KOMPAKT
INFO

JAVA-Generierung aus Entscheidungstabellen mit LF-ET

LOHRFINK
software engineering GmbH & Co. KG

www.lohrfink.de

1 Einleitung

Dieses Dokument bietet einen ersten kompakten Überblick, wie aus einer mit LF-ET erstellten Entscheidungstabelle JAVA-Programmcode generiert werden kann.

Die für das Beispiel gewählte Entscheidungstabelle ist ganz bewusst sehr einfach, damit man sich ganz auf das Thema „Code-Generierung“ konzentrieren kann.

Basierend auf dem aktuellen Zustand der Ampel soll der neue Zustand bestimmt werden:

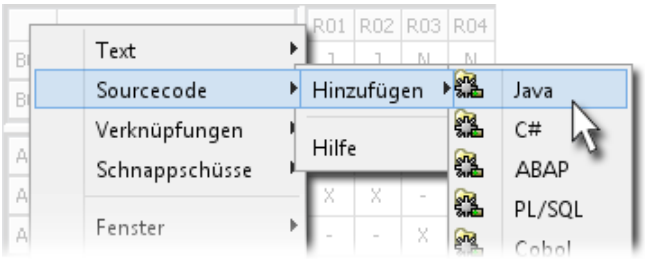
Ampel schalten

		R01	R02	R03	R04
B01	Ist Rot an ?	J	J	N	N
B02	Ist Gelb an ?	J	N	J	N
A01	Neuer Zustand: Rot an	-	X	X	-
A02	Neuer Zustand: Gelb an	-	X	-	X
A03	Neuer Zustand: Grün an	X	-	-	-

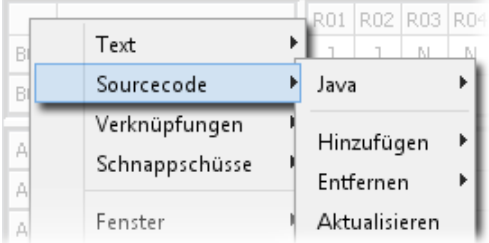
2 JAVA in der Entscheidungstabelle aktivieren

Bei jeder Entscheidungstabelle kann individuell festgelegt werden, für welche Programmiersprachen Programmcode generiert werden soll.

Mit der Funktion ***Sourcecode > Hinzufügen*** wird in der aktuell geöffneten Entscheidungstabelle eine weitere Programmiersprache aktiviert:



Alle aktivierten Programmiersprachen (hier JAVA) erscheinen automatisch unter dem Menüpunkt ***Sourcecode*** und bieten einen direkten Zugriff auf die einzelnen Source-Elemente der Entscheidungstabelle:



3 Einmalige Programmierung der Bedingungen und Aktionen

Damit aus einer Entscheidungstabelle Programmcode generiert werden kann, müssen für JAVA - ***nur ein einziges Mal*** - die folgenden Source-Elemente definiert werden:

- ein ***Prolog*** (Klassen-Definition, Methoden-Signatur, etc.) und
- für jede ***Bedingung*** ein ***logischer Ausdruck*** und
- für jede ***Aktion*** ein (oder mehrere) ***Statement(s)*** und
- ein ***Epilog*** (Methoden-Ende, Klassen-Ende, etc.)

In unserem Beispiel werden die folgenden JAVA-Source-Elemente eingegeben:

	Fachliche Beschreibung	**JAVA-Source-Element**

Prolog		`public class AmpelSchaltung {`
		`  public void schalten(Ampel ampel) {`
		`  boolean rotNeu = false;`
		`  boolean gelbNeu = false;`
		`  boolean gruenNeu = false;`

Bedingung B01	Ist Rot an ?	`  (ampel.isRot())`
Bedingung B02	Ist Gelb an ?	`  (ampel.isGelb())`
Aktion A01	Neuer Zustand: Rot an	`  rotNeu = true;`
Aktion A02	Neuer Zustand: Gelb an	`  gelbNeu = true;`
Aktion A03	Neuer Zustand: Grün an	`  gruenNeu = true;`

