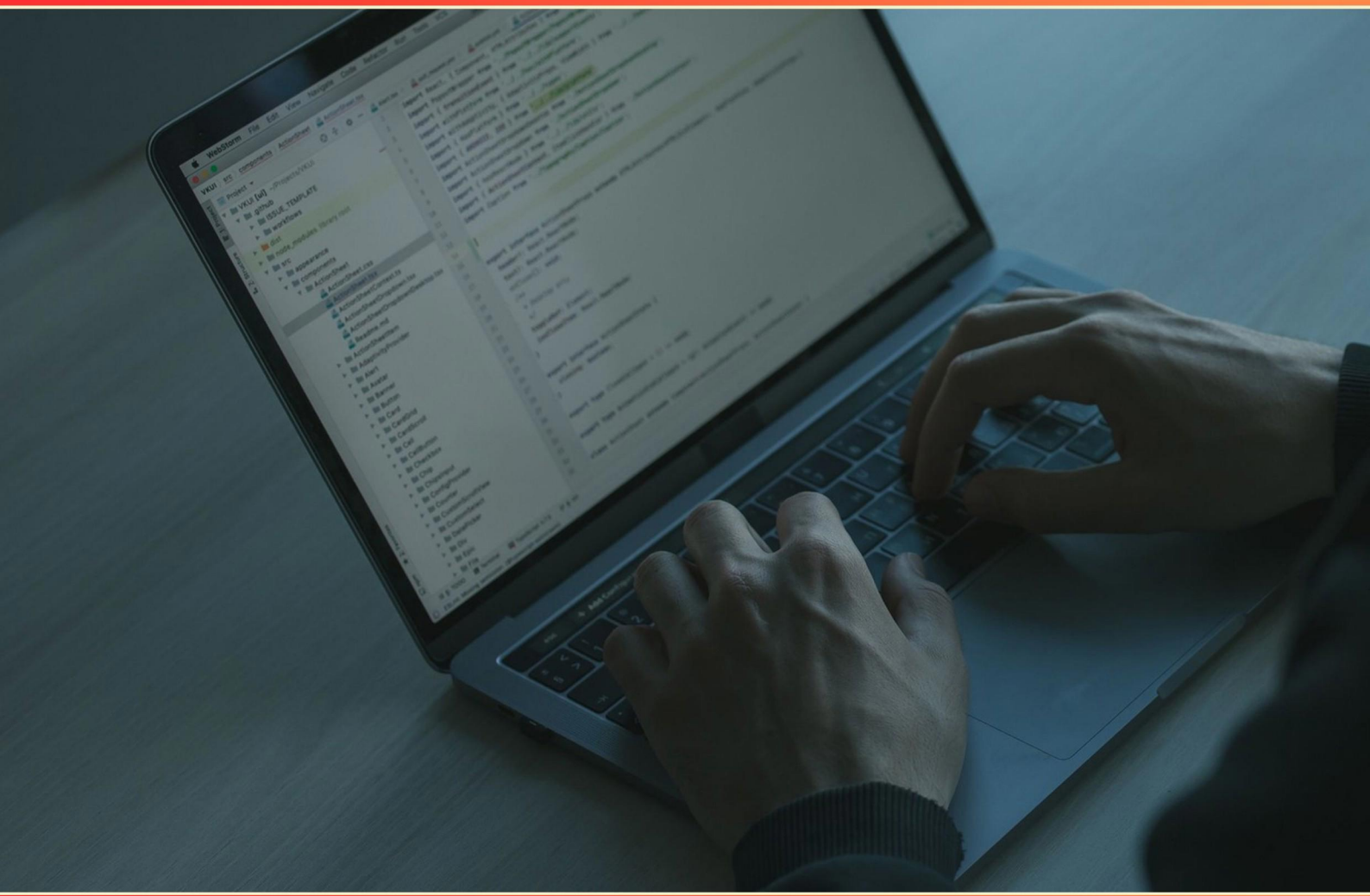
SITECORE 10 DEVELOPER PRACTICE TEST (SAMPLE)

STUDY GUIDE



SAMPLE STUDY GUIDE WITH LIMITED QUESTIONS

VISIT US ONLINE FOR THE FULL VERSION STUDY GUIDE

https://sitecore10dev.examzify.com

Copyright © 2026 by Examzify - A Kaluba Technologies Inc. product.

