1 IntarS 7 Handbuch

1.1 Das Buch

Die Informationen in diesem Buch wurden mit größter Sorgfalt erarbeitet. Dennoch können Fehler nicht vollständig ausgeschlossen werden. Die IntarS Unternehmenssoftware GmbH übernimmt keinerlei Verantwortung oder Haftung für eventuell verbliebene Fehler und deren Folgen.

Alle Warennamen werden ohne Gewährleistung der freien Verwendbarkeit benutzt und sind möglicherweise eingetragene Warenzeichen. Auch Wörter, die nicht ausdrücklich als Warennamen gekennzeichnet sind, können dennoch welche sein. Der Autor richtet sich im Wesentlichen nach den Schreibweisen der Hersteller. Das Werk, einschließlich aller seiner Teile, ist urheberrechtlich geschützt.

Kommentare und Fragen können Sie gerne an uns richten:

IntarS Unternehmenssoftware GmbH
Creidlitzer Straße 106
96450 Coburg

09561 9762709
02642 40526292

info@intars.de
www.intars.de
https://www.facebook.com/IntarsUnternehmenssoftware

1.2 Die Software

Rechtliches zu IntarS 7

```
//  IntarS Enterprise Framework
//  Author 1996-2015  Pirmin Braun
//  all Rights granted to and reserved by IntarS Unternehmenssoftware GmbH
//  Creidlitzer Str. 106 - 96450 Coburg - GERMANY
//  http://www.intars.de info@intars.de
//  licensed under IntarS Open Source Commercial License

--- IntarS Open Source Commercial License (IOSCL) ---
Your Rights
===========
IntarS Unternehmenssoftware GmbH grants you the right to
use, modify, share and distribute IntarS under these

Conditions
==========
Redistributions of source code of IntarS must retain the above copyright
notice and this License.

Redistributions in binary form of IntarS must reproduce the above copyright
notice and this License in a convenient manner.

Non-Commercial Usage of IntarS is allowed for any legal purpose.

Commercial usage requires an additional commercial license.
Commercial usage begins when switching from evaluation to productive usage.

Termination
===========
Failing to conform to a condition will automatically terminate your rights
     under this License and constitute a claim for compensation.


Disclaimer
==========
IntarS comes with absolutely no warranty of any kind to the extent of
     applicable law. Use on your own risk.
--- end of License ---

copyright notices
=================

IntarS is built with these Libraries licensed under GNU LESSER GENERAL PUBLIC
     LICENSE:

GNUstep Base Library
 Copyright (C) 2005 Free Software Foundation, Inc.

GNUstep Performance Library
```

```
 Copyright (C) 2005 Free Software Foundation, Inc.
 Written by:  Richard Frith-Macdonald <rfm@gnu.org>

GNUstep WebServer Library
 Copyright (C) 2004-2010 Free Software Foundation, Inc.
 Written by:  Richard Frith-Macdonald <rfm@gnu.org>
```

Was bedeutet das?

Weil IntarS 7 unter IOSCL steht, darf die Software genutzt, weitergegeben, verkauft, verändert und kopiert werden. Allerdings müssen die Urheberangaben und die Lizenz unverändert bleiben und es fallen für die Nutzung u.U. Lizenzgebühren an, wenn es sich um eine fortgesetzte, nicht-private Nutzung handelt. In letzterem Fall muss der Nutzer für eine Lizenz an IntarS Unternehmenssoftware herantreten.

Wer hilft mir?

- Grundsätzlich können Sie sich selbst helfen, denn IntarS 7 ist ein Open Source System. Die Struktur ist einfach, durchgängig, konsistent und dokumentiert. Jede Datei ist nach dem gleichen Schema aufgebaut und ggf. mit einem Texteditor zu bearbeiten. Sie werden überrascht sein, wie schnell Sie sich hineinfinden.
- Es gibt eine Community von IntarS 7-Benutzern und -Entwicklern, die Sie auf http://sourceforge.net/projects/intars/ kontaktieren können. Möglicherweise finden Sie dort in den Foren die gesuchte Antwort.
- Sie können sich jedoch auch an einen zertifizierten IntarS 7-Partner in Ihrer Nähe wenden. Zertifizierte IntarS 7-Partner sind von uns geschult und haben einen Nachweis ihrer Qualifikation erbracht. Sie haben die aktuellsten Informationen sowie Zugriff auf unsere Experten. Wir arbeiten eng mit ihnen zusammen und leisten 3rd-Level-Support. Eine Liste der zertifizierten IntarS 7-Partner finden Sie auf unserer Internetseite www.intars.de. Gerne geben wir auch Auskunft darüber, ob ein Dienstleister zertifiziert ist. Falls jemand das IntarS 7-Partner-Siegel führt, ohne von uns dazu autorisiert zu sein, handelt es sich um Betrug und wird strafrechtlich verfolgt.

2 Installation

2.1 Native Installation

Besorgen Sie sich den Source, z.B. über http://sourceforge.net/projects/intars/ oder direkt von subversion svn://svn.intars.at/intars/Local/Projects/IntarS7.

Folgen Sie diesen Anweisungen aus /usr/GNUstep/Local/Projects/IntarS7/Install:

Betriebssystem installieren, Pakete dazu, hier Debian 7.3 Wheezy 64 bit

Frameworks GNUstep, Performance, webserver und IntarS

```
- immer als root arbeiten
- Erfahrung mit Linux, Shell und Compiler erforderlich
- nur bash als Shell benutzen
- Rechnerzeit muss aktuell sein, sonst funktioniert install nicht

##########################################
# prerequisite: Install_Debian_*.txt

export LANG=en_US.utf8

# production releases
GS_MAKE_TAG=make-2_6_7
GS_BASE_TAG=base-1_24_8
GS_PERF_TAG=performance-0_5_0
GS_WEBS_TAG=webserver-1_5_4

mkdir -p /usr/GNUstep/Local/Projects
cd /usr/GNUstep/Local/Projects
svn checkout svn://svn.intars.at/intars/Local/Projects/IntarS7 --username
     anonsvn --password 'intars'

##########################################
cd /usr/GNUstep/Local/Projects
echo "Downloading gnustep-make"
if [ -d make ] ; then
  cd make
  svn diff |tee tmp.patch
  svn revert -R .
  svn update
  cd ..
else
  svn checkout http://svn.gna.org/svn/gnustep/tools/make/tags/$GS_MAKE_TAG
     make
fi

##########################################
cd /usr/GNUstep/Local/Projects
echo "Downloading gnustep-base"
if [ -d base ] ; then
  cd base
```

```
  svn diff |tee tmp.patch
  svn revert -R .
  svn update
  cd ..
else
  svn checkout http://svn.gna.org/svn/gnustep/libs/base/tags/$GS_BASE_TAG
      base
fi

echo "Downloading gnustep-performance"
if [ -d performance ] ; then
  cd performance
  svn diff |tee tmp.patch
  svn revert -R .
  svn update
  cd ..
else
  svn checkout
      http://svn.gna.org/svn/gnustep/libs/performance/tags/$GS_PERF_TAG
      performance
fi

echo "Downloading gnustep-webserver"
if [ -d webserver ] ; then
  cd webserver
  svn diff |tee tmp.patch
  svn revert -R .
  svn update
  cd ..
else
  svn checkout
      http://svn.gna.org/svn/gnustep/libs/webserver/tags/$GS_WEBS_TAG
      webserver
fi

##########################################
cd /usr/GNUstep/Local/Projects/make
patch_file="../IntarS7/Install/make.patch"
if [ -f $patch_file ] ; then
  echo "Applying gnustep-make patch"
  patch -Np0 < $patch_file
fi
echo "Building gnustep-make"

if [ ! -f config.status ] ; then
  ./configure CFLAGS=-g --with-layout=gnustep
fi
make install
if [ -f /usr/GNUstep/System/Library/Makefiles/GNUstep-reset.sh ] ; then
  . /usr/GNUstep/System/Library/Makefiles/GNUstep-reset.sh
fi
. /usr/GNUstep/System/Library/Makefiles/GNUstep.sh

##########################################
```

```
# /etc/ld.so.conf erweitern:
vim /etc/ld.so.conf
o

/usr/GNUstep/Local/Library/Libraries
/usr/GNUstep/System/Library/Libraries
/usr/local/lib
# ESC
:wq


##########################################
# gnustep-base
cd /usr/GNUstep/Local/Projects/base
patch_file="../IntarS7/Install/base.patch"
if [ -f $patch_file ] ; then
  echo "Applying gnustep-base patch"
  patch -Np0 < $patch_file
fi
echo "Building gnustep-base"

if [ ! -f config.status ] ; then
  ./configure --with-layout=gnustep
fi
make clean install
ldconfig

# jetzt sind environment variablen generiert worden; nochmal neu sourcen
. /usr/GNUstep/System/Library/Makefiles/GNUstep.sh

##########################################
# gnustep-performance
cd ${GNUSTEP_LOCAL_ROOT}/Projects/performance
patch_file="../IntarS7/Install/performance.patch"
if [ -f $patch_file ] ; then
  echo "Applying gnustep-performance patch"
  patch -Np0 < $patch_file
fi

echo "Building gnustep-performance"
# kein configure
make clean install

##########################################
# webserver
cd ${GNUSTEP_LOCAL_ROOT}/Projects/webserver
patch_file="../IntarS7/Install/webserver.patch"
if [ -f $patch_file ] ; then
  echo "Applying gnustep-webserver patch"
  patch -Np0 < $patch_file
fi

echo "Building gnustep-webserver"
make clean install
```

```
##########################################
# IntarS
cd ${GNUSTEP_LOCAL_ROOT}/Projects/IntarS7

echo "Building IntarS Framework"
make clean install

chmod a+x ${GNUSTEP_LOCAL_ROOT}/Projects/IntarS7/Install/*.sh
chmod a+x ${GNUSTEP_LOCAL_ROOT}/Projects/IntarS7/ShellScripts/*.sh

# IntarS_environment.sh installieren
# in diese Start Scripts eintragen:

src_intars_env=".
     /usr/GNUstep/Local/Projects/IntarS7/Install/IntarS_environment.sh"
echo $src_intars_env >> /etc/profile
echo $src_intars_env >> ~/.profile
echo $src_intars_env >> ~/.bash_profile
echo $src_intars_env >> /etc/bash.bashrc
echo $src_intars_env >> ~/.bashrc

# Test: bash aufrufen; danach sollte commandprompt farbig sein:
bash

##########################################
# bei Virtualisierung die Mandanten auf dem Host halten für Edit, Merge,
     Search, svn
# von Hand:
mount.cifs //two/ll_IntarS /usr/GNUstep/Local/Library/IntarS -o guest
mount.cifs //two/lp_IntarS7 /usr/GNUstep/Local/Projects/IntarS7 -o guest

# automatisch mittels /etc/fstab
//two/ll_IntarS /usr/GNUstep/Local/Library/IntarS cifs  guest 0 0
//two/lp_IntarS7  /usr/GNUstep/Local/Projects/IntarS7 cifs  guest 0 0


# sicherstellen, dass auch nicht-interaktive Shell Scripts das IntarS
     Environment bekommen
# z.B. in crontab
export
     BASH_ENV=/usr/GNUstep/Local/Projects/IntarS7/Install/IntarS_environment.s
     h
export ENV=/usr/GNUstep/Local/Projects/IntarS7/Install/IntarS_environment.sh
```

den IntarS Mandant

```
- immer als root arbeiten
- Erfahrung mit Linux und Shell erforderlich
- nur bash als Shell benutzen
- Rechnerzeit muss aktuell sein, sonst funktioniert install nicht

##########################################
```

```
# prerequisite: Install_Debian_*.txt, Install_IntarS.txt

. /usr/GNUstep/Local/Projects/IntarS7/Install/IntarS_environment.sh
# Beispiel fuer Mandant 000330
# gibt Fehlermeldung "No such file or directory", wenn noch nicht
     installiert; setzt aber $mandant
m 330

##########################################
# auschecken
mkdir -p /usr/GNUstep/Local/Library/IntarS
cd /usr/GNUstep/Local/Library/IntarS
svn checkout -r HEAD
     svn://svn.intars.at/intars/Local/Library/IntarS/_K_$mandant  --username
     anonsvn --password 'intars'
mp
chmod a+x *.sh

##########################################
# Start/Stop Script für runlevels
cp $INTARS_PP/ShellScripts/etc_initd_intars7.sh /etc/init.d/int7$mandant
vim /etc/init.d/int7$mandant
... mandanten-nr, Instanzen, Startparameter konfigurieren

insserv -dfv int7$mandant
# oder: /usr/lib/insserv/insserv -dfv int7$mandant
# falls wieder entfernt werden soll: insserv -rfv int7$mandant

# unnoetige Dienste weg:
insserv -rvf /etc/init.d/exim4
insserv -rvf /etc/init.d/courier-authdaemon
insserv -rvf /etc/init.d/cups
insserv -rvf /etc/init.d/cups-browsed
insserv -rvf /etc/init.d/motd

# System-Module und Scripts aus Framework kopieren
mp
rsy

# my_config.txt erstellen, das nicht unter svn Kontrolle steht
cp config.txt my_config.txt

# ggfs. editieren z.B. fuer MariaDB Zugangskennung

##########################################
# Demodaten einlesen falls gewünscht; wird ohne Datenbank gestartet, legt
     IntarS diese selbst an und befüllt elementare Tabellen wie z.B.
     Mengeneinheiten, Waerungen

mysql -uroot -proot  -e "drop database intars_$mandant;"
mysql -uroot -proot  -e "create database intars_$mandant CHARACTER SET utf8;"
mysql -uroot -proot --default-character-set=utf8 intars_$mandant <
     $INTARS_MP/DataArchiv/dumps/Sicherung_Demo.sql
```

```
# testweise starten und dabei ggfs. Datenbankunterschiede aufloesen:
adj

# ggfs. Datenbank anpassen
adjbat

##########################################
# samba einrichten: alles offen
cd /etc/samba
mv smb.conf smb.conf.bak
vim smb.conf
# das hier reinpasten:
[global]
workgroup = WORKGROUP
# fuer Windows-Namensaufloesung:
dns proxy = no
netbios name = ${hostname}
name resolve order = lmhosts host wins bcast
writeable = Yes
read only = No
browseable = Yes
security = USER
map to guest = Bad User
guest ok = yes
guest account = root
wins support = Yes
[Alles]
path = /
writeable = Yes


/etc/init.d/samba restart

##########################################
# crontab einrichten
vim /etc/crontab
# so editieren:

SHELL=/bin/bash
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
BASH_ENV=/usr/GNUstep/Local/Projects/IntarS7/Install/IntarS_environment.sh

# Standard
0 23 * * * root m 330; $INTARS_PP/ShellScripts/Batch_nightly.sh

# oder individuelles Script
00 3 * * * root m 330; $INTARS_MP/Batch_nightly.sh

# test alle 3 Minuten
*/3 * * * * root m 330; $INTARS_PP/ShellScripts/Batch_dummy.sh

##########################################
# https einrichten
Zertifikat und Key hinterlegen
```

```
$INTARS_MP/Resources/SSL/ca-cert.pem
$INTARS_MP/Resources/SSL/ca-key.pem

in /etc/init.d/int7$mandant den Startparameter "-https" hinzufügen

##########################################
Starten und Operating s.
$INTARS_PP/Install/intars7_commands.txt
```

