● Do you have a need for multiple variations? ● If so, is it for these reasons: • Dynamic data input • Authoring versatility • Design or brand governance • Legal, compliance, regional or business unit governance • Design nuance • Personalization	Author

Dynamic data.

Card Component - Dynamic Article Card Variation.

7	● Is there information in your design that is shared from component to component? ● Evaluate HTML structure ● CSS overlap ● JS functionality ● Similar data inputs ● How do you optimize the authoring experience in your component dialogs? ● Can you remove authoring difficulties by customizing and utilizing dynamic data to drive the creation of the component? ● What can you automate with development time after time on your current components?	Author and developer

Regional requirements.

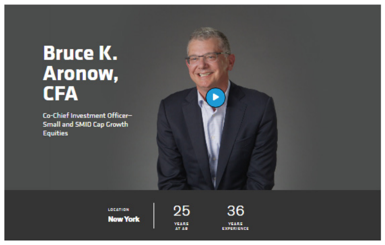
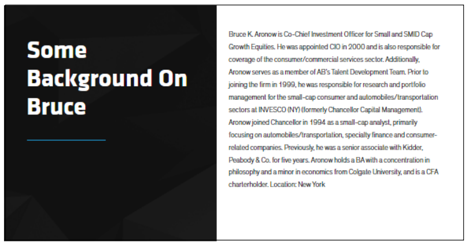
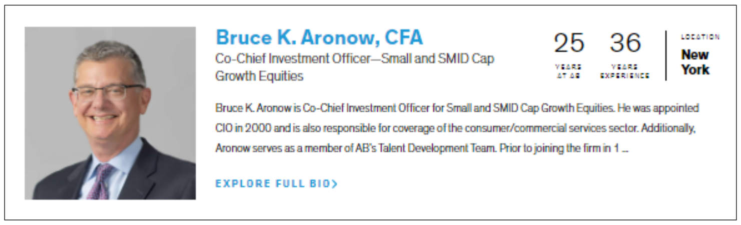
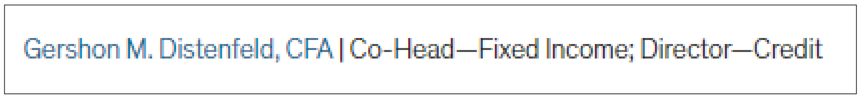
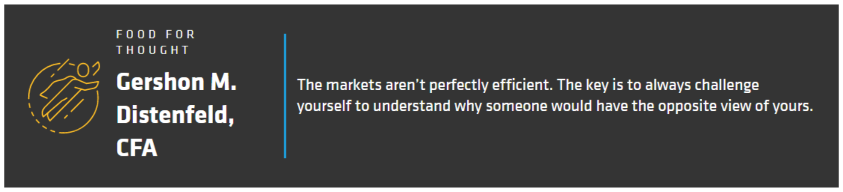
Hero Component – Regional nuance globally shared.

Step	Instruction	Track
8	● Hero component – Regional nuance globally shared? ● Do you have components that are globally shared? ● Do you have specific components per region? ● Can you create variations on a singular component to allow for regional nuances? ● Are there compliance implications for who can view or use different component variations? ● How can you improve the authoring dialog experience? ● Create screenshots previews of the component variation to use. ● Create instructional text or tooltips in dialogs to help describe use cases. ● Name components and variations intentionally. ● Can you utilize site structure or templates to mitigate risk of your regional requirements? ● Allow only certain countries or regions to see components or templates or access component or templates.	Author

Component data fed from content fragments.

Bio Component - Content Fragments

9	● Can you use content fragments as a data source to power dynamic and reusable components? ● Can you automate experience or page creations with either content fragments or experience fragments? ● Can you create different dynamic variations utilizing the same content fragment model? ***Example***: This slide shows a component that leverages five different component variations of our bio snippet component. These variations are leveraged from the information entered in our custom bio content fragment model and then rendered based on the variation and the specific information need from the model to support it.	Author and developer

