Mit 'Babysteps' in die XTEXT-XTEND Welt

Teil 7: LUECKEN(X)TEXT, eine DSL mit dynamischer Anbindung

Anbei noch einmal die Code Listings wie sie der Editor zeigt, also mit Syntax Highlites.
Passend zum Artikel im Eclipse Magazin.

von Ulrich Merkel

Lxt0: erste Unterstützungen für den Nutzer, alles noch hart codiert

Der, der uns die verfügbaren Dateien anzeigt: Lxt0ProposalProvider.xtend

```
/*
 * generated by Xtext: Lxt0ProposalProvider.xtend
 */
package de.ulrichmerkel.lueckenxtext.lxt0.ui.contentassist

import java.io.File
import org.eclipse.emf.ecore.EObject
import org.eclipse.xtext.Assignment
import org.eclipse.xtext.RuleCall
import org.eclipse.xtext.ui.editor.contentassist.ContentAssistContext
import org.eclipse.xtext.ui.editor.contentassist.ICompletionProposalAcceptor

class Lxt0ProposalProvider extends AbstractLxt0ProposalProvider {
override completeMergefile_Name(EObject model, Assignment assignment,
        ContentAssistContext context, ICompletionProposalAcceptor acceptor) {
        completeRuleCall(((assignment.getTerminal()as RuleCall)), context, acceptor);
        addFilesFromDir("..\\!lueckenxtext_home","txt",context,acceptor)
}
def addFilesFromDir(String dirPath, String fileExtension,
        ContentAssistContext context, ICompletionProposalAcceptor acceptor) {
        for(File x: new File(dirPath).listFiles){
                if (x.isFile && x.getName.endsWith("."+ fileExtension)) {
                        var xnoext = x.getName.substring(0,x.getName.indexOf("."))
                        acceptor.accept(createCompletionProposal(xnoext , context))
                }
        }
}
}
```

Der, der die Angabe für uns "vorschreibt": Lxt0QuickfixProvider.xtend

```
/*
 * generated by Xtext: Lxt0QuickfixProvider.xtend
 */
package de.ulrichmerkel.lueckenxtext.lxt0.ui.quickfix

import de.ulrichmerkel.lueckenxtext.lxt0.validation.Lxt0Validator
import java.io.File
import java.nio.charset.StandardCharsets
import java.util.regex.Matcher
import java.util.regex.Pattern
import org.eclipse.xtext.ui.editor.quickfix.DefaultQuickfixProvider
import org.eclipse.xtext.ui.editor.quickfix.Fix
import org.eclipse.xtext.ui.editor.quickfix.IssueResolutionAcceptor
import org.eclipse.xtext.validation.Issue

import static com.google.common.io.Files.*

class Lxt0QuickfixProvider extends DefaultQuickfixProvider {

@Fix(Lxt0Validator::DEF_MERGEFILE_PLACEHOLDERS)
def addMergefileValues(Issue issue, IssueResolutionAcceptor acceptor) {
        acceptor.accept(issue, 'variables set empty', 'Generate variable set.', '')
        [ context |
                val xtextDocument = context.xtextDocument
                var replaceText = " placeholders=mergefilename=" + '"' +
issue.data.get(0) + '"'
                var mergefilepath = "..\\!lueckenxtext_home\\" + issue.data.get(0) +
"." + "txt"
                replaceText += makeDefaultPlaceholderGroup(mergefilepath)
```

```
                xtextDocument.replace(issue.offset + issue.getLength, 0, replaceText)
        ]
}

def String makeDefaultPlaceholderGroup(String mergefilepath) {
        if (mergefilepath == null) return ""// nothing to check
        var allPlaceholders = <String>newHashSet()
        var Pattern pattern = Pattern.compile("\\$\\{([^}]*)\\}");
        val File incfile = new File(mergefilepath)
        if (incfile.exists) {
                var Matcher matcher = pattern.matcher(toString(incfile,
StandardCharsets.UTF_8));
                while (matcher.find()) {
                        allPlaceholders.add(matcher.group(1))
                }
        }
// building the default DSL text for the placeholder group
        var returnText = System.getProperty("line.separator")
        for (String placeholder : allPlaceholders.sort)
        {
                returnText += makeDefaultPlaceholderPair(placeholder)
        }
        return (returnText)
}

def String makeDefaultPlaceholderPair(String placeholder) {
        var returnText = ""
        if (placeholder != null) {
                returnText += "," + placeholder + '=""' +
System.getProperty("line.separator")
        }
        return (returnText)
}

}
```