Befehlssammlung für Operating

```
# Maintenance und Operating
# s. Aliases in IntarS_environment.sh

# zum Mandant 000330 wechseln
m 330

# update Mandant
mp
svn update
chmod a+x *.sh

# update Framework
pp
svn update
chmod a+x ShellScripts/*.sh
chmod a+x Install/*.sh

# Framework compile
pp
make clean
make
make install

# Manuell interaktiv starten, um DB-Unterschiede zu prüfen:
# mit drop Statements ####################################
adj

# DB-Differenzen kontrollieren und beheben:
adjbat

# f. Demo-Server dump erzeugen, der nachts wieder restored wird
# nach svn update und Datenbankanpassung Dump erzeugen und committen
mysqldump -uroot -proot -h localhost --opt --default-character-set=utf8
     intars_$mandant > $INTARS_MP/DataArchiv/dumps/Sicherung_Demo.sql

# Ausführen eines Batch-Scripts:
$INTARS_APP -mandant $mandant -n 99 -batch _Batch/notify_wv

# normaler Start im Hintergrund:
DATE=`date +"%Y%m%d%H%M"`
LOGPATH=$INTARS_MP/Resources/Logs
$INTARS_APP -mandant $mandant &>$LOGPATH/NSLog_1_${DATE}.txt &
```

```
# 2. und weitere Instanz bei Bedarf (Skalierung durch parallele Prozesse)
$INTARS_APP -mandant $mandant -n 2 &>$LOGPATH/NSLog_2_${DATE}.txt &

##########################################
# Demodaten einlesen
mysql -uroot -proot  -e "drop database intars_$mandant;"
mysql -uroot -proot  -e "create database intars_$mandant CHARACTER SET utf8;"
mysql -uroot -proot --default-character-set=utf8 intars_$mandant <
     /$INTARS_MP/DataArchiv/dumps/Sicherung_Demo.sql

# Als Deamon über individuelles /etc/init.d/int$mandant oder falls nicht
     vorhanden generisch starten/stoppen
sta
sto

# interaktiv mit autologin starten:
sti

# neuesten  log analysieren:
logs

# zusammencatten aller logs:
find . -name 'log_?_*.txt' -mtime -3 -print -exec cat {} > allLogs.txt \;
less allLogs.txt

20.02.2011 22:00:04
sort -S 20% -t ' ' -k 1.7,1.10 -k 1.4,1.5 -k 1.1,1.2 -k 2,2 allLogs.txt >
     sortedAllLogs.txt

##########################################
# Mandant deinstallieren
rm -Rf $INTARS_MP
mysql -uroot -proot -e "drop database intars_$mandant;"
insserv -rv int7$mandant
rm /etc/init.d/int7$mandant
aus crontab und samba.conf rauslöschen
```

2.2 Virtual Appliance

IntarS 7 wird auch als sog. Virtual Appliance ausgeliefert. Dies ist vereinfacht gesagt, ein komplett vorinstalliertes, sofort lauffähiges System in einer einzigen Datei. Die Datei wird Virtual Image genannt. Um IntarS 7 laufen zu lassen, benötigen Sie noch eine Virtual Machine, welche das Virtual Image lädt und ausführt. IntarS 7 wird von folgenden frei erhältlichen und kostenlosen Virtual Machine-Produkten unterstützt:

- VMWare Player, VMWare Server von VMWare (tm)
- VirtualBox von Innotek (tm), jetzt Sun Microsystems (tm)
- VirtualPC 2004 und 2007 von Microsoft (tm)

Diese sind kostenlos per Download erhältlich:

- http://www.vmware.com/de/products/free_virtualization.html
- http://www.virtualbox.org/wiki/Downloads

Wählen Sie die Virtualisierungslösung, die Ihnen am meisten zusagt bzw. von Ihrem Host-Betriebssystem unterstützt wird. Virtual PC 2007 eignet sich für Windows Vista (tm), jedoch weniger für Windows XP (tm). Für Windows 2000 ist VirtualPC 2004 von Vorteil. Die anderen beiden Virtualisierungsprodukte sind kompatibler. Für Windows 7 ist VirtualPC kostenlos per Download erhältlich.

Warum Virtualisierung?

- Es ist einfach und funktioniert sofort.
- Es ist sicher; IntarS 7 wird in einer geschützten Umgebung betrieben.
- Es gibt keine unerwünschten Wechselwirkungen mit bereits installierter Software (z.B. inkompatible DLLs).
- IntarS 7 ist eine komplexe Unternehmenssoftware mit Datenbank, Applikations-, Mail-, Fax-, Druckserver u.v.m. Eine klassische Installation würde einen ganzen Tag in Anspruch nehmen und viel Erfahrung erfordern. Diese Arbeit haben wir Ihnen vorab schon abgenommen. Sie erhalten eine echte, harmonisierende Serverlösung, ohne dass Sie sich um die Details kümmern müssen.
- Höhere Qualität und Zuverlässigkeit: in dem Virtual Image passt alles optimal zusammen; unangenehme Überraschungen mit nicht kompatiblen Paket-Versionen, Systemeinstellungen und Derivaten sind ausgeschlossen.
- Kein Ärger mit Registry, DLLs oder Installer.
- Denkbar einfache Sicherung: nur eine Datei (das Virtual Image) ist zu sichern.
- Einfacher Umzug auf einen anderen Rechner: lediglich das Virtual Image auf den neuen Rechner kopieren und sofort weiterarbeiten.
- Schnelles "Disaster-Recovery": das Virtual Image der letzten Sicherung wird auf einen neuen Rechner kopiert.

Gehen Sie bitte folgendermaßen vor:

Laden Sie sich die Datei von http://sourceforge.net/projects/intars/ oder von unserer Homepage http://www.intars.de herunter. Dies ist eine ca. 400 MB große, gepackte VMWare Appliance. Entpacken Sie sie und starten Sie VMWare Player oder VMWare Workstation o. ä. Es ist ein Debian-System mit komplett installiertem und konfiguriertem IntarS-Mandant und vollständigen Demodaten.

1. Falls Sie noch keine Virtual Machine im Einsatz haben, besorgen Sie sich eine (s. oben).
2. Kopieren Sie das Virtual Image auf die Festplatte, entpacken es und laden es in die Virtual Machine.

Starten Sie die Virtual Machine. Warnungen bezüglich unterschiedlicher Prozessor-Architektur können ignoriert werden.

3. VMWare fragt evtl. nach, ob eine neue ID erzeugt werden soll, da die VM kopiert wurde. Klicken Sie auf "Ja".
4. In der Virtual Machine bootet Debian und IntarS 7 wird hochgefahren. Sobald das System gebootet hat, kann man über den Browser damit arbeiten.
5. Öffnen Sie den Mozilla Firefox oder Chrome-Browser und geben Sie ein: http://IntarS 7:12301
6. Es sollte sich das Anmelde-Panel zeigen.
7. Die URL mit Anmeldedaten ist: http://IntarS 7:12301?pw=root&loginname=Administrator
8. Falls das nicht funktioniert, muss über die IP-Adresse gearbeitet werden: loggen Sie sich dazu mit Kennung "root" und Passwort "root" in die Konsole der Virtual Machine ein und geben "ifconfig" ein. Die IP von eth0 ist die gewünschte.
9. Falls eth0 nicht zu sehen ist, oder keine ordentliche IP erhalten hat, sollten Sie die Netzwerkonfiguration der Virtual Machine ändern oder eine andere Virtual Machine verwenden.
10. Wenn über IP gearbeitet werden muss, lautet die URL: http://xxx.xxx.xxx.xxx:12301 wobei xxx.xxx.xxx.xxx die IP-Adresse ist.
11. Loggen Sie sich in IntarS 7 mit Kennung "Administrator" und Passwort "root" ein. Anschließend befinden Sie sich in der laufenden Anwendung. Für den Anfang lesen Sie am besten die Hilfe.
12. Wenn das System sich Ihnen erschließen soll, können Sie sich mithilfe von SSH einloggen (root, root) und die Struktur ergründen. Beginnen Sie am besten bei /usr/GNUstep/Local/Library/IntarS/_K_000201.

3 Erste Schritte

3.1 Starten und Stoppen

Virtual Machine

IntarS 7 startet, wenn Sie die Virtual Machine starten. Warten Sie, bis der Startvorgang abgeschlossen ist. Dies ist der Fall, wenn in der Konsole ein Login-Prompt zu sehen ist. Danach kann über den Firefox oder Chrome mit IntarS 7 gearbeitet werden.

Zum Stoppen klicken Sie als Administrator im "Home"-Modul unter "Service" auf den "Herunterfahren"-Button. Beobachten Sie in der Virtual Machine-Konsole den Shutdown. Wenn die Meldung "System halted" erscheint, können sie die Virtual Machine abschalten.

Native Installation

Betreiben Sie IntarS 7 nativ, wird IntarS 7 beim Starten des Rechners über die "init"-Scripts gestartet. Manuell können Sie IntarS 7 mit

```
/etc/init.d/int7000201 start
```

starten und mit

```
/etc/init.d/int7000201 stop
```

wieder stoppen. Weitere Informationen zum Starten und zur Systempflege finden Sie in der Datei "intars_commands.txt" im Mandantenpfad /usr/GNUstep/Local/Library/IntarS/_K_000201/.

Kommandozeilen-Parameter

```
-secure
```

Die Anwendung läuft im sicheren Modus. Alle Funktionalitäten, welche das System beschädigen oder die Sicherheit gefährden könnten, sind deaktiviert. Auch der Administrator kann diese nicht mehr ausführen. Dies dient dazu, die Anwendung als Demo auf einem öffentlich zugänglichen Server zu betreiben.

```
-batch scriptname
```

Die Anwendung wird im Batch-Modus wie ein Kommandozeileninterpreter gestartet, führt das angegebene Script mit den Berechtigungen des Batch-Users aus und beendet sich wieder. Damit werden zeitgesteuerte automatische Verarbeitungen realisiert, z. B. nächtliche Statistik-Verdichtungen.

```
-mandant nummer
```

Gibt an, unter welchem Mandanten die Instanz laufen soll. Wird normalerweise aus dem Filename des Binaries extrahiert. Diese Option wirkt stärker.

```
-n nummer
```

Gibt die Instanznummer an. Default ist '1'. Die Anwendung skaliert, indem mehrere Instanzen (=Prozesse) parallel laufen. Jede Instanz muss eine eigene Nummer haben. Der Loadbalancer verteilt die Last auf die Instanzen.

```
-port nummer
```

Optionale Definition eines eigenen Ports zur Kommunikation zwischen dem Apache-Modul und der Applikations-Instanz. Standardmäßig wird der Port automatisch aus Mandant- und Instanznummer berechnet.

```
-db_mandant nummer
```

Angabe eines abweichenden Mandanten für die Datenbank.

3.2 Aufrufen

1. Öffnen Sie den Firefox oder Chrome Browser.
2. Geben Sie als URL http://IntarS 7:12301 ein.
3. Erscheint die Login-Seite, speichern Sie die URL als Bookmark.
4. Evtl. ist es notwendig, über IP zu gehen. Dann lautet die URL http://xxx.xxx.xxx.xxx:12301 wobei xxx.xxx.xxx.xxx die von DHCP vergebene IP des IntarS 7-Systems ist.

Anmelden

Geben Sie auf der Login-Seite die Kennung und das Passwort ein, das Ihnen vom Systemverwalter mitgeteilt wurde. Sind Sie selbst der Systemverwalter und melden sich zum ersten Mal an, lautet die Kennung

```
Administrator
```

und das Passwort

```
root
```

Drücken Sie dann die "Return"-Taste. Funktioniert alles, erscheint Ihre persönliche Start-Seite.

Ist die Kennung unbekannt, das Passwort falsch oder die Kennung gesperrt, teilt Ihnen das System dies mit.

Das Passwort wird übrigens als MD5-Hash gespeichert. Es kann von niemandem gesehen oder rückverschlüsselt werden. SQL-Insertion funktioniert nicht. Auch der Systemverwalter kann nur ein neues Passwort vergeben, aber nicht nachschauen, wie das aktuelle Passwort lautet.

Sie können sich mit Ihrer Kennung beliebig oft im System anmelden. Vor allem im Mozilla Firefox oder Chrome mit dessen tabbed Browsing eröffnet dies großartige Möglichkeiten der Arbeitsorganisation. Verwenden Sie z. B. ein Fenster, um Notizen zu erfassen, ein zweites, um Angebote zu erstellen und ein drittes, um in Geschäftsvorfällen zu recherchieren.

Haben Sie das Administrator-Passwort vergessen, muss direkt über SQL der Zugang wieder hergestellt werden. Sie benötigen dazu einen lokalen Zugang zum MySQL-Server, da dieser standardmäßig so konfiguriert ist, dass er keine Verbindungen von einem anderen Rechner akzeptiert. Mit folgendem Befehl kommen Sie in die Kommandozeile des MySQL-Clients:

```
mysql -uroot -proot intars_000201
```

So stellen Sie den Administrator-Benutzer wieder her:

```
update vid_benutzer set cryptedpw = md5('root'), salt='' where kennung =
     'Administrator';
```

Alternative Anmeldung

Sie können beim ersten Aufruf gleich User und Kennwort als Parameter mit übergeben:

```
http://IntarS 7:12301?pw=root&login=Administrator
```

Dies ist komfortabel, aber gefährlich. Sind Sie der einzige Benutzer des Systems oder kennt jeder die Passwörter von jedem Kollegen, ist das vertretbar.