Step	Instruction	Track
9	Here's what the bio snippets look like: **Detail variation**  Bruce K. Aronow, CFA Co-Chief Investment Officer—Small and SMID Cap Growth Equities LOCATION New York | 25 YEARS AT AB 36 YEARS EXPERIENCE  Some Background On Bruce Bruce K. Aronow is Co-Chief Investment Officer for Small and SMID Cap Growth Equities. He was appointed CIO in 2000 and is also responsible for coverage of the consumer/commercial services sector. Additionally, Aronow serves as a member of AB's Talent Development Team. Prior to joining the firm in 1999, he was responsible for research and portfolio management for the small-cap consumer and automobiles/transportation sectors at INVESCO (NY) (formerly Chancellor Capital Management). Aronow joined Chancellor in 1994 as a small-cap analyst, primarily focusing on automobiles/transportation, specialty finance and consumer-related companies. Previously, he was a senior associate with Kidder, Peabody & Co. for five years. Aronow holds a BA with a concentration in philosophy and a minor in economics from Colgate University, and is a CFA charterholder. Location: New York **Long variation**  Bruce K. Aronow, CFA Co-Chief Investment Officer—Small and SMID Cap Growth Equities 25 YEARS AT AB 36 YEARS EXPERIENCE | LOCATION New York Bruce K. Aronow is Co-Chief Investment Officer for Small and SMID Cap Growth Equities. He was appointed CIO in 2000 and is also responsible for coverage of the consumer/commercial services sector. Additionally, Aronow serves as a member of AB's Talent Development Team. Prior to joining the firm in 1 ... EXPLORE FULL BIO › **Text only variation**  Gershon M. Distenfeld, CFA | Co-Head—Fixed Income; Director—Credit **Hightlight variation**  FOOD FOR THOUGHT Gershon M. Distenfeld, CFA The markets aren't perfectly efficient. The key is to always challenge yourself to understand why someone would have the opposite view of yours.	Author and developer

Importance of discovery and requirements gathering.

Audience

10	• Are you building a site with an audience from one country? One region? Multiple regions? The globe? • Will you need to support other regions in the future if you are not scaled now?	Author and developer

Components

Step	Instruction	Track
11	• Do you have a need for unique customization (snowflake components) due to specific requirements? • Do you want your components to be as reusable as possible and less dev dependent? • How many atomic components do you want to use in your component library? • How many components will be molecular in nature and act more as containers?	Author and developer

Branding

12	• Are you building a new design? • Are you carrying over a design from a previous build that you have to keep the same? • Are you doing a mix of both? • How will these designs affect your decision for customization of components on Experience Manager?	Author and developer

Build

13	• Are you building a new site for the first time on Experience Manager? • Are you migrating a site or sites from a different CMS or codebase? • Are you migrating a site or sites from a different codebase on Experience Manager?	Author and developer

Functionality

14	• How much of your content is fully authorable? • How much of your content is fully dynamic? • API? • Dataset? • Content Fragment? • Service? • How much of your content requires both? • Where do these instances occur and how often?	Author and developer

Find similarities, address your gaps.

Overlap

Step	Instruction	Track
15	• Determine a visualization exercise to evaluate what you need versus what you have: ◦ Create a "war room" where you print pages and use a virtual board of your envisioned website and evaluate gaps and similarities in website elements. . ◦ Flag what components you already have vs what components you need to build or evaluate. • Then determine the following: ◦ Is there correlation of HTML structure in multiple components where HTML structure is shared? ◦ Are there components that share the same look and feel, CSS styles, colors, or fonts? ◦ Are there certain components that share functionality in how they render in the front end or how they collect information from the back end? • Are there areas where components are animated or highly interactive in similar ways? • If you have already built some components or you are using core components, but you are looking to add more or add an enhancement, can you leverage something that already exists? ◦ Can you just build off of that and customize it? ◦ Do you have to build something net new to support the requirement?	Developer

Learn from your builds.

