Guidewire Developer Fundamentals Practice Exam (Sample)

Study Guide

BY EXAMZIFY

Everything you need from our exam experts!

Questions

1. **What method is employed to create a new query?**
   - **A. create**
   - **B. createQuery**
   - **C. make**
   - **D. initiate**

2. **What is a common challenge associated with paging in Guidewire?**
   - **A. Database connectivity issues**
   - **B. Repositioning of the cursor when columns are modified**
   - **C. Excessive memory usage**
   - **D. Invalid queries**

3. **For getter properties in Gosu, what should they not do?**
   - **A. Return a computed value.**
   - **B. Alter any data.**
   - **C. Have parameters.**
   - **D. Be declared as final.**

4. **Which of the following statements about Gosu classes are correct?**
   - **A. Add new classes to a customer package space.**
   - **B. Override base class methods automatically.**
   - **C. All classes must be named with a prefix.**
   - **D. Classes can only exist in the default package.**

5. **Which of the following violates best practices for logging levels?**
   - **A. Using INFO for non-critical issues**
   - **B. Turning on DEBUG for production**
   - **C. Logging errors at the ERROR level**
   - **D. Using WARN for user experience issues**

6. What document is used to outline requirements for application changes in Guidewire?

- A. Change Log
- B. User Story
- C. Design Specification
- D. Technical Manual

7. What is the logical combination of fields from ETI and ETX files used for?

- A. To create a temporary view
- B. To define a physical database
- C. To form the logical view of an entity
- D. To eliminate redundancy

8. How are GUnit tests beneficial for developers?

- A. They limit code flexibility
- B. They reduce debugging time
- C. They create complex dependencies
- D. They slow down the development process

9. What is a value of archiving in software development?

- A. It simplifies the data retrieval process
- B. It preserves application performance
- C. It eliminates the need for backups
- D. It enhances user experience

10. In a List View, what does a row typically contain?

- A. A string of text
- B. One or more Cell widgets
- C. An array of user interactions
- D. A summary of all contact details

Answers

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

Explanations

1. What method is employed to create a new query?

A. create

B. createQuery

<u>C. make</u>

D. initiate

The method used to create a new query in Guidewire applications is specifically designed to establish a new instance of a query object with the necessary parameters and configurations for data retrieval. The correct method, 'createQuery,' is part of the query creation conventions in the Guidewire platform, which aims to simplify and standardize the way developers interact with data. Choosing 'createQuery' signifies an understanding of the programming practices that Guidewire employs, and aligns with the framework's encapsulation of data operations. This methodology allows for cleaner and more maintainable code, as it explicitly indicates the intention to create a query, making it easier for developers to understand and follow. The other options do not correspond with the correct syntactical naming used within the Guidewire framework for creating queries. While terms like 'create,' 'make,' and 'initiate' may imply similar actions, they do not accurately represent the method defined by Guidewire for constructing queries. The precise nature of the naming in Guidewire's API is crucial for proper implementation and avoiding potential errors in code development.

2. What is a common challenge associated with paging in Guidewire?

A. Database connectivity issues

<u>B. Repositioning of the cursor when columns are modified</u>

C. Excessive memory usage

D. Invalid queries

Paging in Guidewire often presents the challenge of repositioning the cursor when columns in the queried data are modified. This is particularly relevant in databases where the order of records can change based on updates to the columns, which can lead to inconsistencies in the data being paged through. When pages are loaded, if the underlying data structure changes—such as when new rows are added or existing rows are modified—the position of the cursor may no longer accurately reflect the intended page of data. This concern is crucial because developers need to ensure that users experience consistent and reliable access to data across multiple pages. If the cursor does not point to the right records after changes in the database, users may see unexpected results, making it vital to handle paging effectively to maintain data integrity. Other challenges, such as database connectivity issues, excessive memory usage, and invalid queries, are relevant to broader database interactions but do not specifically highlight the nuances of paging as repositioning the cursor does. The intricacies of paging make the challenge of cursor repositioning a distinctive concern in the context of Guidewire applications.

3. For getter properties in Gosu, what should they not do?

A. Return a computed value.

B. Alter any data.

C. Have parameters.

D. Be declared as final.

Getter properties in Gosu are designed to retrieve a value associated with an object without causing any side effects or modifying the object's state. The key principle behind a getter is that it retrieves data in a way that is predictable and does not change any aspect of the system's state. When a getter alters data, it can lead to unexpected behavior and makes the property less intuitive for users of the class, who typically expect getters to simply return values. By avoiding any alterations to data within a getter, you maintain the integrity and simplicity of your code, allowing for easier maintenance and a clearer understanding of how the properties interact within the program. The other options relate to practices that are acceptable for getter properties: they can return computed values, they do not require parameters, and they can be declared as final, depending on the specific design considerations within the class. Each of these aspects contributes to flexibility in implementation but does not compromise the purpose and expected behavior of a getter property in Gosu.

4. Which of the following statements about Gosu classes are correct?

A. Add new classes to a customer package space.

B. Override base class methods automatically.

C. All classes must be named with a prefix.

D. Classes can only exist in the default package.

The statement regarding adding new classes to a customer package space is accurate because Gosu, which is the language used in Guidewire, allows developers to create new classes within a defined package structure that organizes code effectively. This capability is essential for maintaining code quality and modularity in large applications. By adding classes to a customer package, developers can extend existing functionality, implement custom business logic, and ensure that their code is easy to manage and integrate within the larger Guidewire platform. The other statements do not hold true in the context of Gosu classes. For instance, overriding base class methods is not an automatic feature; developers must explicitly define overrides for methods in subclasses. Similarly, while naming conventions are followed, there is no strict requirement that all classes must have a specific prefix. Finally, classes are not limited to the default package; they can be organized into various custom packages according to the design and structure of the project. This flexibility allows for better organization and code management, promoting a more scalable development environment.