Mit weiteren Parametern können Sie gleich ein Modul ansteuern und sogar Datensätze aufrufen:

```
http://IntarS 7:12301?pw=root&login=Administrator&zielmodul=vid_kunde
```

3.3 Start

Nach dem Anmelden befinden Sie sich im "Start"- oder "Home"-Modul. Es ist vergleichbar mit der Homepage einer Webseite. Hier beginnt alles und hier hin kann man jederzeit zurückkehren.

Sinnvollerweise liegen im Start-Modul alle Elemente, die Sie zuerst sehen möchten.

Da dies bei jedem anders ist, können Sie sich die Startseite selbst einrichten.

Das Hauptmenü und die Modulsuche sind auch auf der Startseite. Diese beiden Elemente werden in separaten Kapiteln behandelt.

Wenn Sie einfach das Browser-Fenster schließen, besteht die Sitzung noch eine Stunde lang weiter, bis sie automatisch beendet wird. Der Server bemerkt es nicht, wenn Sie einfach das Fenster schließen. Es wird weder die Anwendung dadurch beendet noch kommt es zum Datenverlust.

Im Gegensatz zu PHP-Anwendungen arbeitet IntarS 7 nicht mit Cookies. Somit ist es möglich, sich von einem Rechner mit mehreren Benutzern anzumelden.

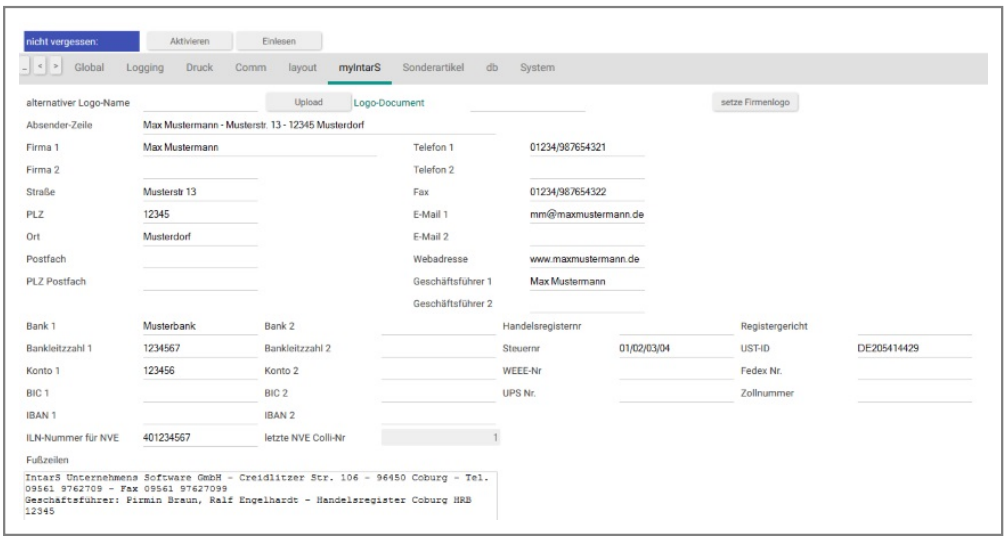
3.4 Wo finde ich Hilfe

Hier finden Sie Hilfe:

- Hilfe-Modul (oben rechts im Browser)
- Gedrucktes Handbuch
- Modulhilfe - in aktivem Modul nochmals auf dessen Name klicken; erklärt jedes Feld und jeden Button des Moduls
- Button-Tooltipps
- Im IntarS-Anwenderforum auf http://sourceforge.net/projects/intars
- Hotline der IntarS Unternehmenssoftware GmbH (www.intars.de)

3.5 Firmendaten erfassen

Ihre Firmenstammdaten erfassen Sie im Modul "Config" unter dem Register "myIntarS". Farben und Layout können anders aussehen. Die Firmenstammdaten werden u. a. auf der Geschäftskorrespondenz angedruckt.

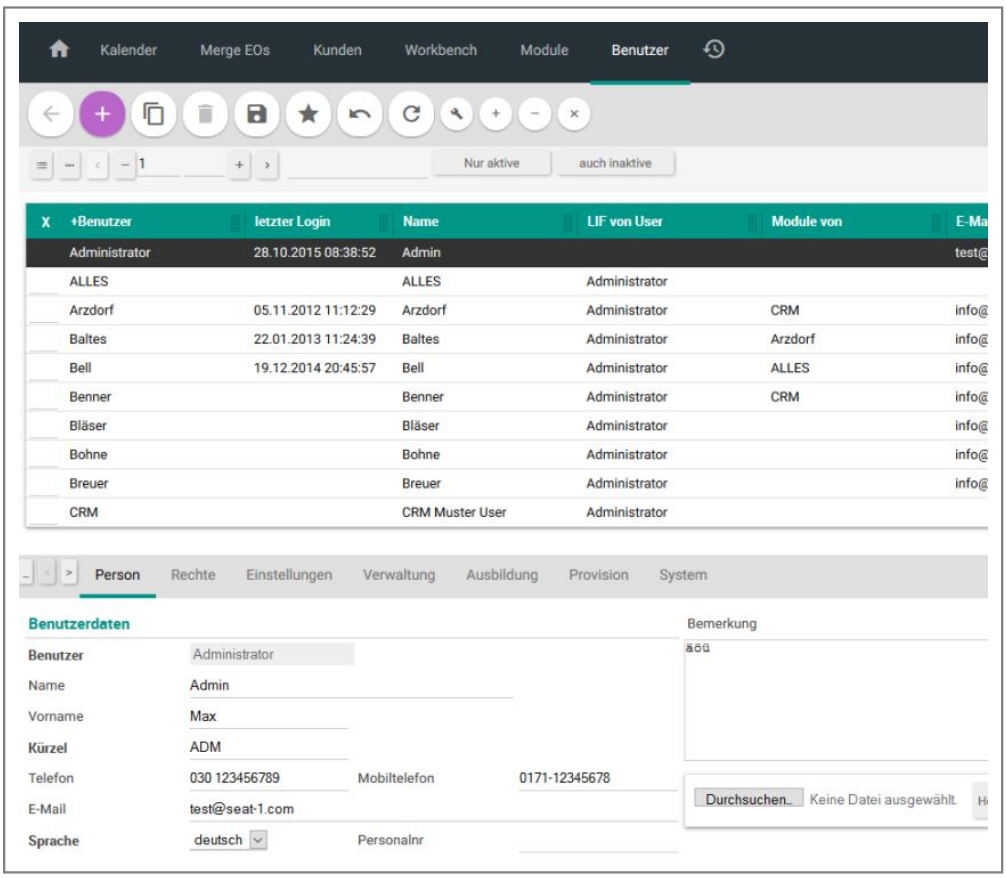


Firmenstammdaten erfassen

Laden Sie Ihr Firmenlogo mit "Upload" hoch, wählen Sie das dabei entstandene Dokument und setzen es als Firmenlogo für Ihre Geschäftskorrespondenz ein.

3.6 Benutzer einrichten

Der Administrator, welcher uneingeschränkte Berechtigungen im System hat, ist bereits vorkonfiguriert. Sollen jedoch weitere Benutzer mit dem System arbeiten, müssen sie im Modul "Benutzer" erfasst und ihre Rechte verwaltet werden.



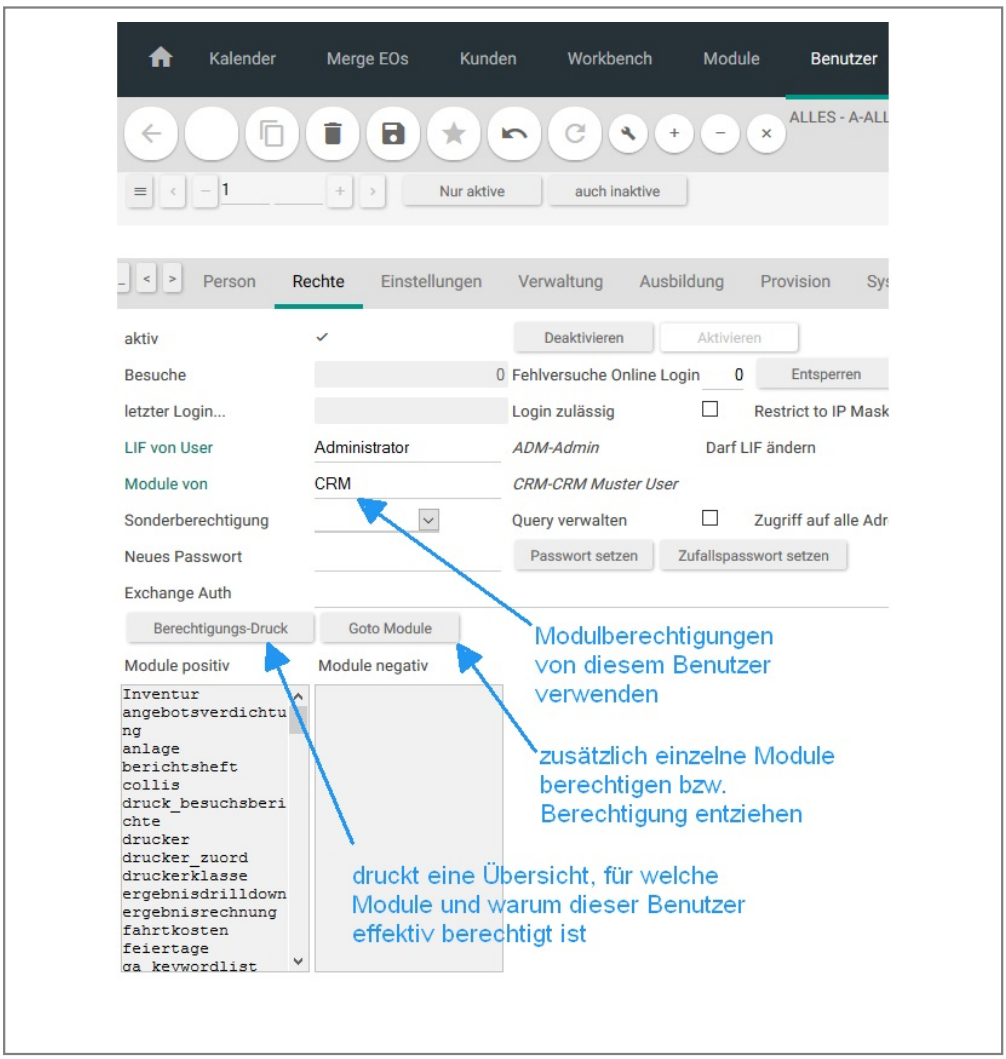
Benutzer Daten

Zunächst müssen Sie die Personendaten erfassen. Beachten Sie dabei die gültigen Datenschutzbestimmungen (Einwilligung des Benutzers nötig).

Die "Signature" wird an E-Mails angehängt, die der Benutzer aus dem System heraus verschickt.

"Session-Timeout" ist die Zeit in Sekunden, nach der eine inaktive Session vom System automatisch beendet wird.

Im Feld "Modul f. Prestarting" lassen sich, mit Kommata getrennt, Modulnamen aufführen, die beim Login gleich vorgestartet werden sollen. Man erreicht dies auch, über die Buttons "+" und "-" unter den Modulnamen. Das System trägt dann das aktuelle Modul ein bzw. nimmt es heraus. Hier kann das Feld zusätzlich manuell verwaltet werden.



Benutzer Rechte

Unter dem Register "Rechte" wird gesteuert, welche Berechtigung der Benutzer besitzt. Mit der Sonderberechtigung "Administrator" hat er wie der Administrator uneingeschränkte Berechtigungen.

Über "Aktivieren"/"Deaktivieren" lässt sich ein Benutzerprofil ein-/ausschalten. Ist es deaktiviert, kann sich der Benutzer nicht mehr anmelden und wird auch nicht mehr in Verarbeitungsläufen (z. B. Mailings) berücksichtigt. Möchten Sie lediglich die Anmeldung verhindern, ist dies über die Einstellung "Login zulässig" möglich.

Im Feld "Fehlversuche Online Login" zählt das System mit, wie oft das Passwort falsch eingegeben wurde. Hat der Benutzer (oder ein Hacker) zu oft falsch geraten, wird das Profil vom System gesperrt. Der Administrator kann die Anzahl Fehlversuche dann manuell wieder auf "0" stellen.

Mit "Restrict to IP Mask" kann der Netzwerkbereich eingeschränkt werden, aus dem sich der Benutzer einloggen darf. Ist "Query verwalten" angehakt, darf der Benutzer Queries erstellen. Das sollten nur erfahrene Benutzer. Ausserdem ist es ein Sicherheitsrisiko, weil in den Queries freies SQL eingegeben werden kann.

Mit dem Sonderrecht "Zugriff auf alle Adressen" erhält ein Benutzer auch die Kunden, für die er nicht als Vertreter drinsteht.

"Exchange Auth" ist der Zugriffs-Schlüssel für die Microsoft Exchnange Schnittstelle.

Soll der Administrator einem anderen User ein neues Passwort setzen, gibt er dies ins Feld "Neues Passwort" ein und drückt auf "Passwort setzen".

