

Invoca Systems

Application Development Framework

Systemdokumentation

Vers. 1.2.2 / 08.02.2005

Konzepte:

- **Architektur** (Backend – Applicationserver – Client)
- Java-Database **Objectrelational Mapping** (=> Doku)
- Java **Object Browser** Application Framework (=> Doku)
- **Database Browser** (=> siehe Build-Prozess, DBMetaData)
- **ClassMetaDataBuilder** (=> Doku: *Tag-basierte Funktionen*)
- **Multilingual** Support (=> siehe Build-Prozess: LanguageBuilder)
- **ClassmanagerFramework** (=> siehe Build-Prozess)
- Komponentenmodell / **Repository** (siehe folgende Seite)
- Database **Sync Tool** (Bsp: OS-Commerce)
- **Trigger-Implementation** (Bsp: TNT-Modul, Adress-Eingabe)

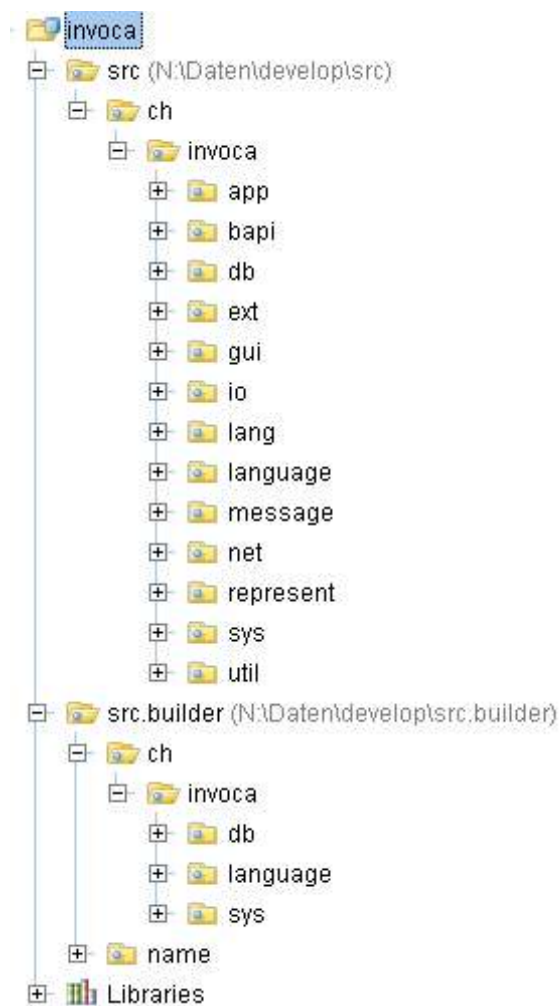
Tools:

- Java Database Migration Framework „**DatabaseDuplicator**“ (=> Doku)
- **JavaElementNameBuilder** (=> siehe Build-Prozess)
- **UnitTester** (=> siehe Build-Prozess)
- **Code-Builder** (ch.invoca.sys.codebuilder.*)

Themen:

GUI {Bilder und Dokumente in der Datenbank: **X_DATA**, **Shared-Objects**, **Undo/Redo**, **Tags-Funktionsdefinition/Action**, **Drag and Drop**, Dia-Viewer, **Bild-Integration**} **Class-Filter**, **Cache-Freeze**: Chained Elements Cache, **PropertyHandler**, **MessageGateway**

Komponentenmodell / Repository



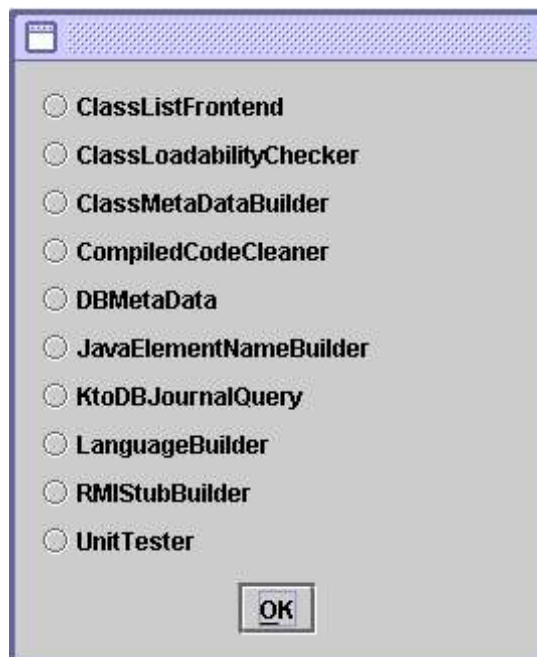
Angaben über den Projekt-Quellcode:

