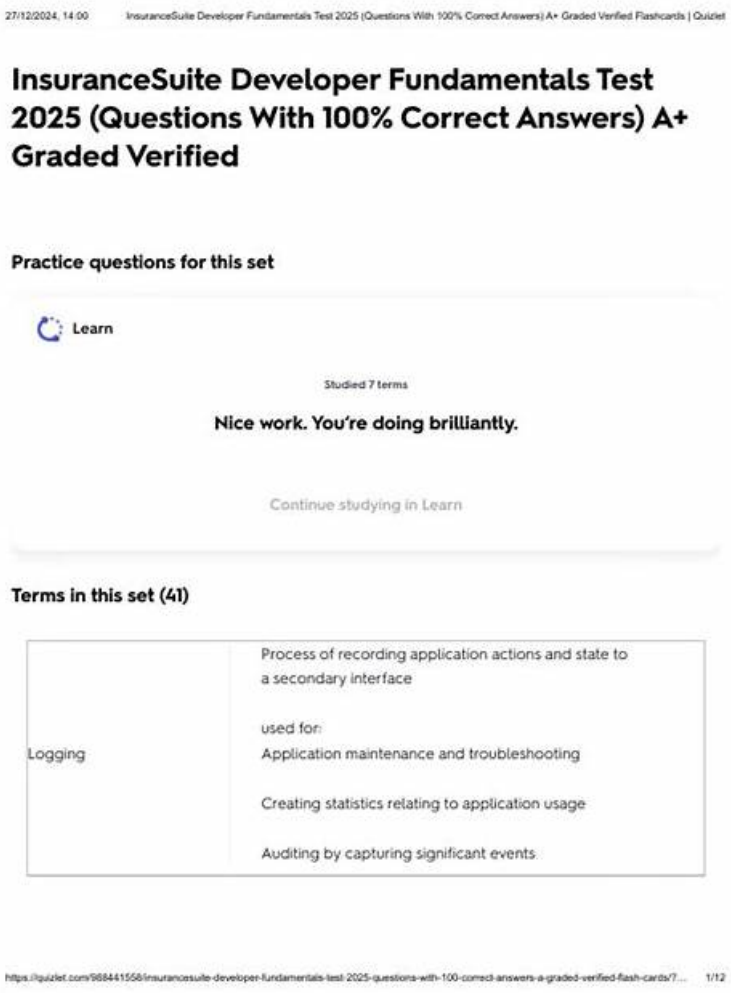
Quiz High Hit-Rate InsuranceSuite-Developer - Associate Certification - InsuranceSuite Developer - Mammoth Proctored Exam Examcollection Vce


27/12/2024, 14:00 InsuranceSuite Developer Fundamentals Test 2025 (Questions With 100% Correct Answers) A+ Graded Verified Flashcards | Quizlet

InsuranceSuite Developer Fundamentals Test 2025 (Questions With 100% Correct Answers) A+ Graded Verified

Practice questions for this set

Learn

Studied 7 terms

Nice work. You're doing brilliantly.

Continue studying in Learn

Terms in this set (41)

Logging	Process of recording application actions and state to a secondary interface used for: Application maintenance and troubleshooting Creating statistics relating to application usage Auditing by capturing significant events

https://quizlet.com/988441558/insurancesuite-developer-fundamentals-test-2025-questions-with-100-correct-answers-a-graded-verified-flash-cards/?... 1/12

P.S. Free & New InsuranceSuite-Developer dumps are available on Google Drive shared by DumpsTests: https://drive.google.com/open?id=1pOeTfwUvbksP3b1mw9xLyV_J2EfYB2iM

Why don't you begin to act? The first step is to pass InsuranceSuite-Developer exam. Time will wait for no one. Only if you pass the exam can you get a better promotion. And if you want to pass it more efficiently, we must be the best partner for you. Because we are professional InsuranceSuite-Developer Questions torrent provider, we are worth trusting; because we make great efforts, we do better. Here are some reasons to choose us.

DumpsTests almost aimed to meet the needs of all candidates who want to pass the InsuranceSuite-Developer exam. If someone who don't have enough time to prepare for their exam, our website provide they with test answers which only need 20-30 hours to grasp; If someone who worry about failed the InsuranceSuite-Developer Exam, our website can guarantee that they can get full refund. In summary, the easiest way to prepare for InsuranceSuite-Developer certification exam is to complete InsuranceSuite-Developer study material.