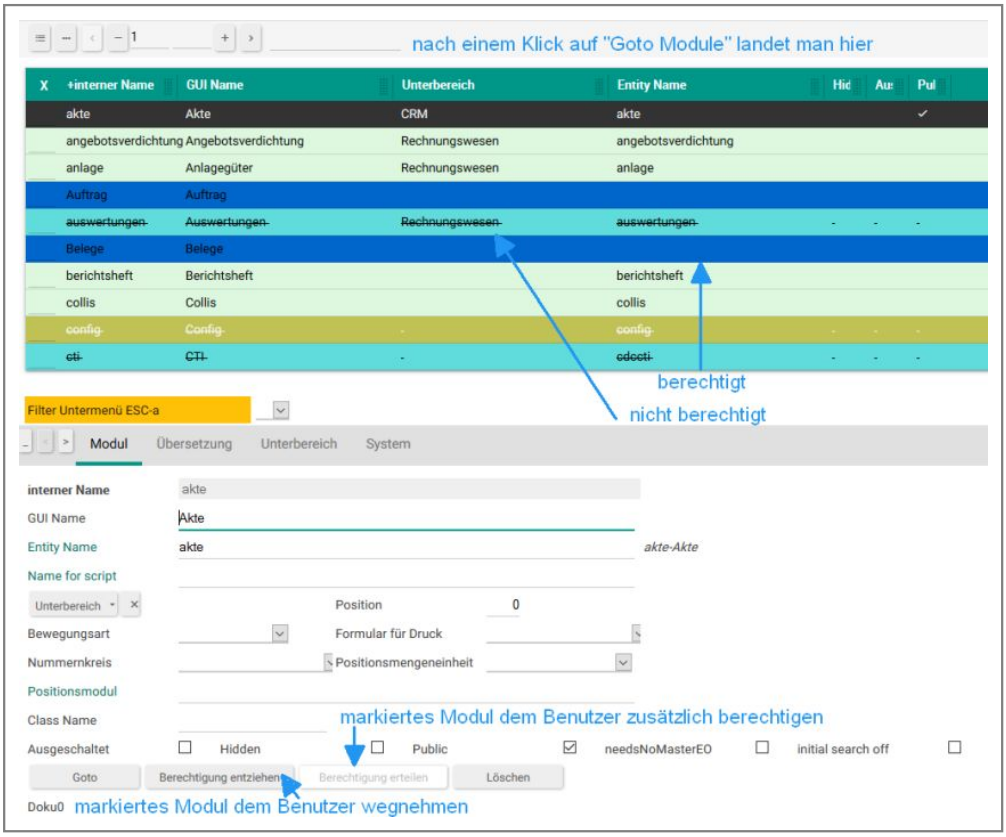
Ein wirklich sicheres Passwort liefert die Funktion "Zufallspasswort setzen". Es wird vom System ermittelt, eingetragen und einmalig angezeigt, um es dem Benutzer mitteilen zu können.

Das Oberflächenlayout bezieht ein Benutzer primär von einem anderen Benutzer, der in "LIF von User" eingetragen wird. So lassen sich einheitliche LIFs für mehrere Benutzer realisieren. Die Oberfläche wird zentral unter dem "LIF von User"-Benutzer eingestellt und die Benutzer, die darauf verweisen, haben dann diese Einstellung. Ob der Benutzer den 'C' Button zur Anpassung der Oberfläche angeboten bekommt, wird über das Flag "Darf LIF ändern" gesteuert. Wird bei "LIF von User" nichts hinterlegt, wird automatisch "Administrator" angenommen. Verändert dann der Benutzer seine Oberfläche, wird unter seinem Namen ein eigenes LIF angelegt.

Mit dem Button "Berechtigungs-Druck" wird für den markierten Benutzer gedruckt, für welche Module er effektiv berechtigt ist und warum.

Modulberechtigungen bekommt ein Benutzer zweistufig über ein Berechtigungsprofil und nachgeschaltete individuelle Zusatzberechtigungen bzw. Einschränkungen. Das Berechtigungsprofil ist ein anderer Benutzer, dem man für einen bestimmten typischen Aufgabenbereich alle dazu notwendigen Modulberechtigungen erteilt hat. Diese Berechtigungsprofil-Benutzer sind nicht zum anmelden gedacht, sondern lediglich, um die Berechtigungen an einer Stelle zentral einstellen zu können.

Individuelle Zusatzberechtigungen bzw. Einschränkungen werden für den markierten Benutzer verwaltet, indem man auf "Goto Module" klickt.



Benutzer Rechte

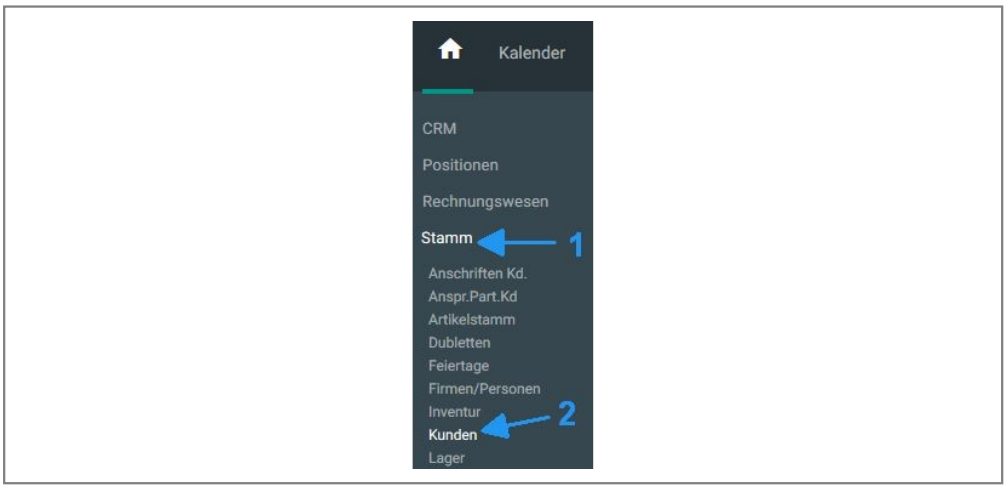
Dort sieht man im Module-Modul, für welche Module der Benutzer momentan berechtigt ist und für welche nicht (durchgestrichen). Mit Klick auf die Buttons "Berechtigung entziehen" bzw. "Berechtigung erteilen" kann dies für die markierten Module geändert werden.

Benutzer können mit dem Modul "Profil" bestimmte persönliche Daten, darunter ihr Passwort selbst verwalten.

Der Benutzer "Administrator" nimmt eine Sonderrolle ein. Er kann nicht gelöscht oder gesperrt werden, hat immer umfassende Berechtigungen und immer ein eigenes LIF. Es ist das Default-LIF des Systems.

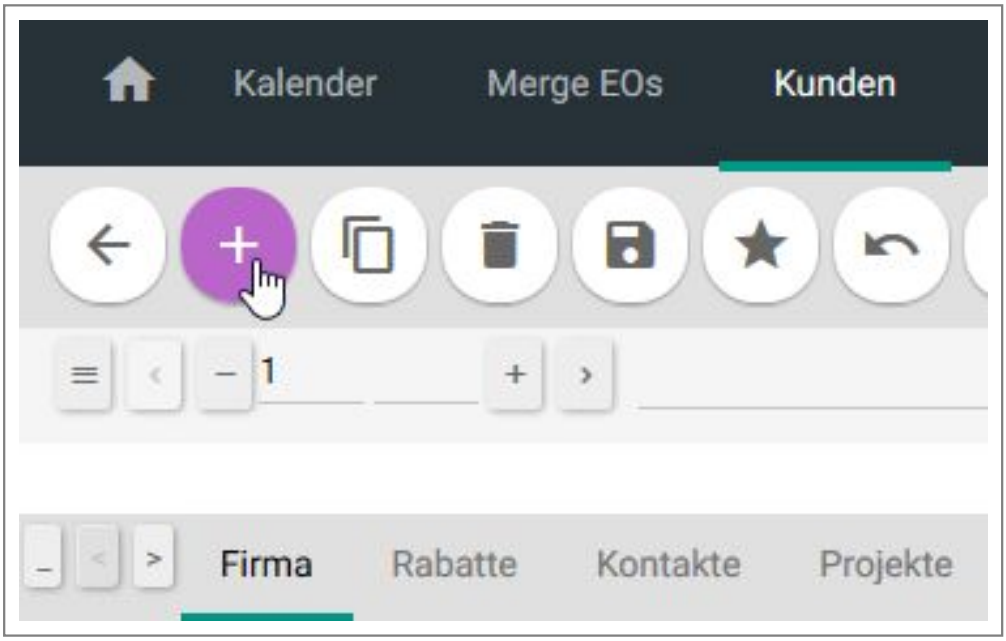
3.7 Stammdaten erfassen

Kunden erfassen



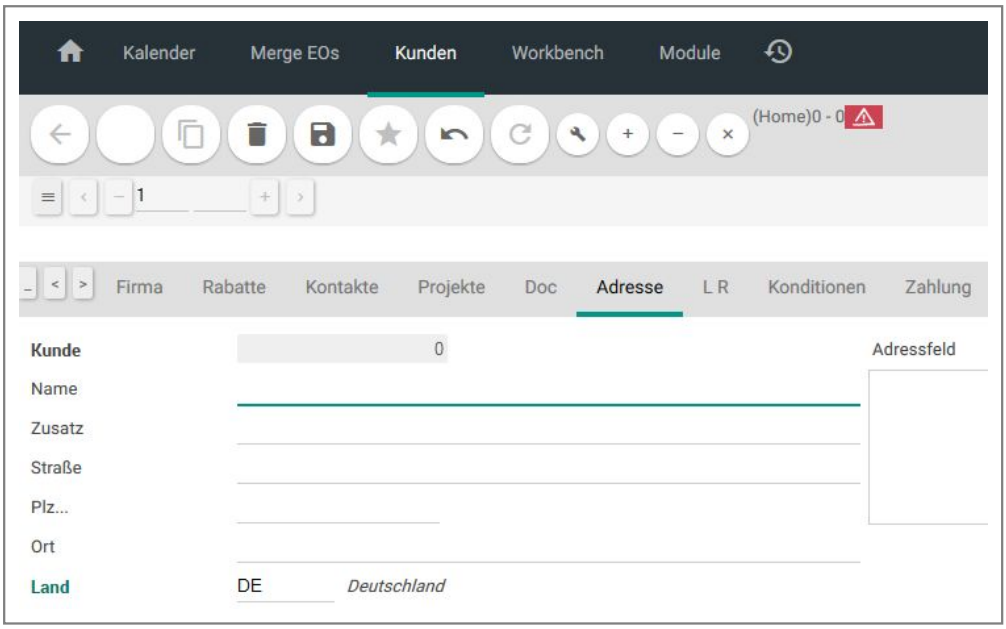
Kundenmodul aufrufen

Navigieren Sie zum Kundenmodul, klicken Sie auf "Neu" oder drücken die Tastenkombination "Alt-n". Beachten Sie das blinkende Dreieck. Es signalisiert, dass ungesicherte Daten vorliegen. Dies sagt Ihnen auch der Tooltipp, wenn Sie den Mauszeiger mind. 2 Sekunden auf das Dreieck positionieren. Allen anderen Bedienelementen sind ebenfalls Tooltipps hinterlegt. Probieren Sie es aus, indem Sie den Mauszeiger mind. 2 Sekunden auf verschiedene Buttons platzieren.



Neuen Kunden anlegen

Füllen Sie die Felder aus und arbeiten Sie sich durch die Register. Haben Sie alles eingegeben, klicken Sie auf "Sichern" oder drücken die Tastenkombination "Alt-s". Sollten Sie Pflichtfelder nicht gefüllt haben, wird das System Ihnen dies mitteilen und nicht sichern. Vervollständigen Sie dann Ihre Eingabe. Das Gleiche passiert, wenn Sie ungültige Werte (z. B. nicht existierende Schlüssel von Hilfstabellen) eingeben.



Felder ausfüllen

Hilfstabellen

Bei der Erfassung sind Sie auf verschiedene Hilfstabellen, wie z. B. Land, Skonto-Profil, Artikelgruppe, gestoßen. Evtl. müssen Sie dort weitere Daten erfassen. Einträge aus Hilfstabellen lassen sich suchen, nachschlagen und auswählen.

Artikel erfassen

Verfahren Sie analog mit dem "Artikelstamm"-Modul. Sie werden erkennen, dass es exakt dieselbe Struktur aufweist. Dies ist ein herausragendes Merkmal von IntarS 7. Jedes Modul lässt sich einheitlich bedienen.

3.8 Drucken

IntarS 7 druckt, indem ein PDF nativ erzeugt wird. Das PDF-Dokument wird unter Verwendung eines PDF-Viewers (z.B. Sumatra, Foxit-Reader oder Adobe Reader) in einem neuen Browser-Fenster oder -Tab angezeigt. Dieser muss daher auf dem Client installiert sein und es muss im Browser die Erlaubnis erteilt werden, dass Popup-Windows von der IntarsS-Server Adresse geöffnet werden dürfen.

Chrome stellt das PDF ohne ein zusätzliches Plugin dar.

In neueren Firefox Versionen (Stand 2013) ist ebenfalls ein in Javascript geschriebener PDF-Viewer eingebaut, der aber noch nicht ordentlich funktioniert.

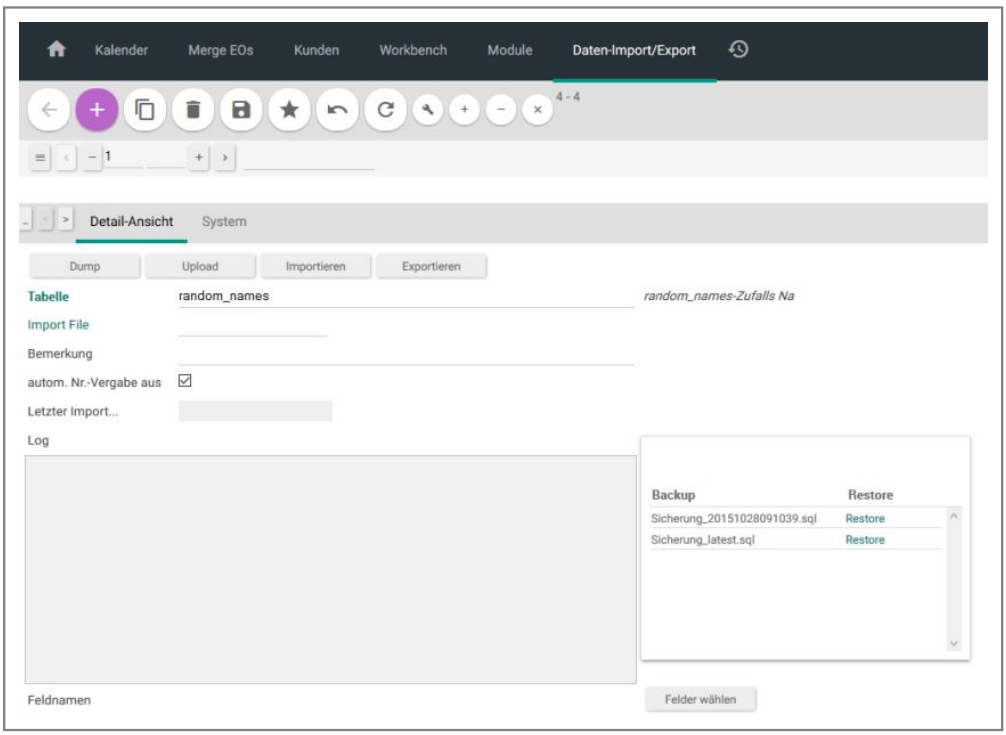
3.9 Daten importieren

Für freie Datenübernahme aus CSV-Files gibt es das Modul "Daten-Import/Export". Es arbeitet tabellenbezogen und erwartet CSV-Files in UTF-8 Kodierung, deren erste Zeile die Spaltennamen beinhaltet. Man kann damit auch den Tabelleninhalt dumpen und wieder restoren. Ein Plugin zeigt die bisherigen dumps, jeweils mit Datum.