ALL RIGHTS RESERVED.

No part of this book may be reproduced or transferred in any form or by any means, graphic, electronic, or mechanical, including photocopying, recording, web distribution, taping, or by any information storage retrieval system, without the written permission of the author.

Notice: Examzify makes every reasonable effort to obtain accurate, complete, and timely information about this product from reliable sources.

SAMPLE

Table of Contents

Copyright 1

Table of Contents 2

Introduction 3

How to Use This Guide 4

Questions 5

Answers 8

Explanations 10

Next Steps 16

SAMPLE

Introduction

Preparing for a certification exam can feel overwhelming, but with the right tools, it becomes an opportunity to build confidence, sharpen your skills, and move one step closer to your goals. At Examzify, we believe that effective exam preparation isn't just about memorization, it's about understanding the material, identifying knowledge gaps, and building the test-taking strategies that lead to success.

This guide was designed to help you do exactly that.

Whether you're preparing for a licensing exam, professional certification, or entry-level qualification, this book offers structured practice to reinforce key concepts. You'll find a wide range of multiple-choice questions, each followed by clear explanations to help you understand not just the right answer, but why it's correct.

The content in this guide is based on real-world exam objectives and aligned with the types of questions and topics commonly found on official tests. It's ideal for learners who want to:

- Practice answering questions under realistic conditions,
- Improve accuracy and speed,
- Review explanations to strengthen weak areas, and
- Approach the exam with greater confidence.

We recommend using this book not as a stand-alone study tool, but alongside other resources like flashcards, textbooks, or hands-on training. For best results, we recommend working through each question, reflecting on the explanation provided, and revisiting the topics that challenge you most.

Remember: successful test preparation isn't about getting every question right the first time, it's about learning from your mistakes and improving over time. Stay focused, trust the process, and know that every page you turn brings you closer to success.

Let's begin.

SAMPLE

How to Use This Guide

This guide is designed to help you study more effectively and approach your exam with confidence. Whether you're reviewing for the first time or doing a final refresh, here's how to get the most out of your Examzify study guide:

1. Start with a Diagnostic Review

Skim through the questions to get a sense of what you know and what you need to focus on. Your goal is to identify knowledge gaps early.

2. Study in Short, Focused Sessions

Break your study time into manageable blocks (e.g. 30 – 45 minutes). Review a handful of questions, reflect on the explanations.

3. Learn from the Explanations

After answering a question, always read the explanation, even if you got it right. It reinforces key points, corrects misunderstandings, and teaches subtle distinctions between similar answers.

4. Track Your Progress

Use bookmarks or notes (if reading digitally) to mark difficult questions. Revisit these regularly and track improvements over time.

5. Simulate the Real Exam

Once you're comfortable, try taking a full set of questions without pausing. Set a timer and simulate test-day conditions to build confidence and time management skills.

6. Repeat and Review

Don't just study once, repetition builds retention. Re-attempt questions after a few days and revisit explanations to reinforce learning. Pair this guide with other Examzify tools like flashcards, and digital practice tests to strengthen your preparation across formats.

There's no single right way to study, but consistent, thoughtful effort always wins. Use this guide flexibly, adapt the tips above to fit your pace and learning style. You've got this!

SAMPLE

Questions

SAMPLE

1. **Where inside your solution architecture should the JSS code be housed when working with a team of JSS developers?**

   A. Feature layer

   B. Foundation layer

   C. Project layer

   D. Environment layer

2. **Which of the following is the basis for Sitecore Helix principles and conventions?**

   A. Atomic design conventions

   B. Software design rules

   C. Principles of Package Design for modular architecture

   D. Best practices for system architecture

3. **What is the role of the Sitecore Salt API?**

   A. Facilitating secure interactions between Sitecore and external applications

   B. Providing content management tools to authors

   C. Managing Sitecore user permissions

   D. Optimizing Sitecore's database performance

4. **_____ are items that typically define the configuration or structure of the implementation and contain metadata for assets in the solutions.**

