DocumentationTree - DV-Konzept

Jens Kötterheinrich

2018-02-20

Inhalt

1. Allgemeines

1.1. Zweck und Struktur des Dokuments

Dieses Dokument beschreibt die Anwendung DocumentationTree. Die Anwendung wird als Bestandteil von Dokumentation von Datenbanken verwendet. Mit diesem Dokument soll der Leser in die Lage versetzt werden, - den Aufbau der Anwendung zu verstehen, - die Aufgaben der Anwendung, - das System über die Benutzeroberfläche bedienen zu können, - die Schnittstellen zu anderen Systemen zu kennen, - im Falle eines Systemfehlers die geeigneten Maßnahmen ergreifen zu können.

2. Übersicht

Die Anwendung DocumentationTree basiert auf der Java Standard Edition. Es wird mindestens die Version 7 vorausgesetzt. Bei der Anwendung handelt es sich um eine JavaFX-Anwendung. Sie lässt sich mit einer Java Runtime (>= Version 7) starten.

3. Funktionalität

Die Anwendung stellt folgende Funktionalitäten bereit:

- Öffnen und Parsen einer Datei mit LuaTable (später auch XML)
- Darstellung als Baumstruktur
- Markieren von Knoten in unterschiedlichen Farben zur schnellen Übersicht und Analyse

4. Dateibeschreibungen

4.1. LuaTable

Eine Lua-Datei besteht aus Lua-Code. Darin wird eine LuaTable erzeugt und zurückgegeben. Der Aufbau der LuaTable basiert auf IupTree.

Beispiel LuaTable

```
example =
{branchname="example",
  {branchname="foo", 'Hello'},
  {branchname="bar", 'world!'}
}

return example
```

5. Programmdesign

5.1. Ablauf

5.1.1. Auswahl einer Datei

Nach dem Start der Anwendung wird ein Dialog angezeigt, um eine Datei auszuwählen. Die erlaubten Endungen der zu öffnenden Dateien sind *.lua* und *.xml*.

5.1.2. Parsen einer ausgewählten Datei

Mit Hilfe des Frameworks

```
<dependency>
    <groupId>org.luaj</groupId>
    <artifactId>luaj-jse</artifactId>
    <version>3.0.1</version>
</dependency>
```

wird die LuaTable geparst und eine Instanz von `org.luaj.vm2.LuaTable` erzeugt. Das Framework wird mit dem Maven-Plugin *maven-assembly-plugin* mit in das JAR gepackt, sodass nur eine Datei benötigt wird.

5.1.3. Erzeugen eines Baums

Auf Basis der erzeugten Instanz der LuaTable werden TreeItems erzeugt. Dabei werden die Elemente der LuaTable rekursiv ausgewertet. Einzelne Knoten werden als `javafx.scene.control.TreeItem` erzeugt, die zu einer javafx.scene.control.TreeView` hinzugefügt werden. Standardmäßig werden Knoten immer aufgeklappt dargestellt. Durch *state = 'COLLAPSED'* kann für einen Knoten definiert werden, dass er initial nicht ausgeklappt dargestellt wird.

5.1.4. Darstellung

Die Klasse `javafx.scene.control.TreeView` ermöglicht die Darstellung von POJOs. Dafür wird die Klasse `de.beckdev.TextNode` verwendet.

```
public class TextNode {
    private String text;
    private String color;

    public TextNode(String text) {
        this.text = text;
        this.color = "#ffffff";
    }

    public TextNode(String text, String color) {
        this.text = text;
        this.color = color;
    }

    public String getText() {
        return text;
    }

    public void setText(String text) {
        this.text = text;
    }

    public String getColor() {
        return color;
    }

    public void setColor(String color) {
        this.color = color;
    }

    @Override
    public String toString() {
        return text;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        TextNode textNode = (TextNode) o;
        return Objects.equals(text, textNode.text) &&
                Objects.equals(color, textNode.color);
    }

    @Override
    public int hashCode() {
        return Objects.hash(text, color);
    }
}
```

Eine Instanz von `de.beckdev.TextNode` kann in mehreren Instanzen von `javafx.scene.control.TreeItem` verwendet werden; immer dann, wenn der Wert der Property `text` der Klasse `de.beckdev.TextNode` identisch ist. Um Knoten zu färben wird die gewünschte Farbe der Property `color` zugewiesen. Nachdem die Farben aller Knoten geändert wurden, wird die `javafx.scene.control.TreeView` neu aufgebaut.