Desweiteren können Sie einen CSV-Export durchgeführen. Dabei werden ebenfalls in der ersten Zeile die Spaltennamen ausgegeben. Am besten gehen Sie folgendermaßen vor:

1. Legen Sie einen neuen Satz für die Tabelle an und schreiben Sie dazu, was und warum Sie das tun möchten.
2. Klicken Sie auf "Dump".
3. Klicken Sie auf "Exportieren".
4. Benutzen Sie das nun erhaltene CSV-File, um die eigenen Import-Daten mit den Spaltennamen zu versehen.
5. Erstellen Sie ein CSV (z. B. mit OpenOffice Calc), welches die gewünschten Spalten enthält.
6. Laden Sie das CSV hoch ("Upload") und erfassen Sie eine Beschreibung, sodass Sie es später wiederfinden.
7. Tragen Sie das dabei entstandene Dokument in den unter 1. angelegten Satz ein und speichern Sie.
8. Klicken Sie auf "Importieren".

Funktioniert irgendetwas nicht, können Sie mit "Restore" das unter 2. erzeugte Dump wieder einspielen. Sie haben dann den ursprünglichen Zustand wieder hergestellt.



Beispiel für Datenimport

Weitere Optionen

Sie können beim Export die Spalten angeben, welche exportiert werden sollen. Auch beim Import müssen nicht alle Spalten vorhanden sein. Es werden die verarbeitet, welche vorhanden sind.

Wollen Sie beim Import vorhandene Sätze updaten, muss die Spalte des Primary Key im Import vorhanden sein. So kann z. B. der Preis im Artikelstamm upgedated werden.

Gibt es in der Datenbank bereits einen Satz mit demselben Primärschlüssel wie ein importierter Datensatz, wird er mit den importierten Daten upgedated. Andernfalls wird aus dem importierten Datensatz ein neuer in der Datenbank erzeugt (insert).

Mit dem Flag "autom. Nr.Verg. aus" kann man steuern, ob beim Insert die automatische Nummernvergabe der Datenbank ausgeschaltet sein soll. Es wird dann versucht, mit dem Wert des Primärschlüssels zu inserten. Das kann sinnvoll sein, um z. B. bestehende Kundennummern zu behalten.

Migrationen

Soll IntarS 7 eine bereits bestehnede Software ersetzen, so bietet es sich an, die Daten zu migrieren, um alle Relationen und Funktionen aufrecht zu halten. Dies erfordert allerdings eine vorherige Prüfung der Daten und ein Abstimmen von IntarS 7 auf die entsprechende Verwendung.

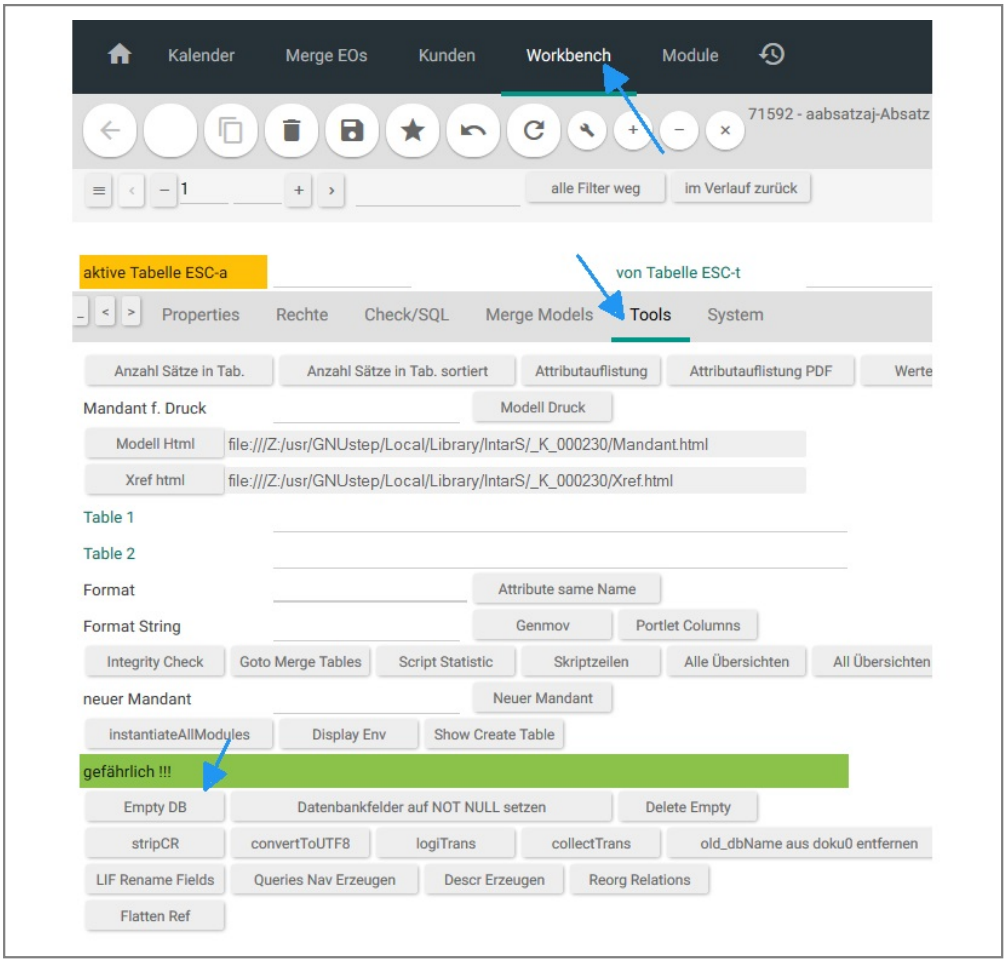
Dabei stehen wir oder einer unserer Partner Ihnen gerne zur Verfügung.

Folgende Systeme haben wir zum Beispiel bereits mehrfach erfolgreich migriert.

- SugarCRM
- Cobra
- 42ERP
- Manus

3.10 Demodaten löschen

IntarS 7 wird mit Demo-Daten ausgeliefert. Damit kann gefahrlos geübt und getestet werden. Bevor richtig mit dem System gearbeitet wird, müssen die Demo-Daten gelöscht werden. Dazu ist ein Skript vorbereitet, das der Administrator ausführen kann. Das Script befindet sich im "Workbench"-Modul, Register "Tools" und heisst "Empty DB". Klickt man darauf, geht das System alle Tabellen durch, zeigt an, wieviel Sätze darin sind und fragt, ob diese gelöscht werden sollen. Bestätigt man, wird gelöscht, sonst wird die Tabelle übersprungen.

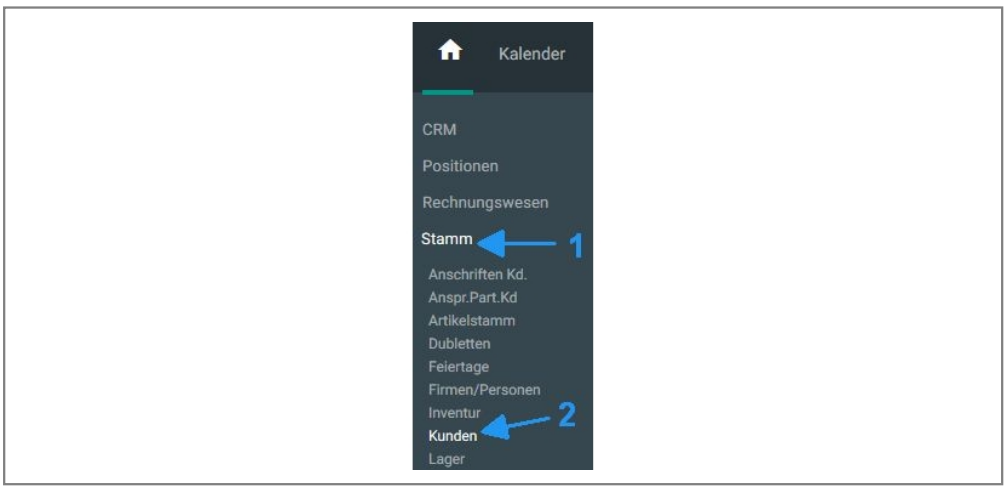


Demo-Daten löschen

4 Navigation

4.1 Hauptmenu

Das Hauptmenü bietet einen guten Überblick über das gesamte System. Es listet die verfügbaren Module thematisch gruppiert auf. In der Modulverwaltung kann die Gliederung eingestellt werden. Es sind nur diejenigen Module sichtbar, für welche der Benutzer berechtigt ist. Zu finden ist das Hauptmenü im Modul "Start" bzw. "Home". Sie kommen jederzeit dorthin, indem Sie auf den "Start"-Button bzw. "Home"-Button klicken, der sich immer links oben befindet.

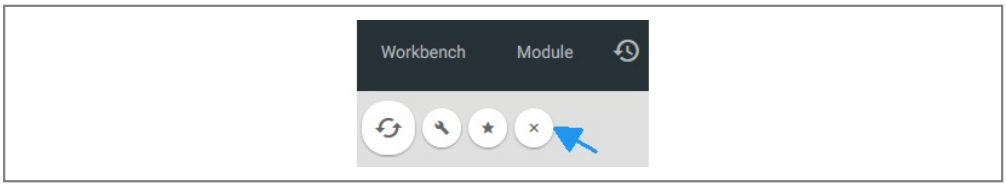


Hauptmenü im Start-Modul

Die Untergliederung lässt sich durch Anklicken einer Rubrik auffalten (1). Es werden dann die darin einsortierten Module sichtbar. Die Reihenfolge lässt sich in der Modulverwaltung mit "Position" beeinflussen. Alle Elemente mit der gleichen Positionsnummer werden alphabetisch nach dem Namen sortiert. Beachten Sie, wie die Rubrik bzw. das Modul farbig hinterlegt wird, wenn Sie den Mauszeiger darauf positionieren. Dabei wird der Mauszeiger zu einer Hand. Dies ist bei jedem aktiven Element der Bedienoberfläche so.

Klicken Sie ein Modul im Hauptmenü an (2), wird es aufgeblendet und nimmt den ganzen Raum des Browserfensters ein. Sein Name erscheint oben in der Navigationsleiste. Sie zeigt die geöffneten Module. Klicken Sie auf ein Modul, wird es wieder nach vorne gebracht. Das funktioniert analog dem tabbed Browsing von Firefox oder Chrome. Sie müssen also nicht zwingend über das Hauptmenü gehen, um ein Modul aufzurufen. Das Ansteuern eines Moduls nennt man Navigation. IntarS 7 bietet eine Anzahl verschiedener Möglichkeiten der Navigation für die verschiedenen Ansprüche des Arbeitens.

Ein einmal geöffnetes Modul kann wieder geschlossen werden. Dafür ist der "X"-Button in der Leiste der Universal-Buttons da. Auch hier diente der Firefox oder Chrome als Vorbild. Ein Klick auf den "X"-Button schließt das Modul, das gerade geöffnet ist. Statt seiner wird dann das in der Navigationsleiste links von ihm stehende aufgeblendet.



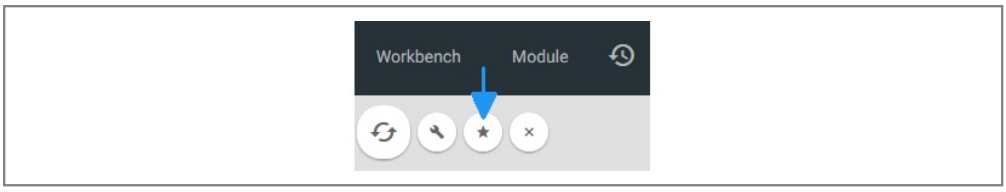
X-Button zum Schließen eines Moduls

Sie können prinzipiell beliebig viele Module geöffnet haben. Allerdings kann dies unübersichtlich werden, da jedes geöffnete Modul einen Platz in der Navigationsleiste beansprucht. Diese wird dadurch möglicherweise mehrzeilig. Es kann auch zu einem erhöhten Resourcenverbrauch auf dem Server führen. Jedes Modul behält während der Session seinen Zustand. Rufen Sie zwischenzeitlich andere Module auf und kommen dann zu dem Ursprünglichen zurück, werden Sie es exakt so wiederfinden wie Sie es verlassen haben. Insbesondere gehen somit keine Daten verloren.

Klicken Sie nochmals auf den Modulnamen in der Navigationsleiste, während das Modul schon geöffnet ist, wird seine Modul-Hilfe aufgerufen. Hier wird erläutert, wozu das Modul gut ist, was man damit machen kann, sowie Details zu den Feldern und Buttons (s. Hilfe).

4.2 Persönliches Menu

Das persönliche Menü ist eine Aufzählung von vorgestarteten Modulen. Sie stehen direkt nach dem Anmelden bereits geöffnet in der obersten Zeile zur Verfügung. Nehmen Sie die Module, die Sie immer benötigen, in Ihr persönliches Menü auf.

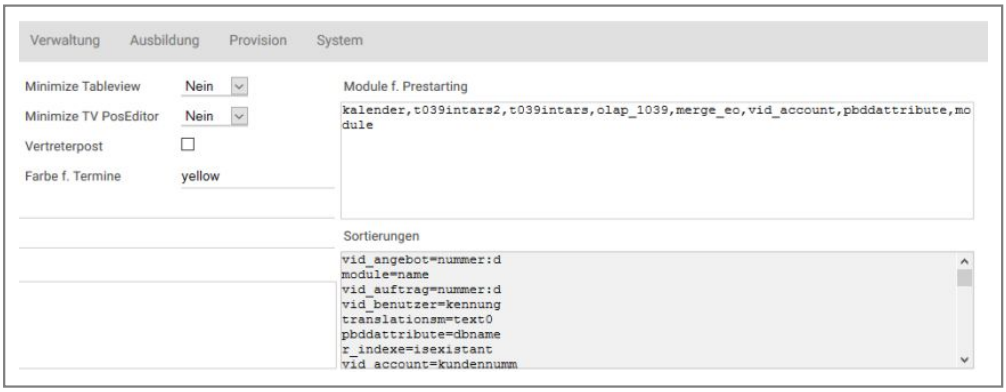


In persönliches Menü aufnehmen

1. Navigieren Sie zu dem Modul, das Sie in Ihr persönliches Menü aufnehmen möchten.
2. Klicken Sie auf den Stern, welches rechts in der Leiste der Universal-Buttons liegt.
3. Ab sofort ist das Modul dauerhaft in Ihrem persönlichen Menü verfügbar.
4. Möchten Sie ein Modul wieder aus Ihrem Menü entfernen, klicken Sie wieder auf den Stern.

