Veröffentlichungscheckliste

[1. Vorbereitung 2](#_Toc288746283)

[2. Erste Schritte zum GUI-Test 2](#_Toc288746284)

[2.1. Vorhandenes Java-Projekt importieren: 2](#_Toc288746285)

[2.2. Vespucci-Diagramme erzeugen: 2](#_Toc288746286)

[3. GUI-Tests 3](#_Toc288746287)

[3.1. Sonderzeichenprüfung 3](#_Toc288746288)

[3.2. Vespucci-Diagramme: Erstellung und Bearbeitung 3](#_Toc288746289)

[3.3. Anstoßen der Prologgenerierung aus dem Package-Explorer: 6](#_Toc288746290)

[3.4. Verhalten der Rote-Linien-Erzeugung überprüfen: 6](#_Toc288746291)

[3.5. Query-Tab in der Propertiesview: 6](#_Toc288746292)

[3.6. Ensemble Verhalten beim Verschieben (siehe Bug ID:3 und 13): 7](#_Toc288746293)

[3.7. Outlineview: 8](#_Toc288746294)

[3.8. Drag and Drop im Vespucci-Plug-In: 9](#_Toc288746295)

[3.9. Prolog-Codegenerierung überprüfen: 12](#_Toc288746296)

[3.10. Diagramm-Validierung: 12](#_Toc288746297)

[3.11. Architektur-Validierung: 12](#_Toc288746298)

[4. Anmerkungen 13](#_Toc288746299)

[5. Abschluss 13](#_Toc288746300)

# Vorbereitung

Alle beschriebenen Prozesse müssen auf einem 32-Bit und 64-Bit Windows und MacOS X 64-Bit Betriebssystem getestet werden, die Betriebssysteme können auch auf einer Virtuellen Maschine installiert sein. Entsprechend des Betriebssystems wird entweder die 32-Bit bzw. die 64-Bit Version von Eclipse verwendet.

|  |  |  |  |
| --- | --- | --- | --- |
| **Prozessbeschreibung** | **Durchgeführt** | | |
| 32-Bit | 64-Bit | OS X |
| Vespucci Projekt vorbereiten: | | | |
| Das zur Abnahme bereitstehende Vespucci Projekt in Eclipse importieren. | ☐ | ☐ | ☐ |
| In dem Projekt „de.tud.cs.st.vespucci/model“ die Datei „vespucci.genmodel“ öffnen und auf dem obersten Knoten Vespucci rechte Maustaste Generate All wählen. | ☐ | ☐ | ☐ |
| In dem Projekt „de.tud.cs.st.vespucci/model“ die Datei „vespucci. gmfgen“ öffnen und auf dem obersten Knoten Vespucci rechte Maustaste Generate diagram code wählen (Grund dieser Maßnahme ist, dass kein Code im Projekt existiert, der nach dem generieren modifiziert wurde und kein „generate not“ Statement hat). | ☐ | ☐ | ☐ |

# Erste Schritte zum GUI-Test

# Vorhandenes Java-Projekt importieren:

|  |  |  |  |
| --- | --- | --- | --- |
| **Prozessbeschreibung** | **Abgenommen** | | |
| 32-BitWin | 64-BitWin | 64-Bit OS X |
| Projekt „ChecklistenProjekt“ in Eclipse importieren. | ☐ | ☐ | ☐ |
| Vespucci-Diagramme können geöffnet werden. | ☐ | ☐ | ☐ |

# Vespucci-Diagramme erzeugen:

|  |  |  |  |
| --- | --- | --- | --- |
| **Prozessbeschreibung** | **Abgenommen** | | |
| 32-BitWin | 64-BitWin | 64-Bit OS X |
| Neues Vespucci-Diagramm kann über [rechte Maustaste auf ein Verzeichnis]🡪New🡪Other🡪Other🡪Vespucci Diagram erzeugt werden. | ☐ | ☐ | ☐ |
| Neues Vespucci-Diagramm kann über File🡪New🡪Other🡪Other🡪Vespucci Diagram erzeugt werden. | ☐ | ☐ | ☐ |
| Neues Vespucci-Diagramm im „Architecture“-Verzeichnis erstellen (Name: „ReleaseDiagram.sad“). | ☐ | ☐ | ☐ |
| Neues Vespucci-Diagramm im Root-Verzeichnis erstellen (Name: „RäleüseD&$2iagröm.sad“). | ☐ | ☐ | ☐ |
| Neues Vespucci-Diagramm im „Architecture“-Verzeichnis erstellen (Name: „Relöse$Diagram.sad“). | ☐ | ☐ | ☐ |
| In Eclipse alle Vespucci-Diagramme schließen. | ☐ | ☐ | ☐ |