src:	5,2 MB	1695 Dateien	in 181 Packages
src.builder	35,8MB	3897 Dateien	in 225 Packages (t.w generierter Code)
Total	41MB	5592 Dateien	in 406 Packages












Class-Files kompiliert: **27,5MB** in 4264 Dateien

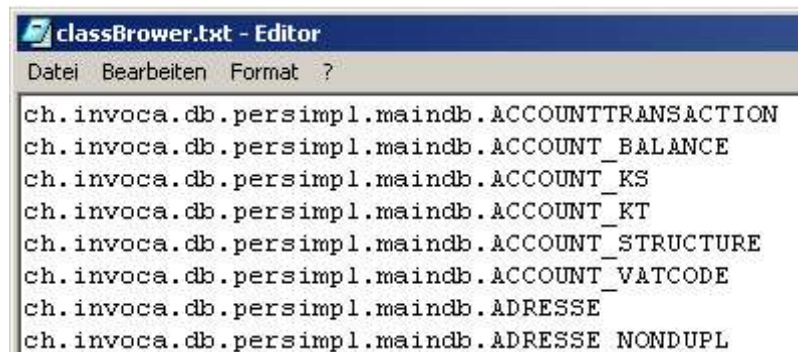
Der Build-Prozess:

Ein einfaches Tool unterstützt den **interaktiven, inkrementell** ausführbaren Build-Prozess:



ClassListFrontent **Selektiert Java-Klassen aus dem gesamten Projekt** gem. frei definierbaren Filtern. Die **Filter** können beliebig verknüpft und verschachtelt sein. Speichert das Resultat in **Textdateien** als Cache.

 classBrower.txt	10 KB
 classMetaDataBuilder.txt	65 KB
 javaElementNameBuilder.txt	142 KB
 pack_ch.invoca.bapi.delivery.tnt.item.txt	1 KB
 pack_ch.invoca.bapi.shopsync.items.txt	1 KB
 printerClasses.txt	1 KB
 rmiStubClasses.txt	2 KB
 supplierItemManagerClasses.txt	1 KB
 systemBuilder.txt	1 KB
 testDBSyncClasses.txt	1 KB
 unitTestRegistry.txt	1 KB



```
classBrower.txt - Editor
Datei  Bearbeiten  Format  ?
ch.invoca.db.persimpl.maindb.ACCOUNTTRANSACTION
ch.invoca.db.persimpl.maindb.ACCOUNT_BALANCE
ch.invoca.db.persimpl.maindb.ACCOUNT_KS
ch.invoca.db.persimpl.maindb.ACCOUNT_KT
ch.invoca.db.persimpl.maindb.ACCOUNT_STRUCTURE
ch.invoca.db.persimpl.maindb.ACCOUNT_VATCODE
ch.invoca.db.persimpl.maindb.ADRESSE
ch.invoca.db.persimpl.maindb.ADRESSE NONDUPL
```

ClassLoadabilityChecker Prüft alle Klassen des Projekts ob sie geladen werden kann (=> statischer Code, der beim Laden ausgeführt wird).