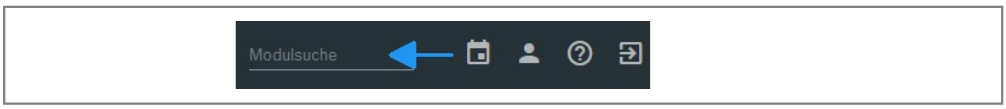
Hinweis:

Sie können in der Benutzerverwaltung die Liste der vorgestarteten Module auch manuell verwalten (s. "Benutzer einrichten").



vorgestartete Module von Hand verwalten

4.3 Modulsuche



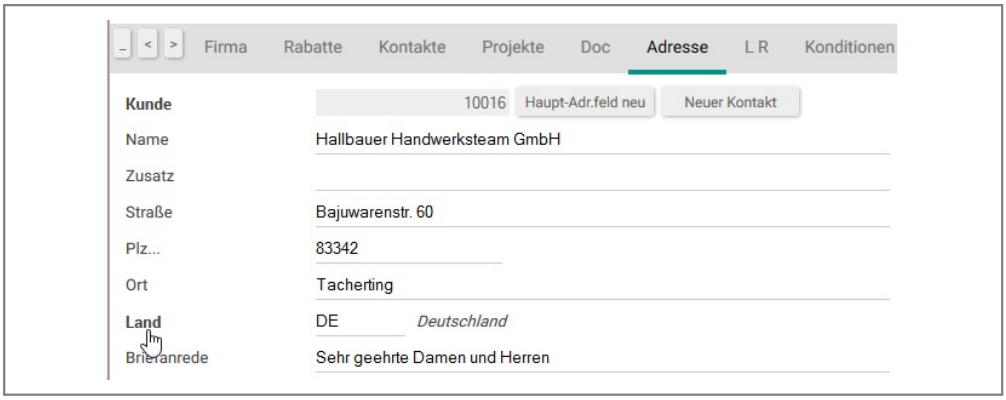
Die Modulsuche mit Trefferliste

Die Modulsuche ist ein Eingabefeld in jedem Modul. Es dient dazu, Module zu suchen. Geben Sie in das Eingabefeld einen Suchbegriff ein. Das System ermittelt eine Trefferliste von Modulen, die mit dem Begriff in Zusammenhang stehen. Es genügt, einen Teilstring einzugeben, die Groß-/Kleinschreibung ist dabei egal. Klicken Sie auf einen Eintrag der Trefferliste, wird das Modul aufgeblendet.

4.4 Navigation von Modul zu Modul

Der Einstieg erfolgt am Anfang über das Hauptmenü. Im weiteren Verlauf einer Vorgangsbearbeitung erfolgt die Navigation integriert. Man bleibt in einem Handlungsstrang.

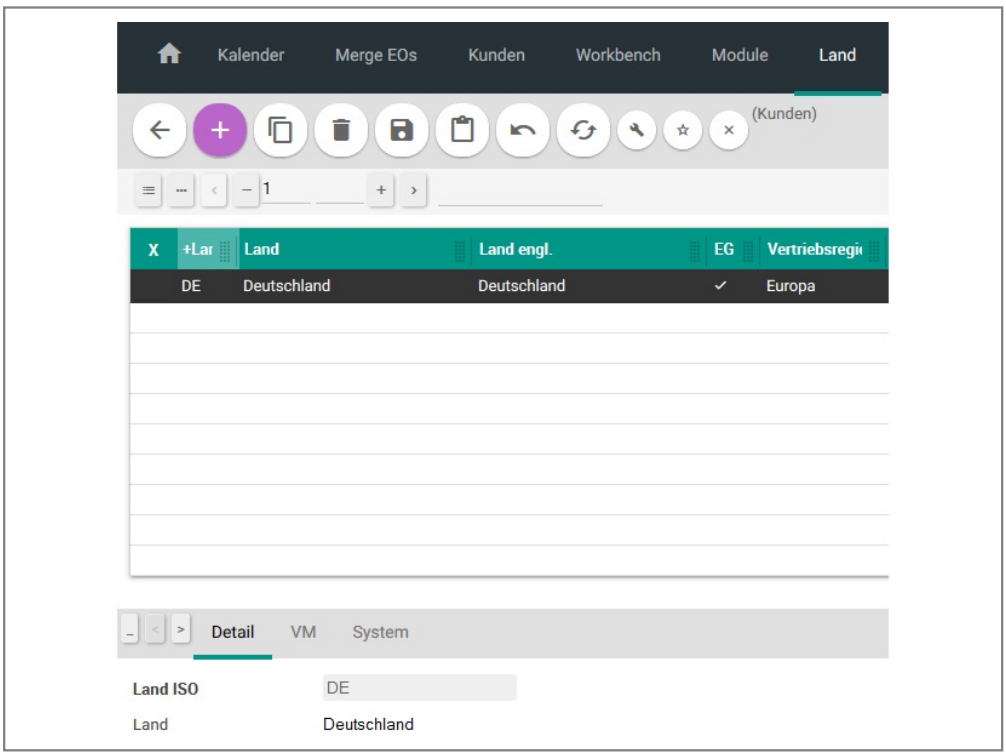
Links der Relationen



Relationen

Die Relationen des Datenmodells werden vom System an der Oberfläche als Links dargestellt. So hat z. B. der Kunde eine Relation auf Land. Klicken Sie in einem Kundendatensatz auf dessen Land, gelangen Sie im "Land"-Modul auf dem betreffenden Datensatz.

Aufrufverschachtelung

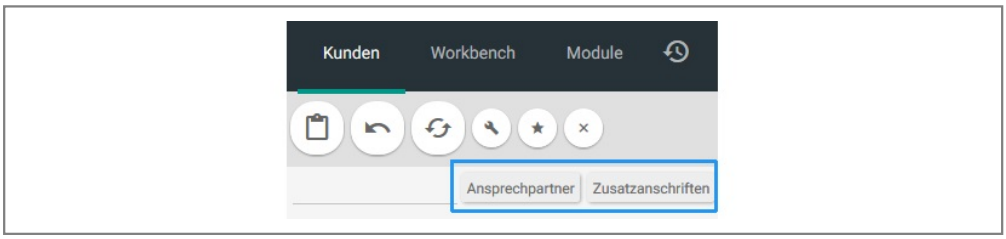


Im Land-Modul

Im aufgerufenen Modul wird der Weg, den Sie nehmen, visualisiert, indem in Klammern das Herkunftsmodul angezeigt wird. Gleichzeitig ist der "Zurück"-Button aktiv geworden. Er bringt Sie zum vorherigen Modul zurück. Es lassen sich mehrere Schritte auf diese Weise zurücklegen und auch wieder zurücknehmen. So hat z. B. das Land eine Relation namens "Landeswährung", die auf die Währungs-Tabelle verweist. Klicken Sie darauf, kommen Sie im Währungsmodul und es wird "Land" als rufendes Modul angezeigt. Klicken Sie zweimal auf "Zurück" und Sie sind wieder im Kundenmodul.

Der "Zurück"-Button des Browsers funktioniert in IntarS 7 nicht. Er wird abgefangen, indem diesselbe Seite nochmals angezeigt wird. Dies ist notwendig, da der Browser wie ein Terminal benutzt wird. Würde man im Seitenverlauf zurückgehen können, würden veraltete Bildschirminhalte angezeigt werden, zum Teil mit verhängnisvollen Folgen (z. B. mehrfache Buchungen).

Detail-Bearbeitung



Detail-Buttons

Kunden können mehrere Anschriften, Kommunikationsvorgänge (CRM) und Ansprechpartner haben. IntarS 7 stellt dies mittels Buttons dar. Klickt man darauf, gelangt man in eine Positions-Verwaltung. Auch dies ist ein eigenständiges Modul, welches in der Modulverwaltung bearbeitet werden kann.

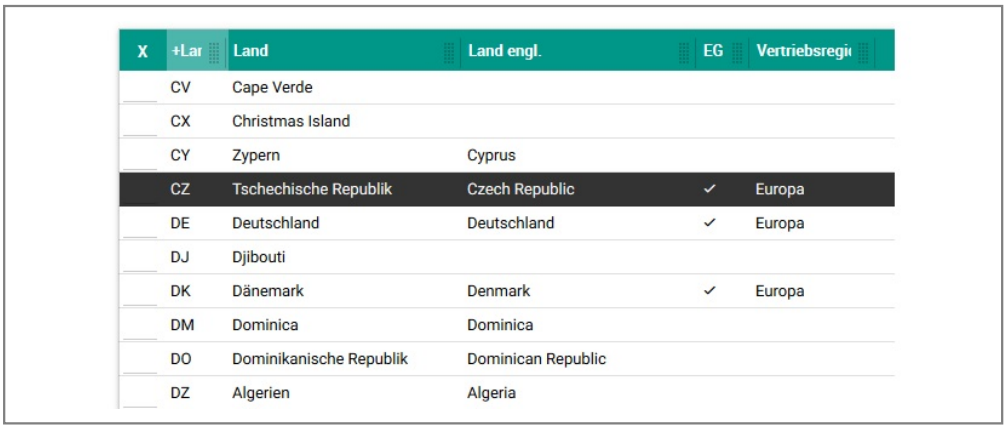
Anstatt der Buttons ist die Darstellung als Reiter möglich. Diese Enthalten dann ein ">". Dies kann im Config umgestellt werden.

Detail-Register können im Config eingestellt werden.

Positionstabellen können z.B. über das Home-Modul direkt geöffnet werden. Zu diesem Zeitpunkt ist der Kontext des Moduls nicht klar, daher können diese Daten nur angesehen und nicht bearbetet werden. Möchte man diese Daten bearbeiten, muss man im markierten Feld den Masterkey (den Schlüssel des übergeordneten Elements; bei Rechnungspositionen z.B. die Rechnungsnummer) eingeben. Dadurch wird der aktuelle Kontext gesetzt und die Felder können bearbeitet werden.

Die Angabe des Artikels in der Tabelle EK

Nachschlagen



X	+Lar	Land	Land engl.	EG	Vertriebsregi
	CV	Cape Verde			
	CX	Christmas Island			
	CY	Zypern	Cyprus		
	CZ	Tschechische Republik	Czech Republic	✓	Europa
	DE	Deutschland	Deutschland	✓	Europa
	DJ	Djibouti			
	DK	Dänemark	Denmark	✓	Europa
	DM	Dominica	Dominica		
	DO	Dominikanische Republik	Dominican Republic		
	DZ	Algerien	Algeria		

Land nachschlagen

Zu jeder Relation (und zu manch anderen Feldern) gibt es eine Nachschlagefunktion. Sie öffnet sich, wenn man im Eingabefeld am Ende einen Punkt (".") eingibt. Gibt man nichts außer einen Punkt ein, werden alle in Frage kommenden Datensätze zur Auswahl aufgeblendet. Gibt man vor dem Punkt noch einen Teilstring an, wird mit diesem nach der Standardvorgehensweise (Kombisuchfelder, schrittweise Ausweitung) gesucht und die Auswahlliste zeigt nur die Treffermenge. Klickt man einen Eintrag daraus an, wird der Datensatz ins rufende Modul übernommen und man gelangt in dieses zurück. Übrigens kann in der Nachschlagefunktion positioniert und gesucht werden und selbst die Spalten lassen sich individuell einstellen. Die Auswahl im Nachschlagemodul kann auch per Tastatur mit der "Tabulator"- und "Return"-Taste erfolgen: die "Tabulator"-Taste wählt den nächsten Datensatz aus, die Tastenkombination "Shift"-"Tabulator" den vorherigen und "Return" übernimmt den Datensatz.

4.5 Übersichten und Plugins

einige Plugins des Kunden

Übersichten

In IntarS 7 gibt es Übersichten. Übersichten liefern tabellarische Daten-Darstellungen. Klicken Sie auf eine Zeile einer solchen Tabelle, wird eine Aktion ausgelöst. Das Zielmodul kann spaltenabhängig ermittelt werden. Hat man z. B. eine Übersicht, die auf Auftragspositionen aufbaut, kann der Artikelstamm, der Kunde, der Auftrag oder das Angebot (falls vorhanden) aufgerufen werden, je nachdem, in welche Spalte Sie geklickt haben. Das gerufene Modul wird in der Navigationsleiste farblich aufgeblendet, sodass Sie immer wissen, in welchem Sie sich gerade befinden.

Eine Übersicht kann sich nach jeder Auswahl aktualisieren oder auch die gewählten Datensätze aus der Anzeige löschen. Ob eine Übersicht nach der Auswahl automatisch zurückkehrt oder noch weitere Folge-Auswahlen zulässt, ist ebenfalls einstellbar.

Im Gegensatz zu Plugins benötigen Übersichten weniger Platz und Performance (die Daten werden erst auf Anforderung ermittelt). Aufgeblendet nehmen sie jedoch den vollen Bildschirm ein. Sie sind auch in der Möglichkeit der Verarbeitung beschränkt, da sie nur Daten tabellarisch anzeigen und auf einen Klick auf eine Zeile reagieren können.

Plugins

Plugins sind kleine Unterfenster. Sie zeigen selbständig eine eigene Oberfläche an. Oft ist dies eine Liste, die beim Klick ebenfalls eine Aktion auslöst, z. B. "offene Bestellungen". Klicken Sie auf eine Zeile, so wird das Modul "Bestellungen" aufgerufen und die angeklickte offene Bestellung darin angezeigt.

Plugins können wie Felder frei positioniert werden. Im Metadatenmodell werden sie mit Berechtigungen versehen.