# GUI-Tests

# Sonderzeichenprüfung

|  |  |  |  |
| --- | --- | --- | --- |
| **Prozessbeschreibung** | **Abgenommen** | | |
| 32-BitWin | 64-BitWin | 64-Bit OS X |
| Vespucci-Diagramm „RäleüseD&$2iagröm.sad“ öffnen. | ☐ | ☐ | ☐ |
| Über den Paletten-Menüpunkt Ensemble ein Ensemble auf der Hauptebene(weiße Diagrammfläche) erzeugen und mit dem Namen „View\s“ versehen. | ☐ | ☐ | ☐ |
| Über den Paletten-Menüpunkt Ensemble ein Ensemble „Cört$%!älßür“ auf der Hauptebene erzeugen. | ☐ | ☐ | ☐ |
| Im Ensemble „View\s“ über den Paletten-Menüpunkt Ensemble zwei Ensembles „Flashcärd“ und „Ütilities“ erzeugen. | ☐ | ☐ | ☐ |
| Outgoing-Dependency aus der Palette wählen und von „View\s“ nach „Cört$%!älßür“ ziehen. | ☐ | ☐ | ☐ |
| Ensemble „Cört$%!älßür“ wählen und über den angezeigten Pfeil, der vom Ensemble weg zeigt mit gedrückter Maustaste selektieren und auf dem Ensemble „Flashcärd“ loslassen. Im angezeigten Popup Menüpunkt die Outgoing-Dependency auswählen. | ☐ | ☐ | ☐ |
| Das Ensemble „ Flashcärd “ in das Ensemble „ Cört$%!älßür “ verschieben. | ☐ | ☐ | ☐ |
| Vespucci-Diagramm „/RäleüseD&$2iagröm.sad“ speichern und schließen. | ☐ | ☐ | ☐ |
| Vespucci-Diagramm „/RäleüseD&$2iagröm.sad“ öffnen. | ☐ | ☐ | ☐ |
| Vespucci-Diagramm „/RäleüseD&$2iagröm.sad“ prüfen ob alle erstellten Ensembles noch mit den eingegebenen Namen und allen Abhängigkeiten vorhanden sind. | ☐ | ☐ | ☐ |
| Vespucci Diagramm „/RäleüseD&$2iagröm.sad“ speichern und schließen. | ☐ | ☐ | ☐ |