ClassMetaDataBuilder Konvertierung der MetaInfoProvider (Java Source Files) => nach XML

[Dependencies: DBMetaData, ClassListFrontend]

```

/**
 * @displayinfo de Adress-Suche
 * @shortinfo de Sucht Adresse nach Email, Telnr oder AdressNr, Name etc.
 *                Es können auch mehrere Angaben gemacht werden.
 * @menu address
 * @command a?
 * @icon addressSearch
 * @userlevel standard
 */
public static ADRESSE chooseByTerm(BigDecimal mandatGrpNr,
                                   ExtendedInteraction interact) throws Exception {
    return chooseByTerm(mandatGrpNr, interact, true, true);
}

```

Dateiname	Größe
ch.invoca.db.persimpl.maindb.ADRESSE\$LetterRepr.xml	9 KB
ch.invoca.db.persimpl.maindb.ADRESSE.xml	171 KB
ch.invoca.db.persimpl.maindb.ADRESSE_NONDUPL.xml	4 KB

```

- <void property="name">
  <string>chooseByTerm</string>
</void>
- <void property="paramMetaInfos">
  - <array class="ch.invoca.app.common.comp.objectbrowser.clsinfo.ParamMetaInfo" length="2">
    + <void index="0">
      + <void index="1">
        </array>
    </void>
  - <void property="signature">
    <string>(java.math.BigDecimal, ch.invoca.app.common.callback.interactionapi.ExtendedInteraction)</string>
  </void>
- <void property="tagsInfos">
  - <array class="ch.invoca.app.common.comp.objectbrowser.clsinfo.tag.TagsInfo" length="6">
    - <void index="0">
      - <object class="ch.invoca.app.common.comp.objectbrowser.clsinfo.tag.GenericTagsInfo">
        - <void property="tagName">
          <string>menu</string>
        </void>
        - <void property="text">
          <string>address</string>
        </void>
      </object>
    </void>
    - <void index="1">
      - <object class="ch.invoca.app.common.comp.objectbrowser.clsinfo.tag.GenericTagsInfo">
        - <void property="tagName">
          <string>command</string>
        </void>
        - <void property="text">
          <string>a?</string>
        </void>
      </object>
    </void>
    - <void index="2">
      - <object class="ch.invoca.app.common.comp.objectbrowser.clsinfo.tag.LanguageTagsInfo">
        - <void property="mainTagName">
          <string>shortinfo</string>
        </void>
        - <void property="tagName">

```

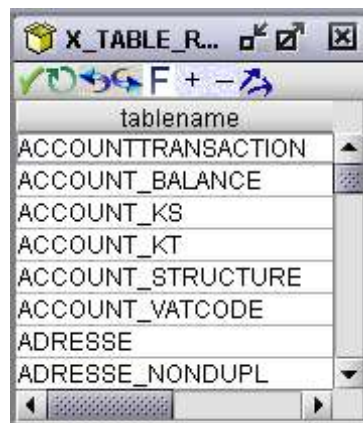
CompiledCodeCleaner

Sucht in den kompilierten Klassen nach solchen, zu dessen kein Quellcode mehr existiert (die Entwicklungsumgebung IntelliJ 4.5 unterstützt diese Funktion)

DBMetaData

Analyse der MetaDaten der Datenbank: Analysiert alle Tabellen, welche in der Tabelle **X_TABLE_READ** gelistet sind und integriert allf. Kommentare (multilingual) zu Tabellen (**X_DOC_TABLE**) und Spalten (**X_DOC_COLUMN**). Erzeugt aus den Daten JavaSourceCode (=> ObjectRelational Mapping)

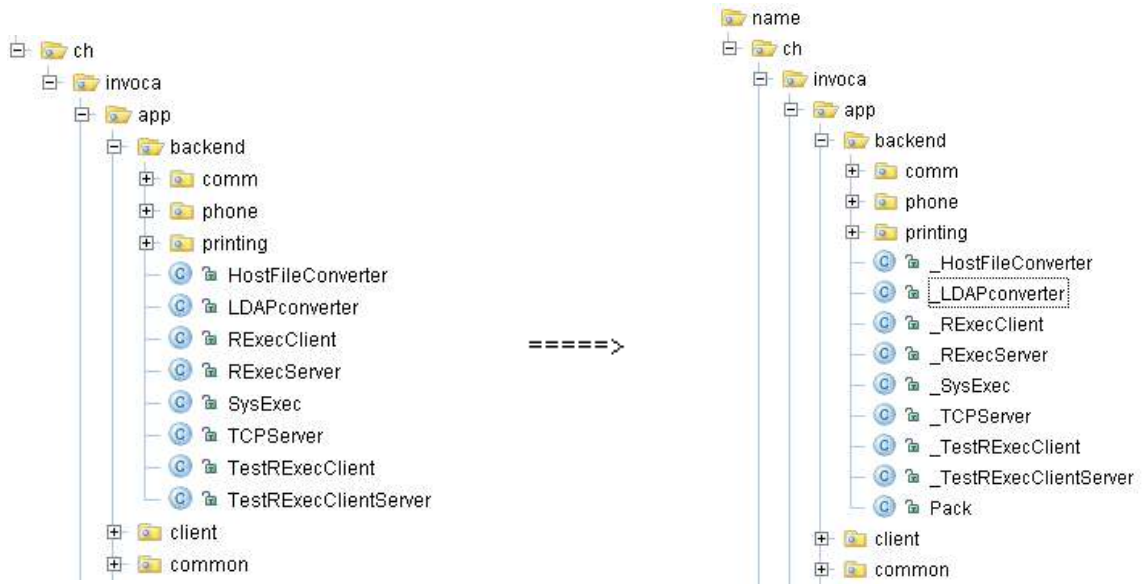
[Dependencies: ClassListFrontend] => Java Compiler