>> InsuranceSuite-Developer Examcollection Vce <<

InsuranceSuite-Developer Free Sample Questions & InsuranceSuite-

Developer Test Questions Answers

DumpsTests believes in customer satisfaction and strives hard to make the entire InsuranceSuite-Developer exam preparation process simple, smart, and successful. To achieve this objective DumpsTests is offering the top-rated and real Guidewire Certification Exams preparation material in three different Guidewire InsuranceSuite-Developer Exam study material formats. These Associate Certification - InsuranceSuite Developer - Mammoth Proctored Exam exam questions formats are InsuranceSuite-Developer PDF dumps file, desktop practice test software and web-based practice test software.

Guidewire Associate Certification - InsuranceSuite Developer - Mammoth Proctored Exam Sample Questions (Q131-Q136):

NEW QUESTION # 131
A developer needs to create a new entity for renters that contains a field for the employment status.
EmploymentStatusType is an existing typelist. How can the entity and new field be created to fulfill the requirement and follow best practices?

- A. Add Renter.etx under Metadata - > Entity with a column EmploymentStatus_Ext.
- B. Create EmploymentStatusType.ttx under Extensions - > Typelist with a type code Renter.
- C. Create Renter_Ext.eti under Extensions - > Entity with a typekey EmploymentStatus.
- D. Add Renter.eti under Extensions - > Entity with a column EmploymentStatus_Ext.

Answer: C

Explanation:
When adding a brand-new entity to the Guidewire data model, developers must use the Extensions directory.
According to Data Model Architecture best practices, custom entities should be defined in an .eti (Entity Internal) file.
Option A is the correct implementation. Creating Renter_Ext.eti (or simply Renter.eti depending on specific project naming conventions, though _Ext is often used to denote custom work) allows the developer to define the new object from scratch. Because the EmploymentStatus field needs to reference an existing typelist (EmploymentStatusType), the field type must be a typekey, not a column. A column is used for primitive types like strings, integers, or decimals, whereas a typekey creates a relationship between the entity and the typelist metadata.
Option B is incorrect because .etx files are used for extending existing base entities (like adding a field to Claim), not for creating new ones. Option C is incorrect because it mistakenly identifies the field as a " column " and unnecessarily adds _Ext to a field on a custom entity (usually _Ext is reserved for extending base entities to avoid future collisions). Option D is completely irrelevant to entity creation as it attempts to add a code to a typelist instead of creating a data structure for a Renter. Following the structure in Option A ensures that the new Renter entity is properly indexed, supports localization via typelists, and is fully integrated into the InsuranceSuite persistence layer.

NEW QUESTION # 132
Given this function:
929 public function checkConnection() {
930 try
931 {
932 var conn = DriverManager.getConnection(url)
933 // logic here
934 }
935 catch (e : Exception)
936 {
937 // handle exception
938 }
939 }
What action will align the function with Gosu best practices?

- A. In line 933, change DriverManager to driver Manager (camel case)
- B. Move left curly braces on lines 931, 934, and 936 to the end of the previous lines
- C. Add a comment for lines with significant code (specifically, lines 933 and 937)
- D. Change line 935 to read ' catch {e: Exception) '

Answer: B

Explanation:
The Guidewire InsuranceSuite Developer Fundamentals course emphasizes the importance of a consistent coding style to ensure that configuration code is readable and maintainable. This consistency is enforced through the Gosu Style Guide, which dictates specific rules for formatting and indentation that all Guidewire developers should follow.
One of the most foundational rules in the Gosu Style Guide concerns the placement of curly braces ({). In Gosu, as in many modern programming languages derived from C-style syntax, there are two primary styles of brace placement: " Expanded " (where the brace is on its own line) and " K & R " or " 1TBS " (where the brace is on the same line as the statement). Guidewire strictly adheres to the practice of placing the opening curly brace at the end of the line that begins the block (the " 1TBS " style).
Therefore, in the provided :
* The brace on line 931 should be moved to the end of line 930 (try {).
* The brace on line 936 should be moved to the end of line 935 (catch (e : Exception) {).
Adhering to this style is more than just a preference; it is a requirement for passing Quality Gates in a Guidewire Cloud environment. When code is pushed to a repository in Guidewire Cloud, automated inspections check for these formatting issues. Code that fails these style checks may be flagged as technical debt or even prevent a successful build if strict quality gates are enabled. By moving the braces to the end of the previous lines (Option A), the developer ensures the code matches the visual pattern of the base Guidewire application, making it easier for other team members and Guidewire support to review and maintain the code over time.

NEW QUESTION # 133
An insurer has a number of employees working remotely. Displaying the employee's name in a drop-down list must include the employee's location (e.g., John Smith - London, UK). How can a developer satisfy this requirement following best practices?