# Vespucci-Diagramme: Erstellung und Bearbeitung

|  |  |  |  |
| --- | --- | --- | --- |
| **Prozessbeschreibung** | **Abgenommen** | | |
| 32-BitWin | 64-BitWin | 64-Bit OS X |
| Vespucci-Diagramm „ReleaseDiagram.sad“ öffnen. | ☐ | ☐ | ☐ |
| Über den Paletten-Menüpunkt Ensemble ein Ensemble auf der Hauptebene(weiße Diagrammfläche) erzeugen und mit dem Namen „Views“ versehen. | ☐ | ☐ | ☐ |
| Prüfen, dass im Outlineview genau die Elemente angezeigt werden, die auch im Vespucci-Diagramm enthalten sind. Weiterhin muss eine eventuell im Vespucci-Diagramm vorhandene Hierarchie auch im Outlineview erkennbar sein. | ☐ | ☐ | ☐ |
| Das Package „de.tud.cs.se.flashcards.model“ via Drag and Drop als neues Ensemble einfügen. Das neuentstandene Ensemble ist so selektiert, dass der Ensemble Name direkt editierbar ist. Der Name des Ensembles soll in „Models“ geändert werden. | ☐ | ☐ | ☐ |
| Prüfen, dass im Outlineview genau die Elemente angezeigt werden, die auch im Vespucci-Diagramm enthalten sind. Weiterhin muss eine eventuell im Vespucci-Diagramm vorhandene Hierarchie auch im Outlineview erkennbar sein. | ☐ | ☐ | ☐ |
| Die Klasse „Store.java“ im Package „de.tud.cs.se.flashcards.persistence“ via Drag and Drop als neues Ensemble einfügen. Das neuentstandene Ensemble ist so selektiert, dass der Ensemble Name direkt editierbar ist. Der Name des Ensembles soll in „Store“ geändert werden. | ☐ | ☐ | ☐ |
| Über den Paletten-Menüpunkt Ensemble ein Ensemble „Controller“ auf der Hauptebene erzeugen. | ☐ | ☐ | ☐ |
| Im Ensemble „Views“ über den Paletten-Menüpunkt Ensemble zwei Ensembles „Flashcard“ und „Utilities“ erzeugen. | ☐ | ☐ | ☐ |
| Outgoing-Dependency aus der Palette wählen und von „Views“ nach „Models“ ziehen. | ☐ | ☐ | ☐ |
| Prüfen, dass im Outlineview genau die Elemente angezeigt werden, die auch im Vespucci-Diagramm enthalten sind. Weiterhin muss eine eventuell im Vespucci-Diagramm vorhandene Hierarchie auch im Outlineview erkennbar sein. | ☐ | ☐ | ☐ |
| Ensemble „Controller“ wählen und über den angezeigten Pfeil, der vom Ensemble weg zeigt mit gedrückter Maustaste selektieren und auf dem Ensemble „Models“ loslassen. Im angezeigten Popup Menüpunkt die Outgoing-Dependency auswählen. | ☐ | ☐ | ☐ |
| Prüfen, dass im Outlineview genau die Elemente angezeigt werden, die auch im Vespucci-Diagramm enthalten sind. Weiterhin muss eine eventuell im Vespucci-Diagramm vorhandene Hierarchie auch im Outlineview erkennbar sein. | ☐ | ☐ | ☐ |
| Über den Paletten-Menüpunkt Empty Ensemble ein Empty Ensemble mit dem Namen „BorderFactory“ auf der Hauptebene erzeugen. | ☐ | ☐ | ☐ |
| Das Empty Ensemble „BorderFactory“ in das Ensemble „Views“ verschieben. | ☐ | ☐ | ☐ |
| Outgoing-Dependency aus der Palette wählen und von „Models“ nach „BorderFactory“ ziehen. | ☐ | ☐ | ☐ |
| Prüfen, dass im Outlineview genau die Elemente angezeigt werden, die auch im Vespucci-Diagramm enthalten sind. Weiterhin muss eine eventuell im Vespucci-Diagramm vorhandene Hierarchie auch im Outlineview erkennbar sein. | ☐ | ☐ | ☐ |
| Expected-Dependency aus der Palette wählen und von „Store“ nach „Controller“ ziehen. | ☐ | ☐ | ☐ |
| Expected-Dependency aus der Palette wählen und von „Store“ nach „Views“ ziehen. | ☐ | ☐ | ☐ |
| Ensemble „Store“ selektieren. Es erscheinen zwei Pfeile. Den Pfeil der vom Ensemble weg zeigt mit gedrückter Maustaste selektieren und über einer leeren Stelle loslassen. Über den Popup Menüpunkt Create Expected To🡪New Element: Ensemble ein neues Ensemble mit dem Namen „Dummy“ erstellen. | ☐ | ☐ | ☐ |
| Ensemble „Store“ selektieren. Es erscheinen zwei Pfeile. Den Pfeil der vom Ensemble weg zeigt mit gedrückter Maustaste selektieren und über einer leeren Stelle loslassen. Über den Popup Menüpunkt Create Expected To🡪Existing Element das Ensemble „Utilities“ auswählen. Eine Expected Dependency von „Store” zu „Utilites” wird angelegt. | ☐ | ☐ | ☐ |
| Prüfen, dass im Outlineview genau die Elemente angezeigt werden, die auch im Vespucci-Diagramm enthalten sind. Weiterhin muss eine eventuell im Vespucci-Diagramm vorhandene Hierarchie auch im Outlineview erkennbar sein. | ☐ | ☐ | ☐ |
| Rechtsklick auf die Hauptebene im Diagramm machen. Über Add können nur Vespucci-Elemente erzeugt werden. **Umsetzung ist bisher nur geplant!** | ~~☐~~ | ~~☐~~ | ~~☐~~ |
| Alle „Flash\*.java“-Klassen im Package „de.tud.cs.se.flashcards.ui“ via Drag and Drop gemeinsam auf das Ensemble „FlashCard“ ziehen. | ☐ | ☐ | ☐ |
| Klasse „Utilities.java“ im Package „de.tud.cs.se.flashcards.ui“ via Drag and Drop auf das Empty Ensemble „BorderFactory“ ziehen. Drop ist nicht möglich und es wird das entsprechende Maus-Icon angezeigt. | ☐ | ☐ | ☐ |
| Klasse „Utilities.java“ im Package „de.tud.cs.se.flashcards.ui“ via Drag and Drop auf Ensemble „Utilities“ ziehen. | ☐ | ☐ | ☐ |
| Empty Ensemble „BorderFactory“ wählen und über einen der angezeigten Pfeil eine Note Attachment erstellen. | ☐ | ☐ | ☐ |
| Prüfen, dass im Outlineview genau die Elemente angezeigt werden, die auch im Vespucci-Diagramm enthalten sind. Weiterhin muss eine eventuell im Vespucci-Diagramm vorhandene Hierarchie auch im Outlineview erkennbar sein. | ☐ | ☐ | ☐ |
| In- and Out-Dependency aus der Palette wählen und von „Utilities“ nach „Flashcard“ ziehen. | ☐ | ☐ | ☐ |
| In- and Out-Dependency aus der Palette wählen und von „Views“ nach „Flashcard“ ziehen. | ☐ | ☐ | ☐ |
| Expected Dependency aus der Palette wählen und von „Utilities“ nach „Controller“ ziehen. | ☐ | ☐ | ☐ |
| Prüfen, dass im Outlineview genau die Elemente angezeigt werden, die auch im Vespucci-Diagramm enthalten sind. Weiterhin muss eine eventuell im Vespucci-Diagramm vorhandene Hierarchie auch im Outlineview erkennbar sein. | ☐ | ☐ | ☐ |
| Das Empty Ensemble „BorderFactory“ in „BorderFactories“ umbenennen. | ☐ | ☐ | ☐ |
| Vespucci-Diagramm „ReleaseDiagram.sad“ speichern und schließen. | ☐ | ☐ | ☐ |
| Vespucci-Diagramm „ReleaseDiagram.sad“ öffnen. | ☐ | ☐ | ☐ |
| Es muss der Fehler „Queries of non leaf Ensembles must be derived“ angezeigt werden. | ☐ | ☐ | ☐ |
| Dieser Fehler muss behebbar sein, indem man die Query des Ensembles auf derived setzt. Nach dem Speichern des Diagramms muss der Fehler verschwinden. | ☐ | ☐ | ☐ |
| Prüfen, dass im Outlineview genau die Elemente angezeigt werden, die auch im Vespucci-Diagramm enthalten sind. Weiterhin muss eine eventuell im Vespucci-Diagramm vorhandene Hierarchie auch im Outlineview erkennbar sein. | ☐ | ☐ | ☐ |
| Vespucci Diagramm „ReleaseDiagram.sad“ speichern und schließen. | ☐ | ☐ | ☐ |
| Sicherstellen, dass bei den oben durchgeführten Drop-Operationen kein Move (Datei verschwindet aus dem Projekt-Explorer) durchgeführt wurde. | ☐ | ☐ | ☐ |