X_DOC_COLUMN : Tabelle							
TABLERNAME	COLUMNNAME	INFOTA	LAN	COU	VA	LANG_ITEM	
ARTIKEL	TEXT	dislengt	--	--	-	20	
AUFTRAG	ADRESSNR	display	de	CH	-	AdressNr	
AUFTRAG	ADRESSNR	short	de	CH	-	Adress-Nummer des Auftraggebers. Falls keine andere Adresse angegeben, entspricht dies auch der Liefer- und Rechnungsadresse.	
AUFTRAG	AUFTRAGSD ATUM	display	de	CH	-	Datum	
AUFTRAG	AUFTRAGSD ATUM	short	de	CH	-	Auftragsdatum: wann wurde der Auftrag erteilt (wann erfolgte die Bestellung)	

JavaElementNameBuilder Abbildung aller Projektklassen sowie derer Elemente (Felder und Methoden) in andere JavaKlassen. Dies ermöglicht, Klassen-, Feld- und Methoden-Namen im SourceCode zu referenzieren.

[Dependencies: ClassMetaDataBuilder] => Java Compiler



```

ADRESSE.java | _LDAPconverter.java | ARTIKEL.java
package name.ch.invoca.app.backend;

public class _LDAPconverter {

    // -----

    public static final String class_name = "ch.invoca.app.backend.LDAPconverter";

    public static final String method_main_java$Dlang$Dstring$A = "main(java.lang.String[])";

    public static final String method_convert_ch$Dinvoca$Dapp$Dcommon$Dcallback$Dinteractionap.

} // end of class _LDAPconverter

```

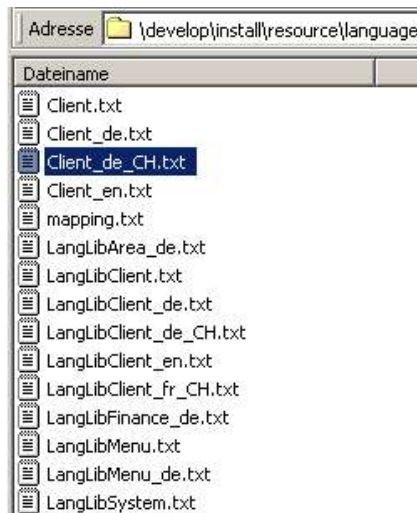
KtoDBJournalQuery Legt Tabelle META_TRANSAKTION an, welche für Journal-Auszüge verwendet wird