5.1.5. Knoten selektieren

Ein Knoten kann durch einen Klick selektiert werden. Dabei wird der Knoten blau markiert. Erst wenn zum ersten Mal ein Knoten markiert wurde, wird der Button *Markieren* aktiviert.

5.1.6. Knoten markieren

Wenn ein Knoten ausgewählt wurde, sollen Knoten nach folgenden Regeln gefärbt werden:

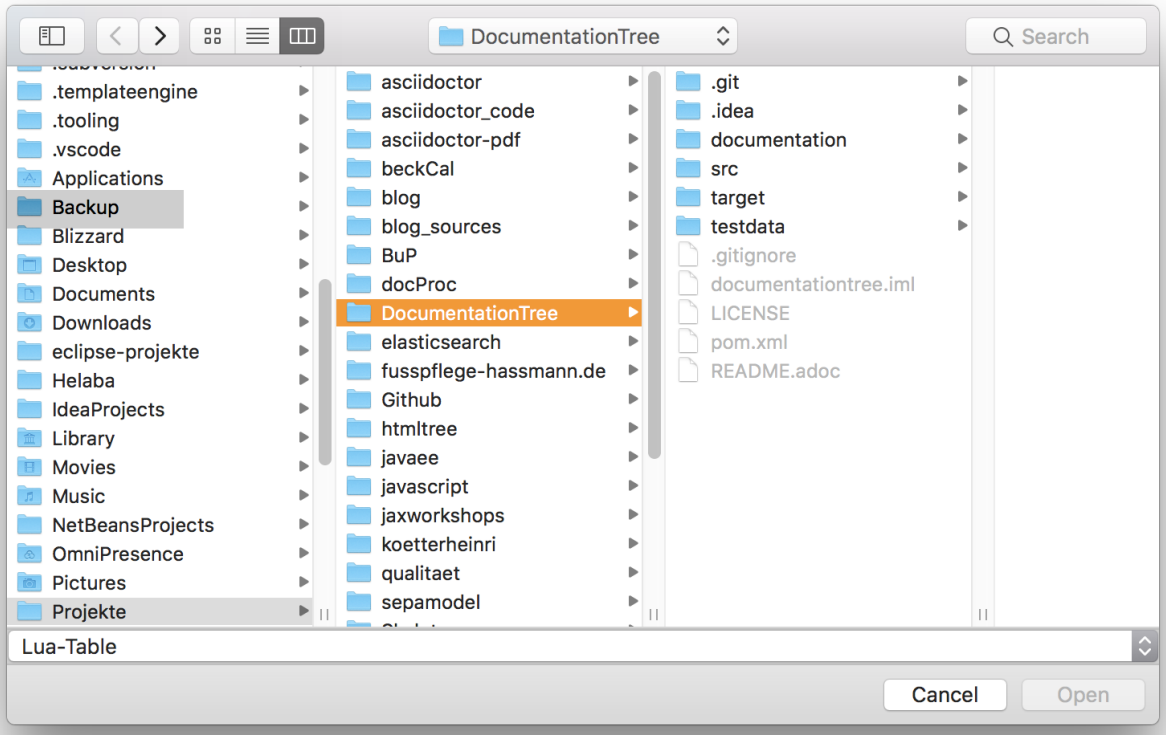
1. Der ausgewählte Knoten wird in blau gefärbt.
2. Knoten, deren Text mit dem des ausgewählten Knoten identisch sind, werden blau gefärbt.
3. Knoten über dem ausgewählten Knoten werden rot gefärbt; Knoten über den Knoten von 2. werden grau gefärbt.
4. Knoten unter dem ausgewählten Knoten werden grün gefärbt.