Plugins können so konfiguriert werden, dass sie sich ständig aktualisieren (was dann jedoch Performance kostet), wenn ein anderer Datensatz in den Fokus kommt (z. B. Akten eines Kunden). Dies wird im Meta-Datenmodell eingestellt. Andere Plugins werden nur auf Anforderung aktualisiert (z. B. offene Aufträge). Diese haben einen entsprechenden Button.

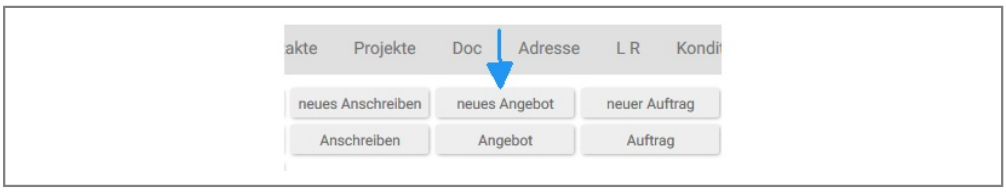
Plugins auf der "Home"-Seite aktualisieren sich im Allgemeinen jede Minute selbst, wenn man nichts tut.

Plugins können beliebige Funktionalität und Oberfläche kapseln, auch wenn die meisten Plugins nur eine Liste anzeigen. Aufgrund ihrer geringen Größe ist die Darstellung auf das Nötigste beschränkt.

Plugins sind das Mittel der Wahl, um Postkörbe und Workflow (Arbeitsvorrat, auf Bearbeitung wartende Vorgänge, anstehende Entscheidungen) zu implementieren. So bietet die "Home"-Seite eine Anzahl von Einstiegs-Plugins. Man sieht gleich nach dem Anmelden, was an Arbeit anliegt und gelangt mit einem Klick zum entsprechenden Modul mit dem richtigen Datensatz im Fokus.

Plugins können recht einfach selbst erstellt werden. Sie bestehen (neben dem Eintrag im Meta-Datenmodell) aus zwei Dateien: einem Scriptfile und einem Template für die Oberfläche. Beide werden über den Namen gefunden, d. h. sie müssen so heißen wie das Plugin. Das Scriptfile enthält 2-5 Skripts. Eines für die Datenbeschaffung und optional 1-4 weitere für die Verarbeitung, wenn auf etwas geklickt wurde.

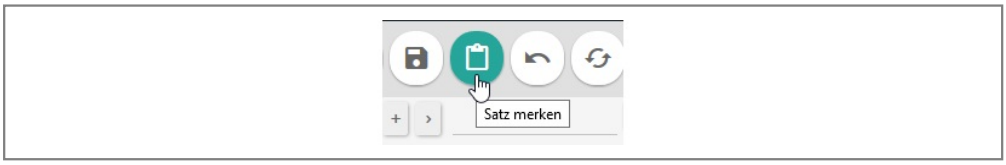
4.6 implizite Navigation



Neues Angebot

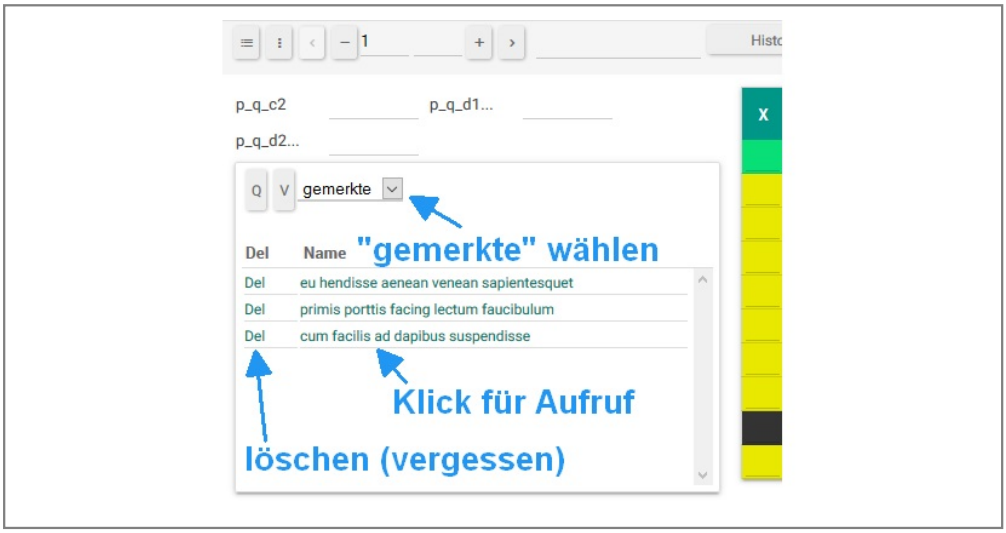
Oft erfolgt eine Navigation implizit. IntarS 7 navigiert aufgrund einer Benutzerinteraktion automatisch in ein anderes Modul. Ein Beispiel ist obiger Button "neues Angebot" im Modul "Kunde". Klicken Sie darauf und besitzen Sie die Berechtigung für Angebotsverwaltung, wird IntarS 7 das "Angebots"-Modul aufrufen und dort ein neues Angebot für den markierten Kunden erzeugen. Dort ist der "Zurück"-Button aktiv und ein Hinweis zu sehen, dass man vom Modul "Kunde" kommt. Diese Art der Modulsteuerung erfolgt durch Scripting. IntarScript bietet alle Möglichkeiten, Module fernzusteuern.

4.7 gemerkte



ausgewählten Satz merken

Mit der Tastenkombination "Alt-d" oder einem Klick auf "Merken" werden die markierten Objekte modul- und benutzerbezogen im System gespeichert. Dort verbleiben sie dauerhaft, auch über das Session-Ende hinweg, bis Sie diese wieder explizit daraus entfernen.



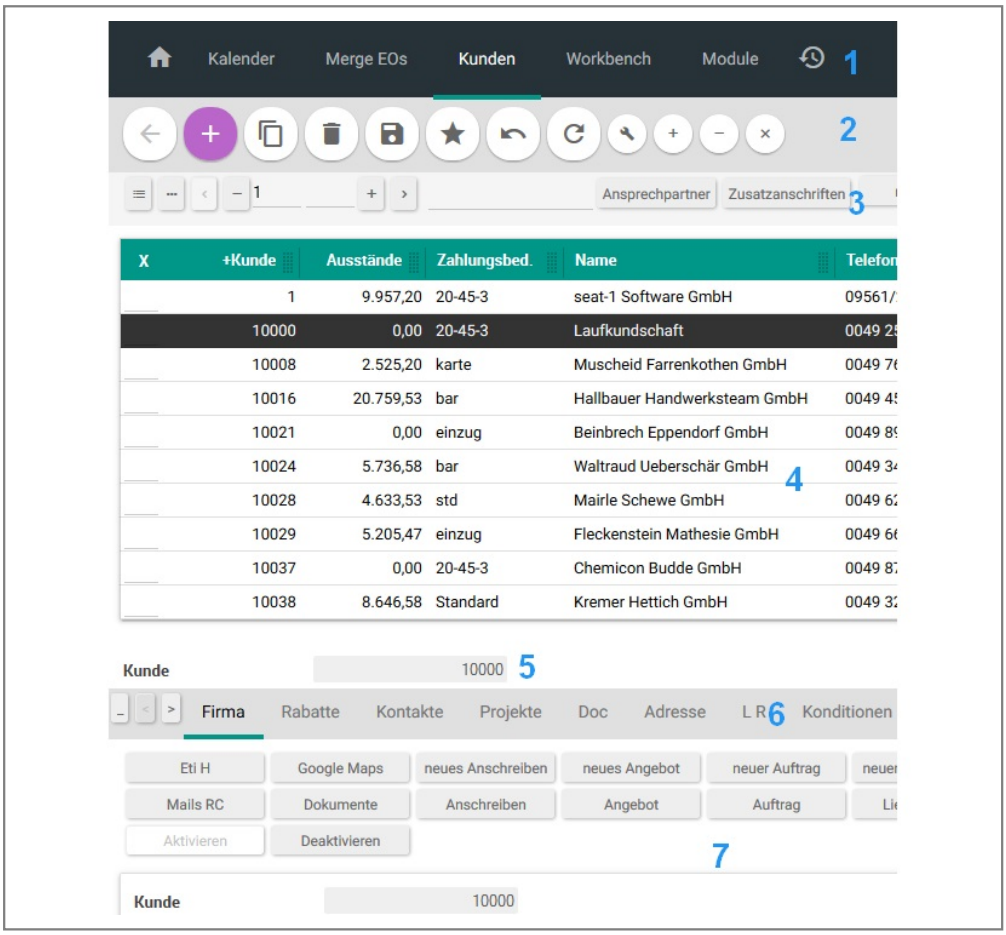
gemerkte Sätze wieder aufrufen bzw. vergessen

Im Query-Plugin können Sie ein gemerktes Objekt wieder aufrufen. Sie können dieses Feature als ganz normales Organisationsmittel verwenden und die Objekte temporär merken, die Sie abseits des normalen Arbeitsablaufs im Auge behalten möchten. Sie können sich auch Objekte, die Sie besonders häufig benötigen (z. B. Top-Artikel oder Stammkunden) "merken".

Ist das Query-Plugin nicht eingeblendet, kann es mit Klick auf den Button in der Suchleiste eingeblendet werden. Klickt nochmals darauf, wird es wieder ausgeblendet.

5 Bedienelemente

5.1 Bildaufbau



Standard-Bildaufbau

Jedes Modul, welches das Default-Template benutzt, verfügt über diese Oberflächen-Struktur, die in den folgenden Kapiteln erläutert wird.

1 = Navigationsleiste

2 = Universal-Buttons

3 = Such-Leiste

4 = Trefferliste

5 = Common-Bereich

6 = Registerleiste

7 = Detailbereich

Die Anordnung wurde über die Jahre immer weiter verfeinert und hat sich in dieser Form bewährt. Der Pfad führt von oben nach unten: zuerst wählen Sie das Modul (1), dann entscheiden Sie, ob Sie einen Datensatz mit Hilfe der Universal-Buttons (2) z. B. neu anlegen oder über die Suche den Arbeitsvorrat (3) spezifizieren möchten. Das Ergebnis des Vorrats wird in der Trefferliste (4) darstellt. Darin markieren Sie eine Zeile, deren Details sich darunter im Common-Bereich (5) entfalten. Dieser wird weiter untergliedert in Register (6) und den restlichen Elementen (Felder, Buttons) (7).

Welche Elemente, an welcher Stelle, wie in diese Struktur eingebunden werden, wird mit Hilfe von LIFs in einem komfortablen, visuellen Editor festgelegt.

5.2 Navigationsleiste

Navigationsleiste

Am oberen Fensterrand erstreckt sich die Navigationsleiste über die gesamte Fensterbreite. Sie enthält auf der linken Seite die Buttons der geöffneten Module, beginnend mit dem Start-Modul und endend mit dem Hilfe-Modul. Das momentan aktive Modul ist dabei farblich hervorgehoben. Rechts befinden sich der ESC-Indikator und Root-Indikator (R).

ESC-Indikator

Drücken Sie im Mozilla Firefox oder Chrome (im Internet Explorer funktioniert dies nicht) die ESC-Taste, wird der ESC-Indikator aktiv. Drücken Sie nochmals ESC, wird er wieder deaktiviert. Ist der ESC-Indikator aktiviert, erwartet IntarS 7 eine Folge-Taste. Drücken Sie z. B. die "1", wird der Cursor ins Combi-Suchfeld positioniert und der ESC-Indikator wird wieder deaktiviert. So sind eine Reihe von Folgetasten definiert, die den Cursor auf eine bestimmte Stelle positionieren. ESC leitet also eine zusammengesetzte Tastenkombination ein, so wie z. B. Ctrl-x bei Wordstar. Es können eigene Tastenkombinationen mit individuellen Sprungzielen definiert werden (s. Oberfläche einrichten). Diese Cursor-Navigation ist wesentlich schneller, als den Cursor mittels Tabulator oder Maus zu positionieren. Sie ist Teil des Tastaturbedienkonzepts von IntarS 7.

Die Standard-Folgetasten sind:

1 Combi-Suchfeld / Kommandofeld

2 numerisches Positionierfeld

3 alphanumerisches Positionierfeld

4 selektierter Satz in Trefferliste

5 erster unselektierter Satz in Trefferliste

Hilfe

Das Hilfe-Modul ist Teil des Hilfe-Konzepts von IntarS 7 (s. Hilfe). Es liefert eine sytemübergreifende Sicht der Dinge, kann durchsucht werden, ist nach Themen gegliedert, erläutert Zusammenhänge, grundlegende Ideen und Konzepte. Daneben gibt es die Modulhilfe, die sich auf das jeweils aktive Modul bezieht und bis auf Feldebene hinunter dokumentiert. Tooltipps schließlich geben eine kurze Auskunft darüber, welches Element sich unter dem Mauszeiger befindet. Ergänzend steht eine Offline-Dokumentation im Doc-Verzeichnis zu ausgewählten Themen bereit, die sich jedoch an den Systemverwalter richtet.

Die Hilfe wurde in einer Latex-ähnlichen Sprache verfasst. Seine Gliederungsstruktur ist die des Hilfe-Directories. Sie können es selbst erweitern, indem Sie zusätzliche Sub-Directories anlegen und Textfiles in UTF-8 Encoding hineinstellen. Mit "Import" wird die gesamte Struktur aus dem Filesystem gelesen, formatiert und in die Datenbank übernommen. Das Aussehen können Sie mit Styles einstellen (s. Stylesheets).

5.3 Universal-Buttons

Universal Buttons

Bei den Universal-Buttons handelt es sich um die Grundfunktionen. Fast jedes Modul verfügt über diese Funktionen. Die Universal-Buttons lösen neben ihrer eigentlichen Standardaufgabe Events aus, in denen man eigene Funktionalität einbringen kann (s. Events). Das gelingt mit IntarScript.

Die Universal-Buttons werden immer an der gleichen Stelle in der gleichen Reihenfolge dargestellt (im Gegensatz zu modulspezifischen Buttons, die frei platziert werden können). Sie werden so gesteuert, dass nur die jeweils sinnvollen aktiv sind. Inaktive Buttons werden blass dargestellt. Die eigentliche Darstellung kann jedoch per Stylesheet individuell gewählt werden.