Der, der den Fehlermarker erzeugt: Lxt0Validator.xtend

```
/*
 * generated by Xtext: Ltx0Validator.xtend
 */
package de.ulrichmerkel.lueckenxtext.lxt0.validation

import de.ulrichmerkel.lueckenxtext.lxt0.lxt0.Lxt0Package
import de.ulrichmerkel.lueckenxtext.lxt0.lxt0.Mergefile
import org.eclipse.xtext.validation.Check

class Lxt0Validator extends AbstractLxt0Validator {
public static val DEF_MERGEFILE_PLACEHOLDERS = 'noValuesSet'
        @Check
def checkMergefileVariablesShouldNotBeEmpty(Mergefile mergefile) {
        if(mergefile.placeholderPairs.size < 1) {
                error('values set should not be empty',
                                Lxt0Package.Literals.MERGEFILE__PLACEHOLDER_PAIRS,
                                Lxt0Validator.DEF_MERGEFILE_PLACEHOLDERS,
                                mergefile.name)
        }
}

}
```

Lxtstart: Erste Routinen für Vorlagedateien, aber keine Hilfen für DSL Nutzer

Eine kleine Vorlagendatei ("pattern1.txt") sieht so aus:

```
// pattern1.txt this is some Pseudocode pattern
// which is found multiple times in the sourcecode
function ${functionname}
variable searchprofile = ${searchprofile}
read ${tablename} via ${primarykeyname} using searchprofile
if $returnedhits < 1
raiseMessage("from ${functionname}: no hits in ${tablename} for
${searchprofile} on ${primarykeyname}")
endif
end function ${functionname}
```