5.1.7. Fehlerbehandlung

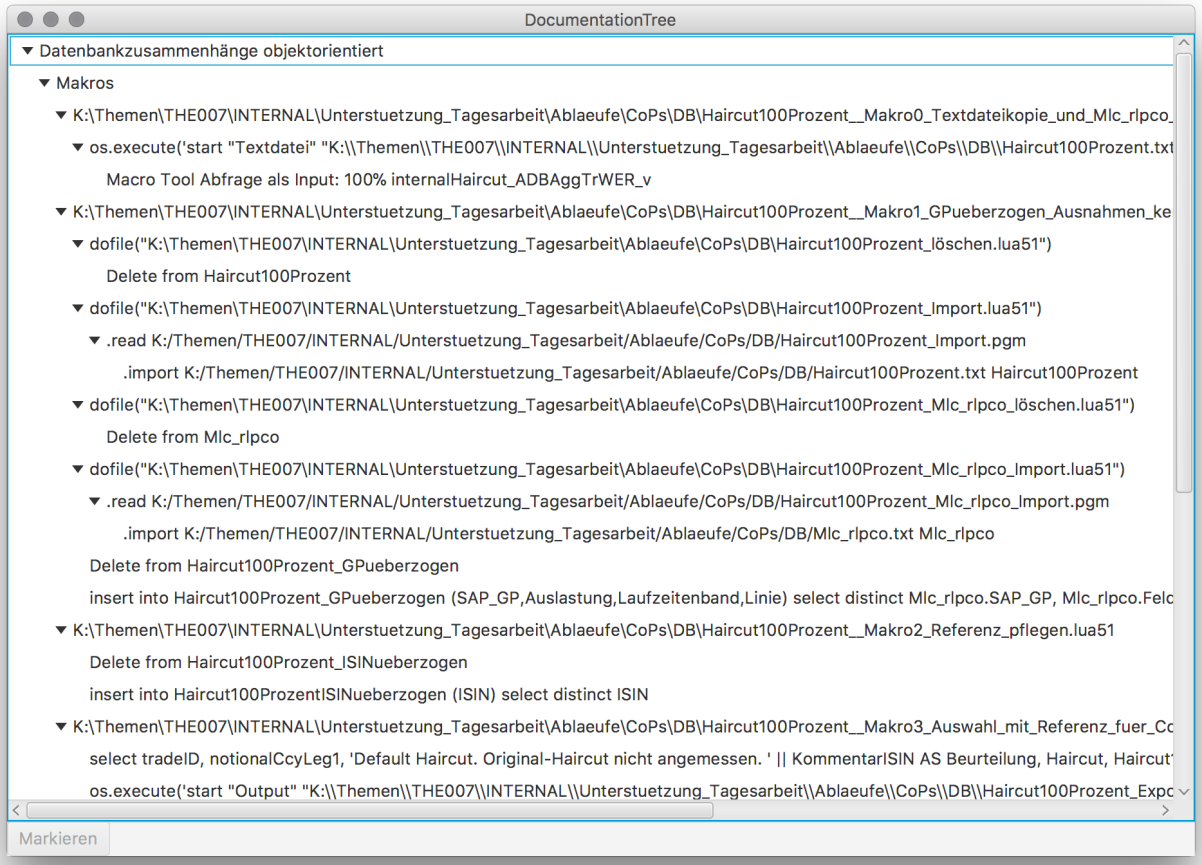
Beim Öffnen einer Datei werden mögliche Fehler abgefangen und in einem Fehlerdialog dargestellt. Durch Bestätigen wird die Anwendung anschließend beendet.