LanguageBuilder

Liest **sprachabhängige Inhalte** aus den Tabellen

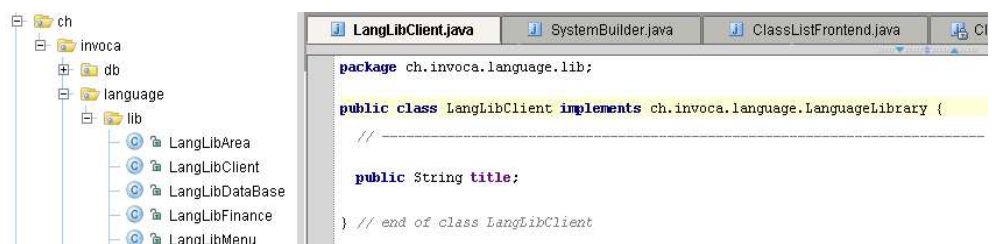
X_LANGUAGE_LIB, X_LANGUAGE_ITEM sowie
X_LANGUAGE_ITEM_TR aus und erzeugt **Java-Klassen**,
dessen Felder Platzhalter für die zur Laufzeit vom Client
gewünschte Übersetzung sind. Die Textdaten werden aus
Performance-Gründen in Sprachfiles extrahiert (schneller
Cache). => Java Compiler

X_LANGUAGE_ITEM_TR : Tabelle						
LANGLIB	LANG_ITEM	LANGCODE	COUNTRYCODE	VARIANT	LANG_ITEM	
area	finance	de	--	-	Finanz	
cli	title	--	--	-	Finalogis Application	
cli	title	de	--	-	Finalogis Application - German Version	
cli	title	de	CH	-	Finalogis Application - (Swiss-Swiss) German Version	
cli	title	en	--	-	Finalogis Application - English Version	
cli	title	fr	CH	-	Finalogis Application - (Swiss-Swiss) Francais Version	



```
Client_de_CH.txt - Editor
Datei Bearbeiten Format ?

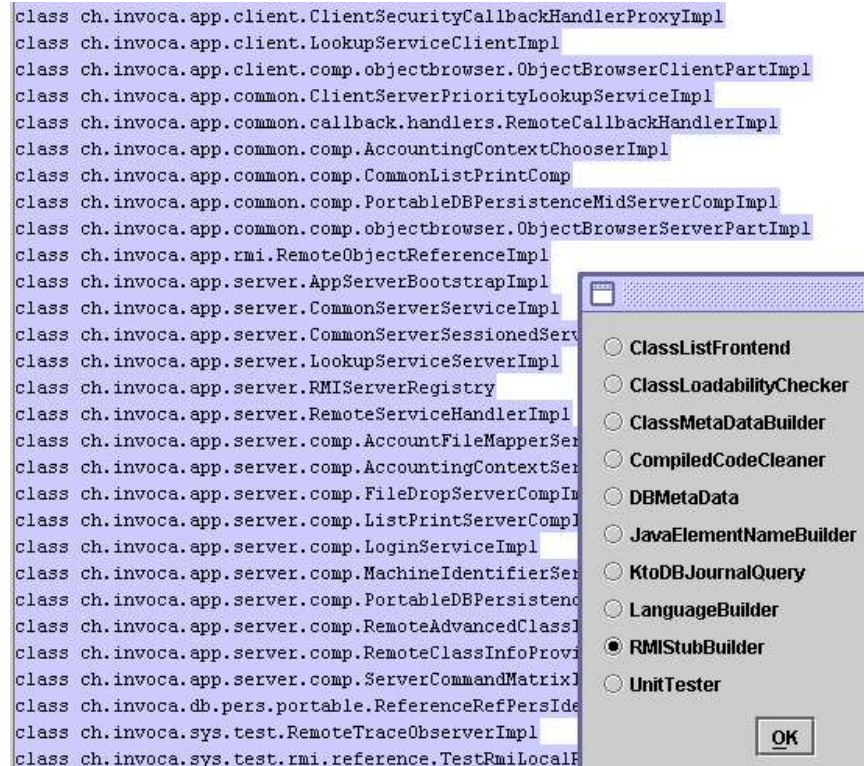
#LanguageProperties generated by com.finalogis.language.LanguageBuilder
#Thu Feb 07 00:06:14 CET 2002
title=Finalogis Application - (Swiss) German Version
```



RMISubBilder

RMI = Remote Method Invocation. Automatische Generierung der Remote-Stubs, welche zur Kommunikation zwischen Client und Server notwendig sind.

[Dependencies: ClassListFrontend]



UnitTester

Test aller Testklassen (=Alle Klassen, welche das Markerinterface Testable implementieren) à la 'JUnit'.

[Dependencies: ClassListFrontend]