# Anstoßen der Prologgenerierung aus dem Package-Explorer:

|  |  |  |  |
| --- | --- | --- | --- |
| **Prozessbeschreibung** | **Abgenommen** | | |
| 32-BitWin | 64-BitWin | 64-Bit OS X |
| Eintrag zum Generieren (Vespucci: Generate Architectural Model) findet sich nur im Kontextmenü von „\*.sad“-Dateien. | ☐ | ☐ | ☐ |
| Aktualisieren des Package-Explorers (Nach dem Generieren einer neue „\*.pl“-Datei wird diese direkt im Package-Explorer angezeigt). | ☐ | ☐ | ☐ |

# Verhalten der Rote-Linien-Erzeugung überprüfen:

|  |  |  |  |
| --- | --- | --- | --- |
| **Prozessbeschreibung** | **Abgenommen** | | |
| 32-BitWin | 64-BitWin | 64-Bit OS X |
| Diagramm „RedLine.sad“ öffnen | ☐ | ☐ | ☐ |
| Jede Kombination von eingeklappt und nicht eingeklappt der Ensembles ausprobieren. Die Dependencies müssen immer richtig angezeigt werden (Es gibt acht verschiedenen Möglichkeiten). | ☐ | ☐ | ☐ |
| Alle Ensembles aufklappen und dann das Ensemble „layer1“ einklappen. | ☐ | ☐ | ☐ |
| Rote Linie von „Ebene2“ nach „layer1“ löschen 🡪 Die dazugehörige Abhängigkeit wird gelöscht | ☐ | ☐ | ☐ |
| Alle Ensembles aufklappen und bei „Ensemble3“ ein NoteAttachment einfügen und in dieses einen beliebigen Text schreiben. Wenn das „Ensemble2“ eingeklappt wird, müssen alle Verbindungen von und zu „Ensemble2“ weiterhin angezeigt werden. | ☐ | ☐ | ☐ |
| Die Schritte alle mittels undo rückgängig machen und per redo wiederherstellen. | ☐ | ☐ | ☐ |