5. Which of the following violates best practices for logging levels?

A. Using INFO for non-critical issues

B. Turning on DEBUG for production

C. Logging errors at the ERROR level

D. Using WARN for user experience issues

Turning on DEBUG level logging in a production environment is not aligned with best practices. DEBUG logging is typically used during development and testing to provide detailed insights for troubleshooting and diagnosing issues. It generates a high volume of logs, which can lead to performance degradation and increased storage costs. Additionally, the verbosity of DEBUG logs could expose sensitive information, making systems more vulnerable to security risks. In contrast, the other options adhere more closely to recommended practices. INFO logging can be used for non-critical issues as it provides useful information without flooding the logs with unnecessary details. Logging errors at the ERROR level is appropriate, as it conveys significant issues that require attention without over-stating their severity. Using WARN for issues related to user experience is also acceptable, as it allows developers to be alerted to potential problems that may not be critical but still warrant attention.

6. What document is used to outline requirements for application changes in Guidewire?

A. Change Log

B. User Story

C. Design Specification

D. Technical Manual

The document that is used to outline requirements for application changes in Guidewire is a User Story. User Stories are a component of Agile methodologies and serve as a tool for capturing the needs and requirements of users in a concise, straightforward manner. They focus on individual user needs and specify what functionality is required from the user's perspective. By framing the requirements in terms of the user experience, User Stories help teams prioritize features based on the value they provide to users, ensuring that development efforts align closely with user needs. This method promotes better communication among stakeholders, developers, and testers, as it provides a clear template for discussing what the application should do, rather than how it should be implemented. In the context of Guidewire, User Stories are valuable as they inform the development teams about the expected outcomes of application changes, guiding the work towards enhancing user satisfaction and system usability.

7. What is the logical combination of fields from ETI and ETX files used for?

- A. To create a temporary view
- B. To define a physical database
- <u>C. To form the logical view of an entity</u>
- D. To eliminate redundancy

The logical combination of fields from ETI (Entity Type Items) and ETX (Entity Type) files is utilized to form the logical view of an entity. This process helps in defining how data is organized and represented within the context of the Guidewire data model. In Guidewire, entities are representations of real-world objects or concepts, and the logical view assists developers and users in understanding how these entities interact, their attributes, and how they relate to one another within the application framework. The ETI and ETX files serve as a blueprint for defining the structure and relationships of these entities, ensuring that the application accesses and manages data consistently. This logical view is crucial for accurate data representation and ensures that applications function as intended by allowing users to work with abstracted data structures, rather than having to deal with the underlying complexities of the physical database design directly.

8. How are GUnit tests beneficial for developers?

- A. They limit code flexibility
- <u>B. They reduce debugging time</u>
- C. They create complex dependencies
- D. They slow down the development process

GUnit tests are designed to enhance the development process by significantly reducing debugging time. When developers write unit tests for their code, they create automated checks that confirm whether the individual pieces of code (or units) are functioning as intended. This proactive approach means that any errors or bugs can be identified and addressed early in the development cycle, rather than waiting until later stages when problems may be more complex and harder to trace. By utilizing GUnit tests, developers can ensure the reliability of their code, allowing them to make changes confidently, knowing that the tests will catch any unintended consequences of those changes. This leads to faster iterations and a smoother development experience, ultimately streamlining the process and improving the overall quality of the software. The other options do not align with the purpose and advantages of GUnit tests. Rather than limiting flexibility, creating complex dependencies, or slowing down the development process, GUnit tests support efficient and effective coding practices.

9. What is a value of archiving in software development?

A. It simplifies the data retrieval process

B. It preserves application performance

C. It eliminates the need for backups

D. It enhances user experience

Archiving plays a vital role in software development by preserving application performance. When a system accumulates a large amount of data over time, it can become slower and less efficient as it has to process more information. By archiving older, less frequently accessed data, the application can reduce the amount of data it actively manages. This leads to improved performance, as the system operates with a smaller, more manageable dataset, allowing for faster retrieval times and reduced processing burden on the system's resources. While other options touch on various aspects of data management, they do not specifically relate to the core benefit of archiving in maintaining optimal application performance. The retrieval process may indeed be simplified, but this is not the primary advantage of archiving itself. Similarly, archiving does not eliminate the need for backups; rather, it complements data management strategies. Enhancing user experience is also a broader goal that may be indirectly affected by improved performance but is not a direct result of archiving. Therefore, the key takeaway is that archiving specifically helps maintain and improve application performance by managing the data load efficiently.

10. In a List View, what does a row typically contain?

A. A string of text

B. One or more Cell widgets

C. An array of user interactions

D. A summary of all contact details

In a List View, a row is composed of one or more Cell widgets. These Cell widgets are responsible for displaying the data associated with that particular row. Each Cell can hold various types of content, such as text, images, or icons, which can be arranged and formatted to present the information clearly and effectively to the user. The flexibility of using multiple Cell widgets within a row allows developers to customize the appearance and layout of the List View, enhancing the overall user experience. While other options refer to elements that might be present in a user interface, they do not accurately describe the components of a row specifically within the context of a List View. For example, a string of text could be a part of a Cell, but it does not encompass the complete structure of a row. Similarly, arrays of user interactions and summaries of contact details do not define what makes up a row in this context. The focus on Cell widgets highlights the modular and structured design typical of List Views in applications like Guidewire.