5.2. Benutzeroberfläche

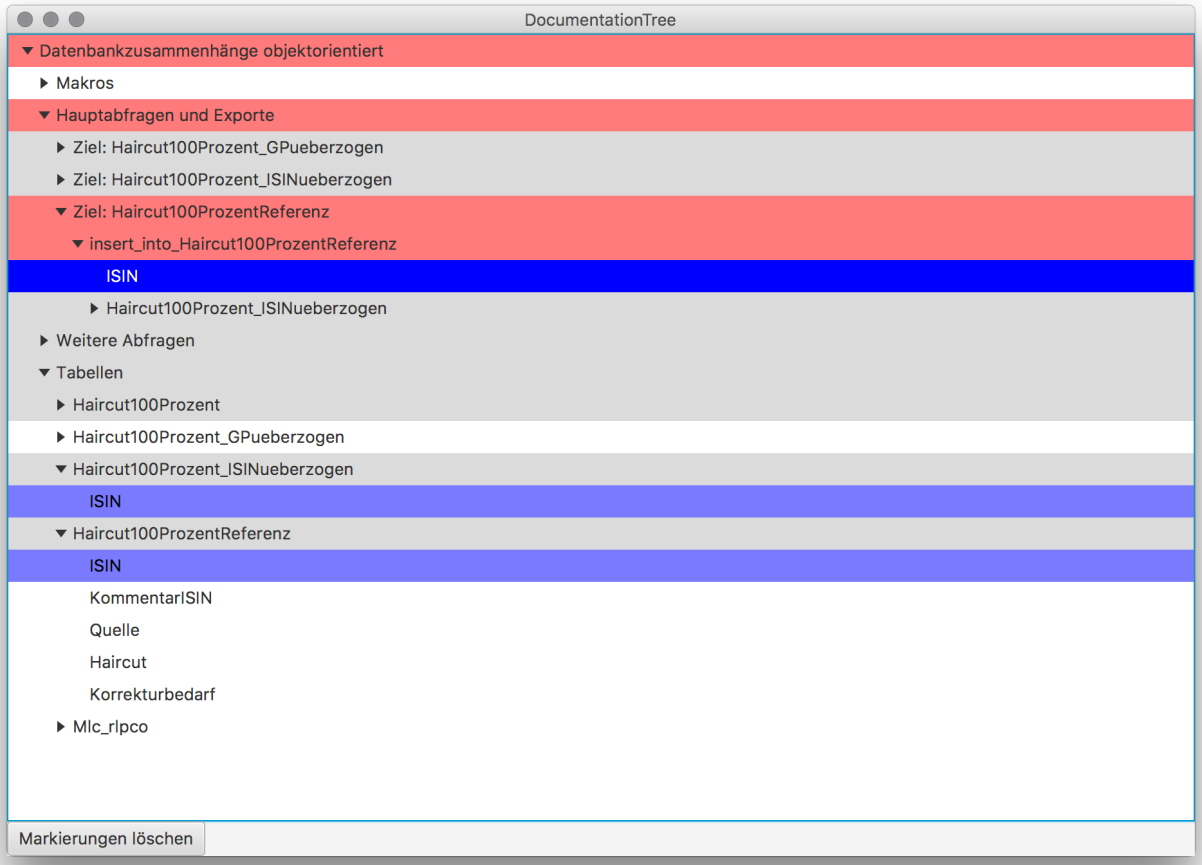
5.3. Start



5.4. Datei als Baum anzeigen

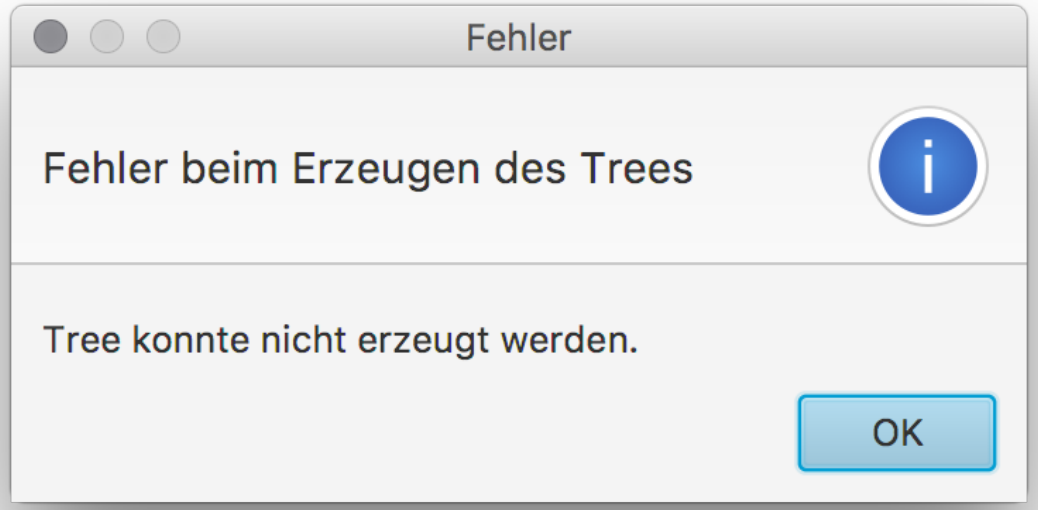


5.5. Gefärbte Knoten



5.6. Fehlermeldung

5.6.1. Keine Datei ausgewählt



5.6.2. Fehler in Datei