# Query-Tab in der Propertiesview:

|  |  |  |  |
| --- | --- | --- | --- |
| **Prozessbeschreibung** | **Abgenommen** | | |
| 32-BitWin | 64-BitWin | 64-Bit OS X |
| Das Vespucci-Diagramm „QueryTab.sad“ öffnen und den Propertiesview selektieren. | ☐ | ☐ | ☐ |
| Die Ensembles „Vater“, „Kind“ und „Solo“ nacheinander selektieren. Bei allen drei Ensembles ist im Propertiesview der Query-Tab aktive. | ☐ | ☐ | ☐ |
| Alle anderen Elemente im Vespucci-Diagramm selektieren. Bei keiner Selektierung darf der Query-Tab im Propertiesview angezeigt werden. | ☐ | ☐ | ☐ |
| Neues Ensemble mit dem Paletten-Menüpunkt Ensemble erzeugen. Der Query-Tab muss im Propertiesview aktive sein. | ☐ | ☐ | ☐ |
| Die Größe des Texteingabebereichs füllt das Query-Tab zu allen Seiten aus. | ☐ | ☐ | ☐ |
| Das Propertiesview sowohl horizontal als auch vertikal größer und kleiner schieben. Während und nach den Größenänderungen füllt der Texteingabebereich den Query-Tab komplett aus. | ☐ | ☐ | ☐ |
| In das Textfeld im Query-Tab so viel Text schreiben das die Höhe des Textes insgesamt größer ist als die Höhe des Textfeldes. Es muss die Scrollbar aktiviert werden und der Scrollbalken am rechten Rand angezeigt werden, der es ermöglicht abschnittsweise den kompletten Text zu lesen. | ☐ | ☐ | ☐ |
| Das Propertiesview in der Höhe größer schieben. Sobald das Textfeld groß genug ist um den ganzen Text darzustellen muss die Scrollbar deaktiviert werden und der Scrollbalken verschwinden. | ☐ | ☐ | ☐ |
| Das Propertiesview in der Höhe kleiner schieben. Sobald das Textfeld zu klein ist um den ganzen Text darzustellen muss die Scrollbar wieder aktiviert werden und der Scrollbalken angezeigt werden. | ☐ | ☐ | ☐ |
| Über das Query-Tab die Query des Ensembles mit dem Namen „Vater“ auf d$“)(„!§ß0kQ, die Query des Ensembles mit dem Namen „Kind“ auf Query <ENTER> des <ENTER> Kindes und die Query des Ensembles mit dem Namen „Solo“ auf SoloqueryT%ÄÖ\*+y ändern (<Enter> meint einen Zeilenumbruch). Sicherstellen, dass beim erneuten Selektieren aller oben genannten Ensembles die geänderte Query dargestellt wird. | ☐ | ☐ | ☐ |
| Sämtliche Queryänderungen aus dem vorangegangen Test mit undo rückgängig machen. Bei jedem undo überprüfen ob es korrekt ausgeführt wurde. | ☐ | ☐ | ☐ |
| Sämtliche im letzten Test mit undo rückgängig gemachten Änderungen mit redo wiederherstellen. Bei jedem redo überprüfen ob redo korrekt ausgeführt wurde (Query des Ensembles überprüfen). | ☐ | ☐ | ☐ |

# Ensemble Verhalten beim Verschieben (siehe Bug ID:3 und 13):

|  |  |  |  |
| --- | --- | --- | --- |
| **Prozessbeschreibung** | **Abgenommen** | | |
| 32-BitWin | 64-BitWin | 64-Bit OS X |
| Das Vespucci-Diagramm „EnsembleVerschiebenTest.sad“ öffnen | ☐ | ☐ | ☐ |
| Die Ensembles „Verschieben1:“ und „Verschieben2:“ nacheinander in das Ensemble „Behälter2:“ verschieben. | ☐ | ☐ | ☐ |
| Die Ensembles „Verschieben3:“ und „Verschieben4:“ in das Ensemble „Behälter3:“ verschieben. | ☐ | ☐ | ☐ |
| Die Ensembles „Verschieben5:“ und „Verschieben6:“ nacheinander auf die Hauptebene des Vespucci-Diagramms verschieben. | ☐ | ☐ | ☐ |
| Die Ensembles „Verschieben6:“ und „Verschieben7:“ nacheinander auf die Hauptebene des Vespucci-Diagramms verschieben. | ☐ | ☐ | ☐ |
| Es muss dieselbe Anzahl Ensembles im Diagramm enthalten sein wie vor dem verschieben (18 Ensembles). | ☐ | ☐ | ☐ |
| Für alle Ensembles die verschoben wurden und ein Kind Ensemble hatten muss gelten das diese immer noch das selbe Ensemble als Kind Ensemble haben. | ☐ | ☐ | ☐ |

# Outlineview:

|  |  |  |  |
| --- | --- | --- | --- |
| **Prozessbeschreibung** | **Abgenommen** | | |
| 32-BitWin | 64-BitWin | 64-Bit OS X |
| Das Vespucci-Diagramm „Outlineview.sad“ öffnen. | ☐ | ☐ | ☐ |
| Das Outlineview muss im komplett ausgeklappten zustand folgende Form haben: | ☐ | ☐ | ☐ |
| Sämtliche Elemente im Outlineview nacheinander selektieren. Es muss das entsprechende Element im Vespucci-Diagramm selektiert werden. Außerdem muss das im Outlineview selektierte Element selektiert bleiben. | ☐ | ☐ | ☐ |
| Sämtliche Elemente im Vespucci-Diagramm nacheinander selektieren. Es muss das dazugehörige Element im Outlineview selektiert werden. Bei Dependencies muss nur einer der Dependency Pfeile im Outlineview selektiert werden. | ☐ | ☐ | ☐ |
| Den Namen des Ensembles „Kind“ in „Sohn“ umbenennen. Nach der Namensänderung muss diese direkt im Outlineview zu sehen sein. | ☐ | ☐ | ☐ |
| Im Ensemble „Vater“ ein Ensemble mit dem Namen „Tochter“ erstellen. Direkt nach der Erzeugung des Ensembles muss dieses auch im Outlineview zu sehen sein. Nach der Namensgebung „Tochter“ muss dies auch direkt im Outlineview zu sehen sein. | ☐ | ☐ | ☐ |
| Ein Ensemble auf der Hauptebene erstellen. Diese muss direkt nach der Erstellung im Outlineview zu sehen sein. | ☐ | ☐ | ☐ |
| Elemente im Vespucci-Diagramm in folgender Reihenfolge löschen: 1. Ensemble „Solo“  2. Ensemble „Sohn“  3. Ensemble „Vater“  4. Empty Ensemble „empty“  5. Text „Textfeld auf der Hauptebene“ 6. Node „Text zum Kind Ensemble“  Nach jeder Lösch-Operation, dürfen die direkt und evtl. automatisch mitgelöschten Elemente nicht mehr im Outlineview angezeigt werden. | ☐ | ☐ | ☐ |
| Jede im vorherigen Test durchgeführte Lösch-Operation einzeln mit undo rückgängig machen. Bei jedem undo-Schritt müssen die wiederhergestellten Elemente wieder im Outlineview angezeigt werden. | ☐ | ☐ | ☐ |