Natürlich können auch Nicht-Quellcodes genutzt werden, wie "begruessung_b.txt" zeigt:

```
Sehr geehrt${geehrt_anredeformel} ${kundenname}
```

```
Wir freuen uns, Sie als unseren neuen Kunden begrüßen zu dürfen.

Für Fragen und Anregungen steht Ihnen jederzeit
Ihr persönlicher Ansprechpartner zur Verfügung:

${kontaktdaten_ansprechpartner}

Da es uns daran gelegen ist, Ihnen den bestmöglichen
Service zu bieten, können Sie bei Schwierigkeiten
auch die Geschäftsleitung unter der Durchwahl -555 erreichen.

Mit freundlichen Grüßen,
Ihre Nirwana Factory Un-Limited
```

```
// Grammar file: Lxtstart.xtext
grammar de.ulrichmerkel.lueckenxtext.lxtstart.Lxtstart
with org.eclipse.xtext.common.Terminals
generate lxtstart
"http://www.ulrichmerkel.de/lueckenxtext/lxtstart/Lxtstart"

Model: modules+=Module+;

Module:       (Mergefile);
Mergefile:    'mergefile' name=ID
       'placeholders=' placeholderPairs+=KeyValuePair
       (',' placeholderPairs+=KeyValuePair)* ;
KeyValuePair:name=ID '=' value=STRING;
```

```
/*
 * generated by Xtext: LxtstartGenerator.xtend
 */
package de.ulrichmerkel.lueckenxtext.lxtstart.generator

import de.ulrichmerkel.lueckenxtext.lxtstart.lxtstart.KeyValuePair
import de.ulrichmerkel.lueckenxtext.lxtstart.lxtstart.Mergefile
import de.ulrichmerkel.lueckenxtext.lxtstart.lxtstart.Model
import de.ulrichmerkel.lueckenxtext.lxtstart.lxtstart.Module
import java.io.File
import java.util.HashMap
import java.util.regex.Matcher
import java.util.regex.Pattern
import org.eclipse.emf.common.util.EList
import org.eclipse.emf.ecore.resource.Resource
import org.eclipse.xtext.generator.IFileSystemAccess
import org.eclipse.xtext.generator.IGenerator

import java.nio.charset.StandardCharsets
import static com.google.common.io.Files.*

class LxtstartGenerator implements IGenerator {

override void doGenerate(Resource resource, IFileSystemAccess fsa) {
       var String scriptFileName =
resource.getURI().lastSegment.toString();
       var String outFileBase = scriptFileName.substring(0,
scriptFileName.lastIndexOf("."));
       fsa.generateFile(outFileBase + '.Start.CODE',
makeTheCode(resource.contents.head as Model))
}

def makeTheCode(Model sm) '''
«FOR Module modul : sm.modules»
«switch modul {
              Mergefile: {addMergeFile(modul)}
       }»
«ENDFOR»
'''
```

```
def addMergeFile(Mergefile mf) '''
	«replaceMergeVariables(mf.placeholderPairs,
addIncludeFile(mf.name).toString, mf.name + ".txt")»
'''

def addIncludeFile(String filename) '''
«val File incfile = new File("..\\!lueckenxtext_home\\" + filename +
".txt")»
«IF incfile.exists()»
«toString(incfile,  StandardCharsets.UTF_8)»
«ELSE»
«System.out.println("Error: addIncludeFile: >" + filename + ".txt< not
found")»
«ENDIF»
'''

def replaceMergeVariables(EList<KeyValuePair> placeholderPairs, String
mergeText, String fileID) {
	var mergeTextWork = mergeText;
	if (mergeText == null) { return "" }
	for (KeyValuePair kvp : placeholderPairs) {
		mergeTextWork = mergeTextWork.replaceAll("\\$\\{" + kvp.name
+ "\\}", kvp.value);
	}
	checkForPlaceholderNames(mergeTextWork, fileID)
	return mergeTextWork
}

def checkForPlaceholderNames(String workText, String fileID) {
	var HashMap<String, String> hashNames = new HashMap()
	if (workText.contains("${")) {
		var Pattern pattern = Pattern.compile("\\$\\{([^}]*)\\}");
		var Matcher matcher = pattern.matcher(workText);
		while (matcher.find()) {
			hashNames.put(matcher.group(1), "")
		}
		System.out.println("Warning1: " + hashNames.size
			+ " unresolved placeholders in "
			+ fileID + ": " + hashNames.keySet.join(", ")
		)
	}
	return ""
}

}

// #### Ausschnitt aus dem Generator (*.XTEND) ####
import java.awt.Toolkit
import java.awt.datatransfer.Clipboard
import java.awt.datatransfer.StringSelection


def setClipboard(String workstring) {
	try {
		var Toolkit toolkit = Toolkit.getDefaultToolkit();
		var Clipboard clipboard = toolkit.getSystemClipboard();
		var StringSelection strSel = new StringSelection(workstring);
		clipboard.setContents(strSel, null);
	} catch (Exception e) {
		e.printStackTrace();
	}
	return workstring
}

// **** Möglichkeit, im Dialog Meldungen angeben zu können
def showAlert(String theMessage, String title) {
	JOptionPane.showMessageDialog(null, theMessage, title
		, JOptionPane.INFORMATION_MESSAGE);
	return ""
}
```

```
<snip>

def makeTheCodeAutocopy(Model sm) '''
«setClipboard(makeTheCode(sm).toString)»
'''
```

Babysteps zum Mitmachen, jetzt unter LUNA

Da wir jetzt eher allgemeine Themen bearbeiten, kann auch mit experimentiert werden. Dazu laden wir uns von www.uli-merkel.de/uli-xtext die Zip-in-Zip Datei "!lueckenxtext_luna-part1-all-in-one.zip".

Grundlage ist die "Eclipse IDE for Java and DSL Developers" von LUNA, die wir von [1] herunterladen können. Für meine Experimente habe ich ein Verzeichnis "C:\!lueckenxtext_luna" angelegt, in das ich das Eclipse Archiv entpacke (das Ausrufezeichen am Anfang bringt das Verzeichnis an den Anfang des Explorers, eine meiner Eigenheiten die sich im Laufe der Jahre angesammelt haben).