Epilog		`  ampel.set(rotNeu, gelbNeu, gruenNeu);`
		`  }`
		`}`

Anmerkungen:

- Die Source-Elemente können (müssen aber nicht) direkt in der Entscheidungstabelle gespeichert werden, als zusätzliche Information zu den jeweiligen Bedingungen und Aktionen.
- Für die Bearbeitung der einzelnen Source-Elemente kann entweder direkt der in LF-ET integrierte Source-Editor, oder aber auch eine beliebige Entwicklungsumgebung, wie z.B. ***Eclipse***, ***NetBeans*** oder ***IntelliJ***, verwendet werden.
- Die Klammern bei den logischen Ausdrücken sind nur eine Stil-Empfehlung.

4 Der generierte Sourcecode

AmpelSchaltung.java

```
// Generated by LF-ET 2.1.4 (140116a)
// From decision table
// "D:/Projekte/LF-ET-Demo/Java/AmpelSchaltung.lfet"
// 11.02.2014 14:28
//
// Prolog Decision Table ---->
public class AmpelSchaltung {

    public void schalten(Ampel ampel) {

        boolean rotNeu = false;
        boolean gelbNeu = false;
        boolean gruenNeu = false;
        // Prolog Decision Table <----

        // Condition B01: Ist Rot an ?
        if (ampel.isRot())
        {
            // Condition B02: Ist Gelb an ?
            if (ampel.isGelb())
            {
                // Rule R01 ---->

                // Action A03: Neuer Zustand: Grün an
                gruenNeu = true;

                // Rule R01 <----
            }
            else
            {
                // Rule R02 ---->

                // Action A01: Neuer Zustand: Rot an
                rotNeu = true;

                // Action A02: Neuer Zustand: Gelb an
                gelbNeu = true;

                // Rule R02 <----
            }
        }
        else
        {
            // Condition B02: Ist Gelb an ?
            if (ampel.isGelb())
            {
                // Rule R03 ---->

                // Action A01: Neuer Zustand: Rot an
                rotNeu = true;

                // Rule R03 <----
            }
            else
            {
                // Rule R04 ---->

                // Action A02: Neuer Zustand: Gelb an
                gelbNeu = true;

                // Rule R04 <----
            }
        }

        // Epilog Decision Table ---->
        ampel.set(rotNeu, gelbNeu, gruenNeu);
    }
}
// Epilog Decision Table <----
// End of generated Java source code
```

Nur ein einziges Mal...

...wurde in der Entscheidungstabelle manuell programmiert:

- der Prolog
- zwei logische Argumente für die beiden Bedingungen
- drei Statements für die drei Aktionen
- und der Epilog

Diese Source-Elemente können vom Generator beliebig oft wiederverwendet werden.

Die geschachtelte Kontrollstruktur (der „IF-ELSE-Regelbaum“) wird bei jeder Generierung aus den Regeln der Entscheidungstabelle komplett neu berechnet und generiert.

Dies unterstützt leistungsstark die Entwickler und garantiert maschinell langfristig die exakte Übereinstimmung von Vorgabe und Programmcode.

Starke Entlastung bei Wartung und Pflege:

- Bei Änderungen an den Regeln - ein ganz typischer Änderungsfall - muss einfach nur neu generiert werden!
- Eine erneute Programmierung ist später nur „punktuell“ notwendig, wenn sich die Bedeutung einzelner Bedingungen oder Aktionen ändert, oder wenn neue Bedingungen oder Aktionen zur Entscheidungstabelle hinzukommen.

5 Anmerkungen zur Generierung

SWITCH-Statements

LF-ET bietet bei Bedarf auch die Möglichkeit, für einzelne Bedingungen anstelle eines IF-Statements ein SWITCH-Statement zu generieren.