# Drag and Drop im Vespucci-Plug-In:

Folgende Tests müssen für folgende Elemente aus dem Package Explorer ausgeführt werden:  
**Unterstützte Elemente:**

* Java-Datei
* Package
* Attribut
* Methode

**Nicht unterstützte Elemente:**Alle nicht unterstützten Elemente.  
Dies wird aus Komplexitäts- und Zeitgründen nur exemplarisch anhand folgender Elemente durchgeführt:

* \*.PDF-Dateien
* \*.SAD-Dateien
* \*.PL-Dateien
* \*.c
* Ordnern

|  |  |  |  |
| --- | --- | --- | --- |
| **Prozessbeschreibung** | **Abgenommen** | | |
| 32-BitWin | 64-BitWin | 64-Bit OS X |
| Neues Vespucci-Diagramm mit dem Namen „DND.sad“ erstellen. | ☐ | ☐ | ☐ |
| Allgemein | | | |
| Drop von unterstützten Elementen aus selbst erstellten Quellcode auf die Hauptebene erzeugt ein neues Ensemble, mit zum Drop-Objekt passendem voll Qualifizierten Namen und Query. | ☐ | ☐ | ☐ |
| Drop von unterstützten Elementen aus einem Referenzierten Package auf die Hauptebene erzeugt ein neues Ensemble, mit zum Drop-Objekt passendem voll Qualifizierten Namen und Query. | ☐ | ☐ | ☐ |
| Drop von unterstützten Elementen die in einem Element definiert sind (z.B. innere Klassen) aus selbst erstellten Quellcode auf die Hauptebene erzeugt ein neues Ensemble, mit zum Drop-Objekt passendem voll Qualifizierten Namen und Query. | ☐ | ☐ | ☐ |
| Drop von unterstützten Elementen die in einem Element definiert sind (z.B. innere Klassen) aus einem Referenzierten Package auf die Hauptebene erzeugt ein neues Ensemble, mit zum Drop-Objekt passendem voll Qualifizierten Namen und Query. | ☐ | ☐ | ☐ |
| Ein Drop von nicht unterstützten Elementen auf die Hauptebene ist nicht möglich. Es wird das entsprechenden Maus-Icon angezeigt. | ☐ | ☐ | ☐ |
| Ein Drop von nicht unterstützten Elementen auf ein Ensemble erweitert die vorhandene Query um ein gültiges Prolog Statement, welches die fallengelassenen Elemente beschreibt. | ☐ | ☐ | ☐ |
| Ein Drop von nicht unterstützten Elementen auf ein Ensemble ist nicht möglich. Es wird das entsprechende Maus-Icon angezeigt. | ☐ | ☐ | ☐ |
| Ein Drop, der unterstützte und nicht unterstützte Elemente enthält, ist nicht möglich. Es wird das entsprechende Maus-Icon angezeigt. | ☐ | ☐ | ☐ |
| Drops, die als Ziel weder ein Ensemble noch die Hauptebene haben, sind nicht möglich. Es wird das entsprechende Maus-Icon angezeigt. | ☐ | ☐ | ☐ |
| Drag and Drop Operationen aus dem Src-Verzeichnis | | | |
| Das Package „de.tud.cs.se.flashcards.model“ per Drag and Drop auf die weiße Fläche des Diagramms ziehen.  Der Name des Ensembles lautet „de.tud.cs.se.flashcards.model“  und die Query „package('de.tud.cs.se.flashcards.model')“. | ☐ | ☐ | ☐ |
| Die Datei „de.tud.cs.se.flashcards.model.Flashcard.java“ per Drag and Drop auf die weiße Fläche des Diagramms ziehen.  Der Name des Ensemble lautet „de.tud.cs.se.flashcards.model.Flashcard“ und der Inhalt der Query ist „class\_with\_members('de.tud.cs.se.flashcards.model','de.tud.cs.se.flashcards.model.Flashcard')“. | ☐ | ☐ | ☐ |
| Die Klasse „de.tud.cs.se.flashcards.model.Flashcard“ per Drag and Drop auf die weiße Fläche des Diagramms ziehen.  Der Name des Ensemble lautet „de.tud.cs.se.flashcards.model.Flashcard“ und der Inhalt der Query ist „class\_with\_members('de.tud.cs.se.flashcards.model','de.tud.cs.se.flashcards.model.Flashcard')“. | ☐ | ☐ | ☐ |
| Die Feld „de.tud.cs.se.flashcards.model.Flashcard.answer“ per Drag and Drop auf die weiße Fläche des Diagramms ziehen.  Der Name des Ensemble lautet „de.tud.cs.se.flashcards.model.Flashcard.answer“ und der Inhalt der Query ist „field('de.tud.cs.se.flashcards.model','de.tud.cs.se.flashcards.model.Flashcard','answer','java.lang.String')“. | ☐ | ☐ | ☐ |