Codebases

16	• Your first codebase is like a debut album. Your first build is going to be a lot of work but it will be a huge learning experience. Do your best to account for everything you will need to scale larger than you currently are. • A second codebase feels like a sophomore album because if you are deciding to iterate, there will be a lot of pressure for it to perform at least as well as the first codebase, but also work better than the first one did. Every new build you should be applying things you learned previously. Keep the components, structures, workflows, templates, and other elements that worked well. Leave the things that didn't behind. • Your third codebase and beyond is like a greatest hits album. You now have two examples to examine against successful practices. Use that to your advantage to build and scale to the best of your ability and eliminate technical debt, like songs that shouldn't make the greatest hits cut.	Developer

Step	Instruction	Track
16	● Try to also consolidate and audit components consistently and what is working in your Experience Manager instance. ● Always consider regression as you build and where that will become taxing for development and QA to manage long term.	Developer

Think outside the "out of the box" components.

Customization of Components

17	● Experience Manager Core Components might work for your instance, but only you are going to know if they will or won't work against your requirements. ● Hit formulas in music usually have certain things in common. But not all hit songs are alike. Most songs are 3:30 min long, in 4/4 time, have a melodic hook and are in a major key. But, for example, the hit song Money by Pink Floyd was 6:22 min, in 7/4 time, the hook was a bassline, and it was in a minor key. So, a formula does not necessarily make a hit. ● The same is true for components. Core components follow the hit formula. They work for a lot of customers, but they don't always work for your needs against requirements. If you can't leverage your Core Component out of the box, you will need to embrace customization and do one of the following: ● Extend the core component and augment it to fit your needs. ● Create a totally custom component with a custom authoring dialog.	Developer

AB component library.

Component library as a design system

18	● How do you plan to govern your component usage? ● How do brand guidelines drive what you represent on your website? ● How do you govern authors use of your components? ● How can you customize your components to leverage them to govern your business and brand and design rules? ● How will you train people to use your system? ● Consider building your training as a folder in your QA environment to automate training and include: • Tutorials • Videos • New component enhancement process • Governance	Developer

Step	Instruction	Track
18	● This will also show live changes and help with QA testing. ● This will showcase how to author variational use cases by viewing the dialog in the authoring experience as the example is shown. ● How will you be catalog and categorize your components and how to use them? ● Can you group them by types? ● Can you group them by variations?	Developer

Top tracks.

Key takeaways

19	● Evaluate your current state. That way, you know where you need to go next to build and scale. ● Do your due diligence. Don't skimp on gathering as much information as possible to drive decision making that will make an impact in your Experience Manager instance. ● Future scalability matters. Try to build in an agile way so you have opportunities to keep growing. Invariably, you will need to. ● Customize when needed. Don't be afraid to customize and go away from what is suggested. No system can fully account for any one company's specific requirements and that's what makes your build unique. ● Allow for variational growth. Instead of always building things in a wholly new way, determine where you can leverage what already exists and build more iteratively. ● Enable Learning and Training. You want your system to be truly self-service in order to achieve success. This becomes even truer the more you scale.	Author and developer

Tips for author and developer handoff.

Authors and business stakeholders should gather all requirements they need before they start talking to development.

- What are absolutes? (legal and compliance driven)
- What are must haves?
- What are nice to haves?
- What is required for MVP vs. Post MVP fast follow?
- How much author customization is necessary?

- If you are solutioning for multiple businesses and regions:
  - Do any of them have excusive specificity?
  - Are any designed differently?
  - Do they have differing functional requirements?
  - Do they have ave different data intake requirements?
  - Are there rules on who can author and edit where?

Walk through the “why” behind the “what” with development on these requests:

- Allow development to solution based upon the goal not the solution that you think you need.
- Be open to consolidation if you start to build too much or need too many variations.

Opportunities for design strategy.

- Train your design team on how Experience Manager works first (make sure they understand the power of reusability and component-based design).
- Platform owners, designers and developers should collaborate closely to be in alignment and support each other through iteration.
- Remind everyone to build off what you already have existing first. If that's not possible, then customize or make something new.

Customization: New, intermediate, advanced.

Customization will really come with time and experience. New users can certainly customize. In fact, it is how you learn, but you will most likely iterate a lot until you get it close to right.

Intermediate users will find the customization patterns that start to make sense and the patterns will showcase where to drive the changes. Advanced users can tell where things need to be customized almost instantaneously.

These are power users that know all the variations and use cases that exist in the system, so they know what they can and cannot do at a moment's notice. That knowledge provides the ability to customize smarter and negate painful iterations to customization.