- A. Create a displaykey that concatenates the name fields and work locations
- B. Create a setter property in a Name enhancement class
- C. Define an entity name that concatenates the name fields and work locations
- D. Enable Post On Change for name fields to modify how the name is displayed

Answer: C

Explanation:
In Guidewire InsuranceSuite, the way an entity is represented in the user interface (specifically in dropdowns or "RangeInputs") is governed by itsEntity Nameconfiguration. This is defined in a specialized metadata file (usually EntityName.en).
The best practice for this requirement is todefine an Entity Name(Option A) that specifies how the object should "stringify" itself. By configuring the Entity Name for the User or Contact entity to concatenate the name and location, the developer ensures a global, consistent behavior. Anywhere that entity is referenced in a dropdown throughout the entire application, it will automatically show the formatted string. This is much more efficient than creating custom logic on every single page.
Option B (Displaykeys) is generally used for static labels or simple parameter substitution, not for defining the core identity of a data object. Option C (Setter) would modify the actual data in the database, which is not the goal-the goal is only to change how it isviewed. Option D (Post On Change) is a UI refresh mechanism and does not address the underlying logic of how a record is displayed in a list.

NEW QUESTION # 134
Succeed Insurance would like a list of all Notes related to all Policies for an Account. Which approach follows best practices for retrieving this data more efficiently?

- A. Code snippetvar policyNotes : ArrayList < Note > for(policy in account.Policies) {for (note in policy. Notes) {policyNotes.add(note)} }
- B. Code snippetvar policyNotes = account.Policies*.Notes*.DisplayName
- C. Code snippetvar policyNotes = Query.make(Note).compare(Note#Policy, Relop.Equals, policy). compare(Note#Account, Relop.Equals, account).select()
- D. Code snippetvar policyNotes = account.Policies*.Notes.toList()

Answer: D

Explanation:
In Guidewire InsuranceSuite, developers frequently need to " reach across " one-to-many relationships to collect data from nested arrays. In this scenario, the goal is to retrieve a flattened list of all Note entities associated with all Policy objects linked to a specific Account.
According to Advanced Gosu best practices, the most efficient and idiomatic way to handle this is by using the Expansion Operator (*). As shown in Option B, the syntax account.Policies*.Notes performs what is known as " collection flattening. " When the

expansion operator is applied to the Policies array, Gosu understands that it should look at every policy in that collection and access the Notes array for each. It then automatically flattens these multiple sub-collections into a single, comprehensive list of Note objects. Calling .
toList() at the end ensures the result is captured in a standard, manipulatable collection format.
This approach is vastly superior to nested for loops (Option C). Manual iteration through nested arrays is a primary cause of the " N+1 " query problem and " Bundle Bloat. " In nested loops, the system may perform a separate database fetch for every policy and then another for every note, loading every single entity into the current transaction bundle, which consumes excessive memory and CPU time. The expansion operator, however, is highly optimized within the Gosu Runtime to handle these traversals more gracefully.
Option D is incorrect because it uses a second expansion operator to retrieve the DisplayName property, resulting in a list of Strings rather than a list of Note entities. Option A, while using the Query API, is logically disconnected from the root account object already in memory and represents a more complex search- based approach rather than a relationship-based retrieval. Therefore, the expansion operator is the verified standard for efficient, readable data collection in Gosu.

NEW QUESTION # 135
Given the image:
Which container type must be added between Card and Input Column?

- A. Input Set
- B. List View
- C. Detail View PCF File
- D. Detail View

Answer: D

Explanation:
The Guidewire Page Configuration Framework (PCF) follows a strict nesting hierarchy to ensure that the layout engine can correctly render widgets on the screen. According to the InsuranceSuite Developer Fundamentals curriculum, specifically the lesson on " Container Widget Usage, " developers must understand the parent-child relationships required for different layout styles.
A Card widget is a component of a CardViewPanel, used to create tabbed interfaces within a page. However, a Card itself cannot directly host an Input Column. Instead, a Card serves as a container for other panels. To display data fields in the standard column-based layout favored by InsuranceSuite, a DetailViewPanel (commonly referred to simply as a Detail View in the Studio palette) must be placed inside the Card.
The Detail View acts as the intermediate container that establishes the data context (the row or entity being edited) and provides the grid system necessary for the Input Column. The Input Column, in turn, allows developers to align fields vertically. Without the Detail View container, the PCF would be syntactically invalid because the layout engine requires the Detail View to manage the labels and input alignment for any child columns.
Option A is incorrect because a " PCF File " is the entire document, not a widget added to a tree. Option C (List View) is used for tabular data, not column-based input layouts. Option D (Input Set) is a grouping mechanism that sits inside or alongside an Input Column but cannot serve as the parent to one. Therefore, adding a Detail View (B) is the correct and necessary step to bridge the hierarchy between the Card and its Input Columns.