| Den Konstruktor der Klasse „de.tud.cs.se.flashcards.model.Flashcard.Flashcards()“ per Drag and Drop auf die weiße Fläche des Diagramms ziehen.  Der Name des Ensemble lautet „de.tud.cs.se.flashcards.model.Flashcard.Flashcard“ und der Inhalt der Query ist „method('de.tud.cs.se.flashcards.model','de.tud.cs.se.flashcards.model.Flashcard','<init>','void',[])“. | ☐ | ☐ | ☐ |
| Die Methode der Klasse „de.tud.cs.se.flashcards.model.Flashcard.removeObserver“ per Drag and Drop auf die weiße Fläche des Diagramms ziehen.  Der Name des Ensemble lautet „de.tud.cs.se.flashcards.model.Flashcard.removeObserver“ und der Inhalt der Query ist „method('de.tud.cs.se.flashcards.model','de.tud.cs.se.flashcards.model.Flashcard','removeObserver','void',['de.tud.cs.se.flashcards.model.FlashcardObserver'])“. | ☐ | ☐ | ☐ |
| Die Methode der Klasse „de.tud.cs.se.flashcards.model.Flashcard.getRemembered“ per Drag and Drop auf die weiße Fläche des Diagramms ziehen.  Der Name des Ensemble lautet „de.tud.cs.se.flashcards.model.Flashcard.getRemembered“ und der Inhalt der Query ist „method('de.tud.cs.se.flashcards.model','de.tud.cs.se.flashcards.model.Flashcard','getRemembered','java.util.Date',[])“. | ☐ | ☐ | ☐ |
| Das Package „de.tud.cs.se.flashcards.model“,  die Klasse „FlashcardSeriesFilter“,  die Methode „FlashcardSeriesFilter“ (Konstruktor),  die Methode „removeCards“,  und das Feld „searchTerm“ in dieser Reihenfolge gemeinsam selektieren und per Drag and Drop auf eine weiße Fläche des Diagramms ziehen.  Der Name des Ensemble lautet „de.tud.cs.se.flashcards.model.FlashcardSeriesFilter“ und der Inhalt der Query ist „class\_with\_members('de.tud.cs.se.flashcards.model','de.tud.cs.se.flashcards.model.FlashcardSeriesFilter')  or package('de.tud.cs.se.flashcards.model')  or method('de.tud.cs.se.flashcards.model','de.tud.cs.se.flashcards.model.FlashcardSeriesFilter','<init>','void',['de.tud.cs.se.flashcards.model.FlashcardSeries'])  or method('de.tud.cs.se.flashcards.model','de.tud.cs.se.flashcards.model.FlashcardSeriesFilter','removeCards','void',['int'])  or field('de.tud.cs.se.flashcards.model','de.tud.cs.se.flashcards.model.FlashcardSeriesFilter','searchTerm','java.lang.String')“. | ☐ | ☐ | ☐ |
| Die Innere Klasse „de.tud.cs.se.flashcards.model.Flashcard.InnerEvenIterator“ per Drag and Drop auf die weiße Fläche des Diagramms ziehen.  Der Name des Ensemble lautet „de.tud.cs.se.flashcards.model.Flashcard.InnerEvenIterator“ und der Inhalt der Query ist „class\_with\_members('de.tud.cs.se.flashcards.model','de.tud.cs.se.flashcards.model.Flashcard')“. | ☐ | ☐ | ☐ |
| Drag and Drop Operationen aus einer JAR-Datei | | | |
| Die dnsns.jar aus dem „JRE System Library“-Bibliotheken Ordner per Drag and Drop auf die weiße Fläche des Diagramms ziehen. Der Name des Ensembles lautet „dnsns.jar“ und der Inhalt der Query ist „package('sun')  or package('sun.net')  or package('sun.net.spi')  or package('sun.net.spi.nameservice.dns')  or package('sun.net.spi.nameservice')“. | ☐ | ☐ | ☐ |
| Aus dem dnsns.jar das Package „sun.net.spi.nameservice.dns“ per Drag and Drop auf die weiße Fläche des Diagramms ziehen. Der Name des Ensembles lautet „sun.net.spi.nameservice.dns“ und der Inhalt der Query ist „package('sun.net.spi.nameservice.dns')“. | ☐ | ☐ | ☐ |
| Aus JAR-Dateien können alle Typen unterhalb der Packages nicht auf das  Diagramm bzw. ein Ensemble gezogen werden. Es wird das entsprechende  Maus-Icon angezeigt.  Damit sind (Innere)Klassen, Methoden und Felder gemeint. Alle Objekte  innerhalb des roten Rahmens können nicht auf das Diagramm oder eine  Ensemble gezogen werden können (siehe nachfolgendes Bild). | ☐ | ☐ | ☐ |