Änderungen am generierten Programmcode

„Externe“ Änderungen am generierten Sourcecode, zum Beispiel verursacht durch Refactorings, kann LF-ET erkennen und die betroffenen Source-Elemente in der Entscheidungstabelle automatisch synchronisieren.

Generierung von Interfaces

Optional kann LF-ET, über die Generierung von Interfaces, die Implementierung stärker von den Entscheidungstabellen entkoppeln.

Bei dieser Generierungsvariante findet die Programmierung vollständig in der Java-Entwicklungsumgebung statt und die Entscheidungstabellen enthalten keinerlei Sourcecodes.

6 Weiterführende Themen

Dieses Dokument erläutert nur ganz grob, im Sinne einer ersten Übersicht, die wichtigsten Grundprinzipien und Mechanismen der Programm-Generierung mit LF-ET für JAVA.

Weiterführende Themen wären beispielsweise:

- Komplexe Bedingungen, z.B. mit Unterprogrammaufrufen oder Datenbankzugriffen
- Umsetzung von Schleifen-Verarbeitungen, ggf. auch mehrfach geschachtelt
- Modellierung von Automaten
- Verschachtelung von Entscheidungstabellen, d.h. Entscheidungstabellen als Bedingungen oder Aktionen in anderen Entscheidungstabellen
- Stärkere Entkopplung der Implementierung durch Generierung von Interfaces
- Automatische Protokollierung der Regel-Ausführungen
- Statistische Auswertung der Regel-Ausführungen, auch über längere Zeiträume
- Sichere Ableitung aller notwendigen Testfälle direkt aus den Entscheidungstabellen
- Automatische Ermittlung der Test-Deckungsgrade
- Unterstützung weiterer Sprachen, wie z.B.: C#, ABAP, PL/SQL, Natural, Cobol, VB, VBA, C++, C, Delphi, PHP5,...
- Anpassung der Generierung an spezielle Projekt-Bedürfnisse
- u.v.m.

7 Zusammenfassung

Entscheidungstabellen nach DIN 66241 helfen, fachliche Vorgaben leichter, präziser und vollständiger zu beschreiben. Der Umgang mit Entscheidungstabellen ist leicht erlernbar und LF-ET ermöglicht, mit seiner Excel-ähnlichen Oberfläche und zahlreichen leistungsstarken Funktionen, die leichte Beherrschung auch sehr komplexer Regelwerke.

Durch den Einsatz von Generatoren kann maschinell garantiert werden, dass die Software exakt mit den Vorgaben übereinstimmt. Auch der tatsächlich anfallende Programmieraufwand kann dadurch stark reduziert werden.

Insbesondere bei späterer Wartung und Pflege sind erfahrungsgemäß erhebliche Aufwands-Einsparungen zu erwarten, bei gleichzeitig deutlich reduziertem Fehler-Risiko.

Bei Bedarf kann LF-ET aus einer Entscheidungstabelle auch Programmcode für mehrere Programmiersprachen erzeugen.

LF-ET ist ein reines Entwicklungswerkzeug, zur Ausführung der generierten Programme ist ***keinerlei zusätzliche Software erforderlich***.

8 Kontakt

Als IT-Beratungsunternehmen und -Dienstleister unterstützen wir seit vielen Jahren zahlreiche Behörden und Unternehmen aus den unterschiedlichsten Branchen.

Unabhängig vom technischen Umfeld ist dabei die Entscheidungstabelle ein wesentliches Hilfsmittel, um leichter und präziser Prozessabläufe beschreiben, implementieren und testen zu können.

Gerne stehen wir für weitere Informationen zur Verfügung.

LOHRFINK
software engineering GmbH & Co. KG

Marie-Curie-Str. 6
D-70736 Fellbach

Telefon 0711/3424 897-0
Telefax 0711/3424 897-15
info@lohrfink.de
www.lohrfink.de
www.lohrfink.de/lf-et