   A. Definition items

   B. Content items

   C. Serialized items

   D. Item templates

5. **In a multitenant site architecture, which feature allows independent content management for each tenant?**

   A. Shared database

   B. Autonomous website

   C. Single site structure

   D. Centralized administration

SAMPLE

6. Which layer should each tenant have dedicated to their specific tenancy in a multitenant implementation?

A. Project

B. Foundation

C. Feature

D. Foundation and Feature

7. Which Sitecore deployment type provides Commerce capabilities?

A. Sitecore Experience Management XM0

B. Sitecore Experience Commerce XC1

C. Sitecore Experience Platform XP0

D. Sitecore Experience Management XP1

8. When working with SXA, which templates are allowed to exist within the Feature layer, contrary to Helix recommendations?

A. View rendering templates

B. ASP.NET Core rendering templates

C. Datasource templates

D. Feature templates

9. What does the Sitecore Context encapsulate?

A. The current state of the Sitecore application

B. The configuration settings of the Sitecore server

C. The data stored in Sitecore's databases

D. The current version of Sitecore software

10. What are the advantages of using Sitecore Helix when developing your implementation?

A. Reduced technical debt

B. More maintainable code

C. Long-term flexibility

D. Predesigned customization options

SAMPLE

Answers

SAMPLE

1. C
2. A
3. A
4. A
5. B
6. A
7. B
8. C
9. A
10. A

SAMPLE

Explanations

SAMPLE

1. Where inside your solution architecture should the JSS code be housed when working with a team of JSS developers?

A. Feature layer

B. Foundation layer

C. Project layer

D. Environment layer

In a solution architecture where multiple developers are working on a JavaScript Services (JSS) project, housing the JSS code in the project layer is a strategic choice. The project layer serves as the area where all project-specific components and configurations reside, making it the most logical location for the JSS code. The project layer typically contains elements that are unique to the specific application or project, such as custom components, configurations, and business logic that directly pertain to that project. Since JSS is intended to facilitate front-end development with Sitecore by allowing developers to build applications using modern JavaScript frameworks, housing it in the project layer ensures that all JSS components are closely aligned with the project's objectives and features. The feature layer is generally used for reusable functionality across multiple projects, while the foundation layer accommodates shared codebase utilities and services. The environment layer is concerned with infrastructure-specific setups and configurations. Keeping JSS code at the project level allows a team of developers to work collaboratively while maintaining clarity about which components and functionalities belong to the specific project at hand. This organization enhances the maintainability and scalability of the application's architecture.

2. Which of the following is the basis for Sitecore Helix principles and conventions?

A. Atomic design conventions

B. Software design rules

C. Principles of Package Design for modular architecture

D. Best practices for system architecture

The basis for Sitecore Helix principles and conventions is derived from Atomic Design conventions. Atomic Design is a methodology that breaks down user interface components into their fundamental parts, allowing for a modular and systematic approach to building design systems. In the context of Sitecore, this means developers can create reusable components that align with these principles, enhancing maintainability and scalability. By focusing on small, composable pieces that can be combined to form complex interfaces, Sitecore Helix encourages a clean and organized structure in Sitecore projects. This approach helps developers better manage dependencies and promote a clear separation of concerns. As Sitecore projects grow in complexity, adhering to these conventions ensures that components can be reused across different areas of the site, facilitating both design consistency and more efficient updates. Although the other options touch upon important aspects of development and design, they do not specifically capture the foundational approach that Sitecore Helix embodies, which is rooted in the Atomic Design methodology. This alignment with Atomic Design principles is crucial for enabling developers to implement a modular architecture effectively within Sitecore.

SAMPLE

3. What is the role of the Sitecore Salt API?

A. Facilitating secure interactions between Sitecore and external applications

B. Providing content management tools to authors

C. Managing Sitecore user permissions

D. Optimizing Sitecore's database performance

The Sitecore Salt API is designed to facilitate secure interactions between Sitecore and external applications. This API plays a key role in ensuring that data exchanges are both safe and efficient, which is crucial in a web environment where security vulnerabilities can lead to significant issues. By enabling secure communication, the Salt API supports the integration of Sitecore with various external systems, ensuring that data is transmitted safely, authentication is properly handled, and integration points are reliable. This functionality allows developers to build robust solutions that connect Sitecore with other services while maintaining high security standards, essential for protecting sensitive information during interactions with third-party applications. The Salt API's focus on secure communications is what primarily distinguishes it in the Sitecore ecosystem.