Ebenso wird dahin die !lueckenxtext_luna-initial-data.zip entpackt, die uns das Verzeichnis für unsere Vorlagen, "!lueckenxtext_home" samt einiger Beispieldateien liefert. Die Batch-Datei "!!start_eclipse.bat" ist etwas Komfort beim Start der Eclipse Umgebung:

```
cd eclipse
start eclipse.exe
```

Beim Starten von Eclipse wird nach dem Pfad zu einem Workspace gefragt, ich gebe hier immer einen relativen Pfad an, also:

```
..\WS-lueckenxtext-part1
```

Wir sollten die Preferences so setzen, dass die Dateien immer als UTF-8 erstellt werden; ferner habe ich den "Spelling" Modus für Text Editoren deaktiviert.

Jetzt können wir die vorbereiteten Projekte importieren über das Menü:

```
File => Import => General => Existing Projects into Workspace
```

Im darauf folgenden Bildschirm geben wir den Pfad zu unserem Archiv export-lueckenxtext-part1-1408061200.zip an und wählen alle enthaltenen Projekte aus.

UMerkel_Babysteps7_10.JPG

Angabe des Archivs zum Laden der Entwicklungsprojekte

Nach dem Drücken von "Finish" werden unsere Projekte in den Workspace geladen.

UMerkel_Babysteps7_11.JPG

Unsere Entwicklungsprojekte in der neuen Eclipse Umgebung

Als Erstes schauen wir uns die enthaltenen Run Configuration mit dem Namen " Launch RuntimeLueckenXtext_part1" an, die wir eventuell abändern.

```
Run => Run Configurations ...
```

In unserer Abbildung werden wir das Verzeichnis " ../runtime-lueckenxtext_part1_new" nutzen.

UMerkel_Babysteps7_12.JPG

Run Configuration modifizieren

Wenn wir jetzt "Run Configuration" aufrufen, wird dieser Workspace angelegt und wir kommen in ein neues Eclipse. Hier importieren wir die Testprojekte aus unserem Archiv export-runtime-lueckenxtext-part1-1408061440.zip.

Geschafft!! Unser Experimentalbaukasten samt Testumgebung ist fertig und wir können unsere eigenen Erfahrungen machen.

Ende

BILDÜBERSICHT

test_start1.Start.CODE

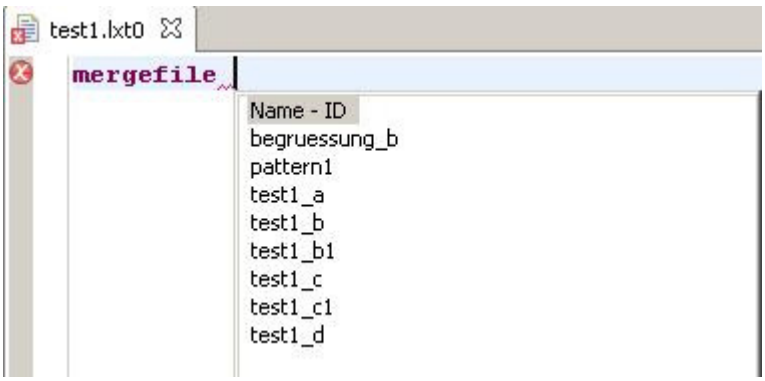
```
// pattern1.txt this is some Pseudocode pattern
// which is found multiple times in the sourcecode
function myFunction
variable searchprofile = Peter*
read Customer via PK01 using searchprofile
if $returnedhits < 1
raiseMessage("from myFunction: no hits in Customer for Peter* on PK01")
endif
end function myFunction
// pattern1.txt this is some Pseudocode pattern
// which is found multiple times in the sourcecode
function myOtherFunction
variable searchprofile = Peter*
read Customer via PK02 using searchprofile
if $returnedhits < 1
raiseMessage("from myOtherFunction: no hits in Customer for Peter* on PK02")
endif
end function myOtherFunction
```