NEW QUESTION # 136
.....

is true of a project requirement, We can provide real InsuranceSuite-Developer Exam Torrent & InsuranceSuite-Developer training materials in three different versions so that you can choose based on your habits.

Realistic InsuranceSuite-Developer Examcollection Vce | Easy To Study and Pass Exam at first attempt & Trusted InsuranceSuite-Developer: Associate Certification - InsuranceSuite Developer - Mammoth Proctored Exam

And we are the leading practice materials in this dynamic market, Top-level faculty and excellent educational experts guarantee high-quality Guidewire InsuranceSuite-Developer practice exam that make users pass exam certainly.

So you can choose as you like accoding to your study interest and hobbies, Make sure that you are checking out all of our InsuranceSuite-Developer training material to improve your preparation level.

- Secure High Grades in Exam using Guidewire InsuranceSuite-Developer Questions □ Download 《 InsuranceSuite-Developer 》 for free by simply searching on " www.vceengine.com " □Reliable InsuranceSuite-Developer Exam Question
- Pass Guaranteed Accurate InsuranceSuite-Developer - Associate Certification - InsuranceSuite Developer - Mammoth Proctored Exam Examcollection Vce □ Open 「 www.pdfvce.com 」 and search for ➤ InsuranceSuite-Developer □ to download exam materials for free □Latest Real InsuranceSuite-Developer Exam
- Valid InsuranceSuite-Developer Test Questions ☑ Authorized InsuranceSuite-Developer Certification Ⓜ InsuranceSuite-Developer Practice Test Pdf □ （ www.vceengine.com ） is best website to obtain □ InsuranceSuite-Developer □ for free download □InsuranceSuite-Developer Valid Test Tutorial
- Latest InsuranceSuite-Developer Examcollection Vce | InsuranceSuite-Developer 100% Free Free Sample Questions □ Copy URL ➠ www.pdfvce.com □ open and search for ▷ InsuranceSuite-Developer ◁ to download for free □Detailed InsuranceSuite-Developer Answers
- InsuranceSuite-Developer Test Tutorials □ Actual InsuranceSuite-Developer Tests □ InsuranceSuite-Developer Actual Exams □ Enter ☀ www.dumpsquestion.com □☀□ and search for ➥ InsuranceSuite-Developer □ to download for free □InsuranceSuite-Developer Associate Level Exam
- Latest InsuranceSuite-Developer Examcollection Vce | InsuranceSuite-Developer 100% Free Free Sample Questions □ Search on □ www.pdfvce.com □ for ▷ InsuranceSuite-Developer ◁ to obtain exam materials for free download □Latest InsuranceSuite-Developer Exam Cram
- Latest InsuranceSuite-Developer Examcollection Vce | InsuranceSuite-Developer 100% Free Free Sample Questions □ Easily obtain ⇛ InsuranceSuite-Developer ⇚ for free download through " www.troytecdumps.com " □InsuranceSuite-Developer Practice Test Pdf
- Pass Guaranteed Accurate InsuranceSuite-Developer - Associate Certification - InsuranceSuite Developer - Mammoth Proctored Exam Examcollection Vce □ Search on 【 www.pdfvce.com 】 for { InsuranceSuite-Developer } to obtain exam materials for free download □Valid Test InsuranceSuite-Developer Braindumps
- Actual InsuranceSuite-Developer Tests □ New InsuranceSuite-Developer Dumps Questions □ Valid Exam InsuranceSuite-Developer Preparation □ Search for ➤ InsuranceSuite-Developer □ and easily obtain a free download on ➽ www.exam4labs.com □ □InsuranceSuite-Developer Customizable Exam Mode
- InsuranceSuite-Developer Discount Code !! Valid Test InsuranceSuite-Developer Braindumps □ Actual InsuranceSuite-Developer Tests □ The page for free download of ⇛ InsuranceSuite-Developer ⇚ on （ www.pdfvce.com ） will open immediately □Valid Test InsuranceSuite-Developer Braindumps
- InsuranceSuite-Developer Associate Level Exam □ InsuranceSuite-Developer Test Tutorials □ Exam Dumps InsuranceSuite-Developer Free □ Search for ➥ InsuranceSuite-Developer □ and download it for free immediately on 【 www.examcollectionpass.com 】 □Authorized InsuranceSuite-Developer Certification
- myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, www.sg588.tw, myportal.utt.edu.tt, isd-data.net, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, myportal.utt.edu.tt, bbs.cilipan.cn, Disposable vapes

P.S. Free & New InsuranceSuite-Developer dumps are available on Google Drive shared by DumpsTests:

https://drive.google.com/open?id=1pOeTfwUvbksP3b1mw9xLyV_J2EfYB2iM