4. _____ are items that typically define the configuration or structure of the implementation and contain metadata for assets in the solutions.

A. Definition items

B. Content items

C. Serialized items

D. Item templates

Definition items are essential components in Sitecore that are designed to establish the configuration and structure of the implementation. They serve as a foundational aspect for organizing and managing assets within the Sitecore environment. These items typically hold metadata and can define relationships between different components, enabling developers to specify how content items are structured and behave. By containing metadata, definition items provide essential information about the context and purpose of other items within the system, which is crucial for properly configuring applications or solutions built on Sitecore. They help maintain a clean architecture, fostering better management practices and ensuring that all elements align with the overall project goals. Other options do not fulfill this specific role as clearly. Content items mainly represent the actual content that will be presented to users, while serialized items refer to the way data is stored and transferred, often in a format that can be easily restored or moved between instances. Item templates, on the other hand, define the structure of content items but are more about the schema rather than the overall configuration of the solution. Thus, definition items stand out as the most appropriate choice for describing items that encapsulate implementation configuration and metadata.

SAMPLE

5. In a multitenant site architecture, which feature allows independent content management for each tenant?

A. Shared database

B. Autonomous website

C. Single site structure

D. Centralized administration

In a multitenant site architecture, an autonomous website feature allows each tenant to manage its own content independently. This means that each tenant operates in a way that provides them with the freedom to create, modify, and manage their content without impacting or being impacted by other tenants in the system. This autonomy is essential in a multitenant setup as it ensures that multiple sites can coexist without interference, enabling organizations to tailor their content management processes according to their specific needs and requirements. The other options do not provide the same level of independence. A shared database typically means that all tenants are using the same underlying data structure, which could lead to complications in content management and data security. A single site structure suggests a level of uniformity across tenants that would contradict the purpose of multitenancy, where each tenant needs distinct capabilities. Centralized administration implies that the content across different tenants may be managed from a single point, which would not support the independence that is required in a multitenant environment.

6. Which layer should each tenant have dedicated to their specific tenancy in a multitenant implementation?

A. Project

B. Foundation

C. Feature

D. Foundation and Feature

In a multitenant implementation, it is essential for each tenant to have its own dedicated layer that handles tenant-specific functionality and configurations. This allows for customization and distinct management for each tenant while ensuring that the underlying system remains stable and shareable among different tenants. The Project layer is designed for this purpose. It contains tenant-specific items, such as templates, content, and any customizations needed that pertain to the unique business logic or requirements of that specific tenant. Isolating these customizations in the Project layer enables developers to maintain clear separation between different tenants' data and functionality, minimizing potential conflicts and ensuring that each tenant can operate independently while leveraging the shared resources of the platform. In contrast, the Foundation layer typically handles cross-tenant shared utilities, while the Feature layer often encompasses functionalities that can be used across multiple tenants but do not need specific customizations. Therefore, the Project layer is the most appropriate choice to ensure each tenant's specific needs are met within a multitenant framework.

SAMPLE

7. Which Sitecore deployment type provides Commerce capabilities?

A. Sitecore Experience Management XM0

B. Sitecore Experience Commerce XC1

C. Sitecore Experience Platform XP0

D. Sitecore Experience Management XP1

The Sitecore Experience Commerce XC1 deployment type specifically includes Commerce capabilities, making it the correct choice for this question. XC1 is designed to facilitate the integration of e-commerce functionalities within the Sitecore platform, enabling the management of products, pricing, and transactions. This deployment provides the necessary tools and frameworks that are essential for building and managing online stores and commerce-related activities. For organizations looking to leverage Sitecore for e-commerce purposes, XC1 stands out as it is tailored to support the unique needs of commerce applications, such as inventory management, customer engagement, and sales analytics. In contrast, the other options are more focused on content management and digital experience without inherent commerce features. For example, Experience Management XM0 and XP1 are primarily centered on managing digital content and experiences, while Experience Platform XP0 encompasses both content and digital marketing tools but does not specifically include the commerce functionalities. Therefore, XC1 is the only option that directly provides e-commerce capabilities within the Sitecore ecosystem.