# Diagramm-Validierung:

|  |  |  |  |
| --- | --- | --- | --- |
| **Prozessbeschreibung** | **Abgenommen** | | |
| 32-BitWin | 64-BitWin | 64-Bit OS X |
| „DerivedTest.sad“ muss den Fehler „Queries of non leaf Ensembles must be derived“ anzeigen. Die Query des zu dem Fehler gehörenden Ensembles muss auf derived gesetzt werden. Danach muss das Diagramm gespeichert werden. Nach dem Speichern darf der Fehler nicht mehr angezeigt werden. | ☐ | ☐ | ☐ |
| „EmptyAndSameNameTest.sad“ muss die Fehler „Ensemble name must be set“ und “Ensemble must have unique name!” enthalten.  Die Namen der Ensembles im Vespucci-Diagramm so anpassen das alle Ensembles einen Namen haben und kein Name doppelt vorkommt. Das so modifiziert Vespucci-Diagramm speichern. Nach dem Speichern darf kein Fehler mehr angezeigt werden. | ☐ | ☐ | ☐ |

# Connection-Assistant-Mechanisms:

|  |  |  |  |
| --- | --- | --- | --- |
| **Prozessbeschreibung** | **Abgenommen** | | |
| 32-BitWin | 64-BitWin | 64-Bit OS X |
| Das Vespucci-Diagramm „Connection-Assistant-Mechanisms.sad“ öffnen. | ☐ | ☐ | ☐ |
| Über die Popup Pfeile am Ensemble „1“ Outgoing dependencies mit dem Ensemble „1“ als Source zu den Target Ensembles „1.1“, „1.2“, „1.3“, „1.4“ anlegen. | ☐ | ☐ | ☐ |
| Über die Popup Pfeile am Ensemble „2“ Outgoing dependencies mit dem Ensemble „2“ als Source zu den Target Ensembles „2.1“, „2.2“, „2.3“, „2.4“ anlegen. |  |  |  |
| Über die Popup Pfeile am Ensemble „1“ Incoming dependencies mit dem Ensemble „1“ als Target von den Source Ensembles „1.1“, „1.2“, „1.3“, „1.4“ anlegen. |  |  |  |
| Über die Popup Pfeile am Ensemble „2“ Incoming dependencies mit dem Ensemble „2“ als Target von den Source Ensembles „2.1“, „2.2“, „2.3“, „2.4“ anlegen. |  |  |  |

# Architektur-Validierung:

|  |  |  |  |
| --- | --- | --- | --- |
| **Prozessbeschreibung** | **Abgenommen** | | |
| 32-BitWin | 64-BitWin | 64-Bit OS X |
| Alle „nichtOk\*.sad“-Diagramme müssen bei der Überprüfung die in den Diagrammen definierten Architekturverstöße melden. | ☐ | ☐ | ☐ |
| Alle „Ok\*.sad“-Diagramme dürfen bei der Architektur-Validierung keinen Fehler erzeugen. | ☐ | ☐ | ☐ |
| Die Architekturdiagramme, die Funktionalität des Vespucci-Plug-Ins modulieren dürfen bei der Architektur-Validierung keine Fehler erzeugen. | ☐ | ☐ | ☐ |
| „DerivedTest.sad“ muss den Fehler „Queries of non leaf Ensembles must be derived“ anzeigen. Die Query des zu dem Fehler gehörenden Ensembles muss auf derived gesetzt werden. Danach muss das Diagramm gespeichert werden. Nach dem Speichern darf der Fehler nicht mehr angezeigt werden. | ☐ | ☐ | ☐ |

# Anmerkungen

|  |
| --- |
|  |

# Abschluss

|  |  |
| --- | --- |
| **Prozessbeschreibung** | **Abgenommen** |
| **Veröffentlichen** |  |
| Erfolgreich überprüften Sourcecode hochgeladen. | ☐ |
| E-Mail an den Auftraggeber. | ☐ |

|  |  |
| --- | --- |
| **Abgenommen von:** | **am:** |
|  |  |
|  |
|  |
|  |