test_start1.lxtstart

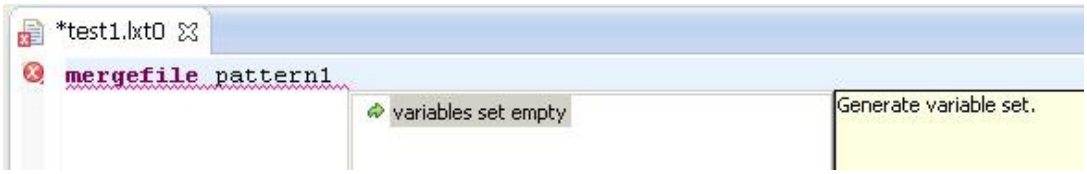
```
mergefile pattern1 placeholders=functionname="myFunction"
,primarykeyname="PK01"
,searchprofile="Peter*"
,tablename="Customer"

mergefile pattern1 placeholders=functionname="myOtherFunction"
,primarykeyname="PK02"
,searchprofile="Peter*"
,tablename="Customer"
```

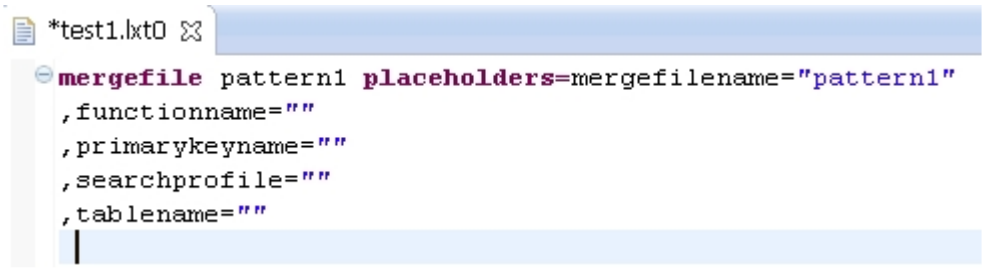
UMerkel_Babysteps7_1.JPG



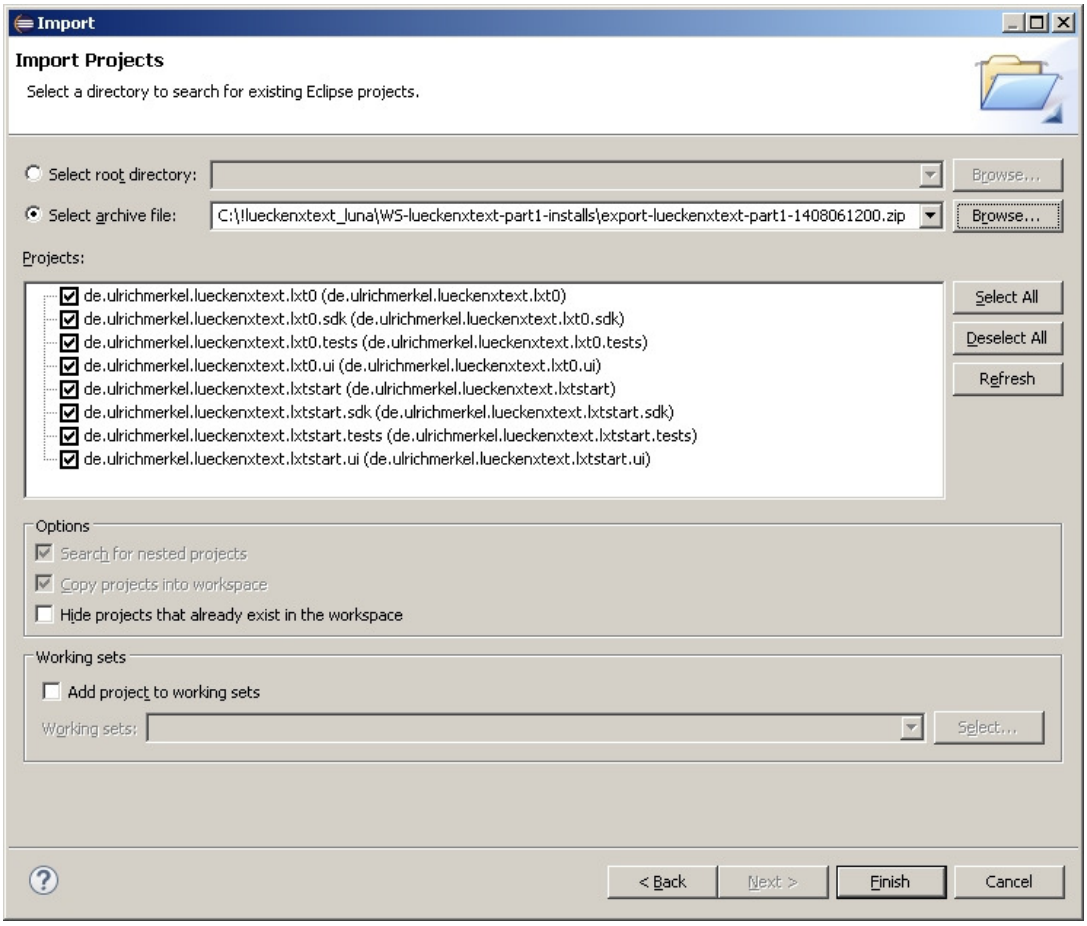
UMerkel_Babysteps7_2.JPG



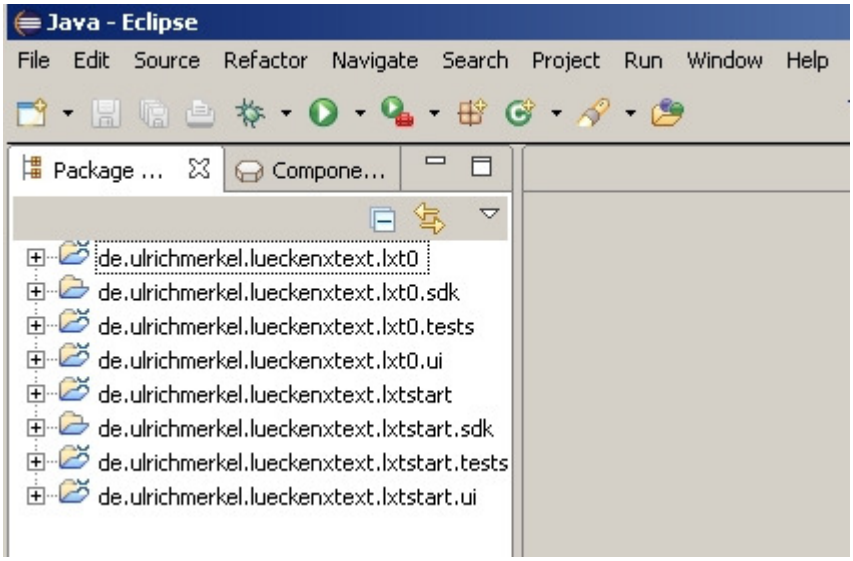
UMerkel_Babysteps7_3.JPG



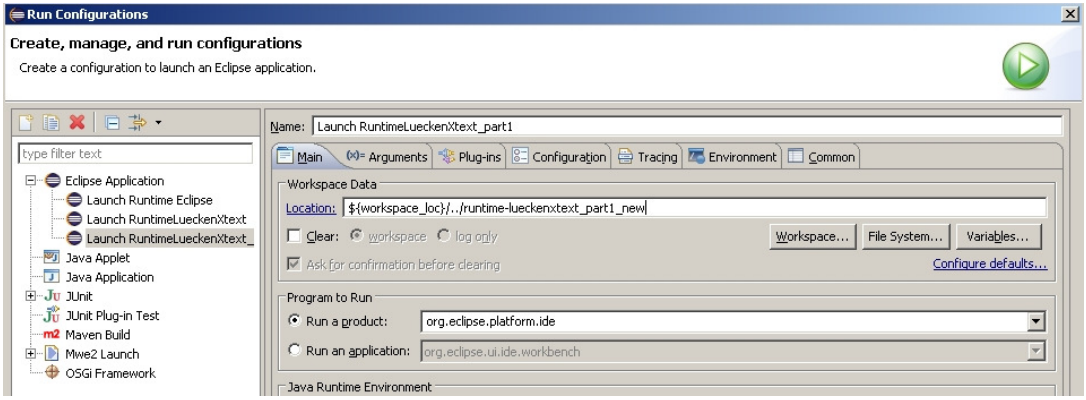
UMerkel_Babysteps7_4.JPG



UMerkel_Babysteps7_10.JPG



UMerkel_Babysteps7_11.JPG



UMerkel_Babysteps7_12.JPG