8. When working with SXA, which templates are allowed to exist within the Feature layer, contrary to Helix recommendations?

A. View rendering templates

B. ASP.NET Core rendering templates

C. Datasource templates

D. Feature templates

The Feature layer in the Helix architecture promotes the use of specific templates that encapsulate functionality. However, when it comes to SXA (Sitecore JSS or Sitecore Experience Accelerator), it permits certain templates, notably datasource templates, to exist within the Feature layer. Datasource templates are integral in defining the content structure for reusable renderings. In SXA, you often create renderings that pull in data from these datasource templates, allowing for a modular approach where multiple components can utilize the same data structure. This flexibility is in line with how SXA is designed to work, emphasizing content-first experiences. In contrast, renderings such as view rendering templates or ASP.NET Core rendering templates are typically not meant to be placed within the Feature layer according to Helix. These types of renderings align more with the Presentation layer, where they denote how data should be displayed rather than the underlying content structure. Thus, their existence in the Feature layer would contradict the Helix guidelines. Similarly, the concept of feature templates within the Helix architecture typically focuses on encapsulating functionality rather than defining data structures or content types, which is why they are also not allowed in that layer. Understanding these distinctions is crucial for maintaining a clean and organized architecture in Sitecore,

SAMPLE

9. What does the Sitecore Context encapsulate?

A. The current state of the Sitecore application

B. The configuration settings of the Sitecore server

C. The data stored in Sitecore's databases

D. The current version of Sitecore software

The Sitecore Context encapsulates the current state of the Sitecore application, which includes valuable information relevant to the ongoing request being processed. This context holds details about the current user, their roles, the active language, the database in use, and the current item's state among other dynamic elements that are essential for Sitecore's operations. It essentially acts as a central point for accessing details that drive the rendering of the content being requested and the execution of workflows. While the configuration settings of the Sitecore server, the data stored in Sitecore's databases, and the current version of Sitecore software are all important aspects of Sitecore's ecosystem, they do not represent the real-time interplay and state management of an ongoing application process as encapsulated by the context. The right answer emphasizes the dynamic and interactive aspect of Sitecore's operation at any given moment during user interactions.

10. What are the advantages of using Sitecore Helix when developing your implementation?

A. Reduced technical debt

B. More maintainable code

C. Long-term flexibility

D. Predesigned customization options

Using Sitecore Helix offers several advantages that contribute to a more efficient and effective development process. One of the most significant benefits is the reduction of technical debt. This framework promotes best practices in software architecture by encouraging a modular design. This modularity allows developers to build components that are easily testable, maintainable, and reusable. As a result, by adhering to the principles of Helix, you effectively minimize the accumulation of technical debt that often arises from poorly structured codebases. Moreover, following Helix guidelines ensures a clear separation of concerns. This delineation helps in organizing code based on functionality and reduces dependencies, making it simpler to implement changes without impacting other parts of the system. Minimizing these dependencies ultimately leads to lower maintenance costs and a smoother development process long-term. By reducing technical debt through the use of Sitecore Helix, you set the foundation for a more sustainable and robust application, fostering an environment where future enhancements can be made with more confidence and less risk. This aspect is crucial as it can significantly enhance the longevity and adaptability of your Sitecore implementation, aligning well with modern agile development practices. The remaining options, while also important, focus on maintainability, flexibility, and customization options, which are indeed benefits of using Hel

SAMPLE

Next Steps

Congratulations on reaching the final section of this guide. You've taken a meaningful step toward passing your certification exam and advancing your career.

As you continue preparing, remember that consistent practice, review, and self-reflection are key to success. Make time to revisit difficult topics, simulate exam conditions, and track your progress along the way.

If you need help, have suggestions, or want to share feedback, we'd love to hear from you. Reach out to our team at hello@examzify.com.

Or visit your dedicated course page for more study tools and resources:

https://sitecore10dev.examzify.com

We wish you the very best on your exam journey. You've got this!

SAMPLE