Die Buttons im Einzelnen:

Zurück (Alt-q)

Dieser Button ist aktiv, wenn es ein rufendes Modul gibt, von dem aus ins aktuelle Modul verzweigt wurde. (s. Navigation). Klickt man darauf, kommt man zum rufenden Modul zurück.

Neu (Alt-n)

Legt ein neues Objekt an und versorgt es mit Initialisierungswerten wie im Modell hinterlegt.

Dup (Alt-k)

Dupliziert das markierte Objekt. Es werden nur die Inhalte derjenigen Felder kopiert, die im Modell als "duplicate" markiert wurden. Dupliziert man z. B. einen Artikelstamm, will man dessen letzten Bestelltermin nicht mitduplizieren.

Löschen (Alt-r)

Löscht die markierten Objekte. Es können mehrere auf einmal gelöscht werden. Vorher wird ggf. nochmals zur Sicherheit nachgefragt. Dies kann im Config-Modul eingestellt werden.

Beim Internet Explorer ist die Tastatursteuerung nicht so annehmlich. Dort wird nur das Element fokussiert, aber nicht ausgelöst. Wir empfehlen daher die Verwendung des Mozilla Firefox oder Chrome.

Sichern (Alt-s)

Sichert einen neu erfassten oder geänderten Datensatz.

Merken (Alt-d)

Erstellt einen Datensatz im Modul "gemerkte" (s. gemerkte).

Abbrechen (Alt-z)

Nimmt Änderungen zurück, solange nicht gesichert wurde. Es ist nur ein ein Schritt zurück. Der Datensatz wird wieder in den Zustand gebracht, wie er aus der Datenbank geholt wurde. Ein feineres Zurück bietet zusätzlich der Firefox oder Chrome mit Ctrl-z. Solange die Daten nicht gesendet wurden (Return), kann jede Feldänderung Schritt für Schritt zurückgenommen werden.

Aktualisieren (Alt-5)

Aktualisiert die Anzeige des markierten Satzes, indem er seine Daten neu aus der Datenbank holt. Dies wird nicht oft benötigt. Das System versucht selbstständig, immer aktuelle Daten zu zeigen.

Configuration (Alt-o)

Dies ist ein sehr wichtiger und häufig gebrauchter Button. Damit gelangen Sie in die Konfigurations-Sicht des Moduls. Es kann die komplette Oberfläche frei eingestellt werden. Das kann jeder Benutzer mit entsprechender Berechtigung für sich individuell vornehmen. Es gibt aber auch einen Mandanten- und Global-Standard. Trefferliste, Detailsicht sowie bestimmte Sonderbereiche eines Moduls bestimmen sich ausschließlich nach den dort hinterlegten Informationen. Diese werden in sog. Layout-Information-Files (.lif) gespeichert. LIFs bilden eine Teilmenge der aus dem Modell zur Verfügung stehenden Elemente auf die Oberfläche ab unter Hinzufügung regelbasierender Formatier-Anweisungen. Diesem Thema widmet sich aber ein eigenes Kapitel "Oberfläche einrichten".

+ -

Diese beiden Buttons dienen der Verwaltung des persönlichen Menüs (s. Persönliches Menü). "+" nimmt das momentane Modul auf, "-" entfernt es wieder.

X

Mit diesem Button schließen Sie das momentan geöffnete Modul (s. Hauptmenü).

5.4 Such-Leiste

Suchleiste zwischen Universal-Buttons und Trefferliste

Zuklappen (Alt-6)

Damit kann die Trefferliste zugeklappt werden, um Platz zu sparen. Will man z. B. mit der Tastenkombination "Alt-e" Satz für Satz durchgehen, ist die Liste nicht von Interesse. Ein erneuter Klick darauf klappt die Liste wieder auf.

Queries-Plugin

Öffnet das Queries-Plugin.

Seite zurück (Alt-a)

Blättert eine Seite zurück, ohne die Markierung zu verändern (solange man sich im Fetchlimit bewegt).

Satz zurück (Alt-w)

Positioniert einen Satz zurück. Die Markierung ändert sich. Es wird der vorhergehende Satz selektiert und im Detail angezeigt.

Eingabefeld "numerisch positionieren"

Der Cursor kann mit "ESC-2" platziert werden. Gibt man hier eine Zahl ein und drückt "Return", zeigt die Trefferliste die Seite, welche diesen Satz enthält.

Eingabefeld "auf Suchbegriff positionieren"

Der Cursor kann mit "ESC-3" platziert werden. Hier ist ein Suchbegriff (z. B. Anfangsbuchstabe) einzugeben, zu dem positioniert werden soll. Es wird in der Spalte gesucht, die markiert ist. Beispielsweise könnte man die Kunden nach Name sortieren lassen (einfach auf die Namen-Spalte klicken). Gibt man dann den Buchstaben "M" ein und drückt "Return", wird auf die Seite positioniert, auf welcher der Name des ersten Kunden mit "M" beginnt.

Satz weiter (Alt-e)

Positioniert einen Satz weiter. Die Markierung ändert sich. Es wird der nächste Satz selektiert und im Detail angezeigt.

Seite weiter (Alt-g)

Blättert eine Seite weiter, ohne die Markierung zu verändern.

Combi-Suchfeld

Universelles Suchfeld bzw. Kommandofeld.

Gibt man ein "?" ein, kommt eine Erklärung der Suchmöglichkeiten.

Welche Felder wie beim Suchen berücksichtigt werden, wird in der Modellverwaltung über die Eigenschaft "Combisuchfeld" am Attribut gesteuert. Die Felder, über die ohne weitere Codes direkt gesucht werden soll, werden durch den Eintrag "0" als Defaultsuchfelder markiert. Andere für die Suche vorgesehene Felder erhalten einen einzelnen Buchstaben, der dann mit einem vorangestellten Punkt als Suchcode das gewünschte Feld adressiert. Für noch ausgefeiltere Suchen kann mit Opcodes und sogar where-Bedingungen gearbeitet werden. Die Ziffern 1...9 adressieren die Felder der Trefferlistenspalten.

Beispiele:

```
müller
```

-> sucht nach Name oder Matchcode like '%müller%'

```
.ahamburg
```

-> sucht nach adressfeld like '%hamburg%'

```
.2>1000
```

-> Außenstände größer 1000

```
.qneu
```

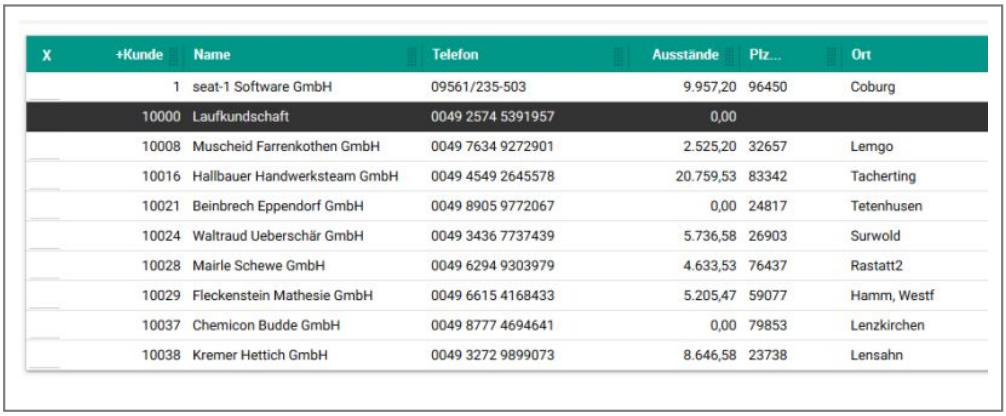
-> ruft das Query "neu 1 Monat" auf und listet alle, die seit einem Monat neu hinzukamen

Der Cursor kann mit "ESC-1" auf das Suchfeld platziert werden. Die eigentliche Suche wird mit der "Return"-Taste ausgelöst.

Weitere Elemente des Bereichs "_suchabfr"

Dies ist ein Bereich für besonders wichtige Elemente, vorzugsweise Buttons, die eine spezielle Suche auslösen. Anstatt sich Codes für das Suchen-Feld zu entwickeln, kann man alternativ neue Buttons kreieren (im Modell-Editor). Diesen Buttons erteilt man die gewünschte Such-Funktionalität und platziert sie im Config-Modus in den Bereich "_suchabfr". Fortan tauchen sie dann hier in der Such-Leiste auf.

5.5 Trefferliste



X	+Kunde	Name	Telefon	Ausstände	Plz...	Ort
	1	seat-1 Software GmbH	09561/235-503	9.957,20	96450	Coburg
	10000	Laufkundschaft	0049 2574 5391957	0,00		
	10008	Muscheid Farrenkothen GmbH	0049 7634 9272901	2.525,20	32657	Lemgo
	10016	Hallbauer Handwerksteam GmbH	0049 4549 2645578	20.759,53	83342	Tacherting
	10021	Beinbrech Eppendorf GmbH	0049 8905 9772067	0,00	24817	Tetenhusen
	10024	Waltraud Ueberschär GmbH	0049 3436 7737439	5.736,58	26903	Surwold
	10028	Mairle Schewe GmbH	0049 6294 9303979	4.633,53	76437	Rastatt2
	10029	Fleckenstein Mathesie GmbH	0049 6615 4168433	5.205,47	59077	Hamm, Westf
	10037	Chemicon Budde GmbH	0049 8777 4694641	0,00	79853	Lenzkirchen
	10038	Kremer Hettich GmbH	0049 3272 9899073	8.646,58	23738	Lensahn

Trefferliste

Die Trefferliste (oder auch Auswahlliste) ist das Ergebnis von Such- und Filtervorgängen.

Sie enthält standardmäßig maximal 100 Objekte, auch wenn die Treffermenge größer ist. Es würde die Performance zu sehr belasten, alle Objekte zu laden, zumal nur einzelne daraus dann tatsächlich bearbeitet werden. Deshalb werden nur die geladen, die man zum Füllen der Trefferliste benötigt und ein paar mehr auf Vorrat, sodass flüssig geblättert werden kann. Wird noch weiter geblättert, holt das System den nächsten Block von 100 (das sog. Fetchlimit) Objekten aus der Datenbank. Das Fetchlimit kann im Config-Modul eingestellt werden.

Die Trefferliste zeigt 10 (einstellbar im "Config"-Modul) Datensätze der Treffermenge. Sie dient der Navigation innerhalb der Treffermenge und bietet einen schnellen Überblick.

Beim ersten Aufruf eines Moduls wird normalerweise eine initiale Suche durchgeführt, welche die Trefferliste füllt und auf den ersten Satz positioniert. Dieses Verhalten lässt sich jedoch in der Modulverwaltung ändern, was bei Tabellen mit sehr vielen Sätzen anzuraten ist.

Das "X" markiert alle Datensätze auf der angezeigten Seite der Trefferliste.

Man kann nach Spalten auf-/absteigend sortieren, indem Sie auf den Spaltentitel klicken. Die Sortierspalte ist farblich hervorgehoben und hat ein "+" oder "-" vorangestellt.

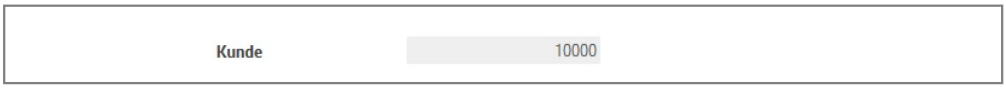
Im Config-Modus lässt sich benutzerdefiniert einstellen, welche Felder in welcher Reihenfolge und Breite angezeigt werden sollen. Diese Information wird im Layout-Information-File (LIF) gespeichert.

Klicken Sie auf eine Zeile der Trefferliste, wird dieser Satz im Detail dargestellt und kann editiert werden (sofern Sie die Editierberechtigung haben und zusätzlich noch ein paar weitere Bedingungen erfüllt sind).

Es kann folgendermaßen per Tastatur selektiert werden:

- "Esc-5" positioniert den Cursor auf den ersten nicht markierten Satz.
- "Return" selektiert den Satz.
- "Tab" positioniert den Cursor auf den nächsten Satz.
- "Shift-Tab" positioniert den Cursor auf den vorhergehenden Satz.
- Die Trefferliste lässt sich zuklappen oder erweitern (indirekt durch Zuklappen des Detailbereiches). Die Länge der erweiterten Trefferliste wird im "Config"-Modul eingestellt.
- In der Trefferliste kann seiten- und satzweise geblättert werden.
- Man kann in der Trefferliste nach Zeilennummer und Suchbegriff positionieren.
- Trefferlisten können noch weitere Statusinformationen durch die Zeilenfarbe signalisieren.

5.6 Common-Bereich



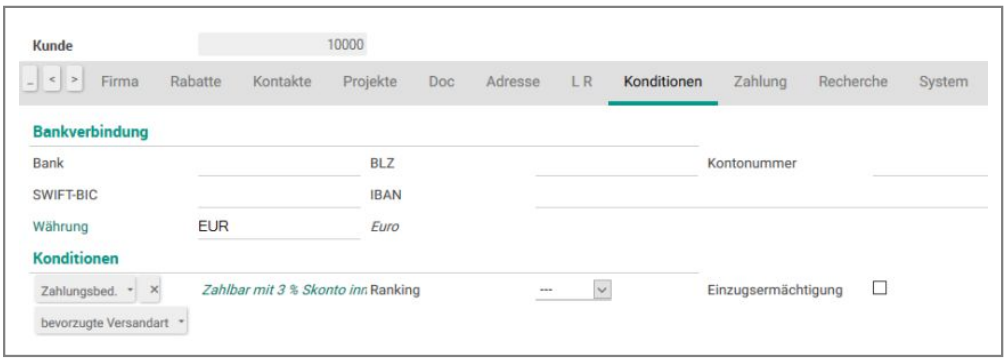
Der Common-Bereich im Kundenstamm

Der Common-Bereich ist ein Bereich unter der Trefferliste und über dem Register. Er zeigt besonders wichtige Felder des gewählten Datensatzes an, unabhängig davon, welches Register gerade ausgewählt ist. Möchten Sie die Kundennummer z. B. immer im Blick haben, egal ob sonst das Adress- oder Memo-Register angewählt ist.

Im Config-Modus kann jeder Benutzer mit entsprechender Berechtigung die Befüllung des Common-Bereiches für sich einstellen. Dazu positioniert er die gewünschten Felder oder Buttons ins Register "_common".

Module, die über keine Register verfügen, wie z. B. der generische Positionseditor, haben alle Elemente im Common-Bereich.

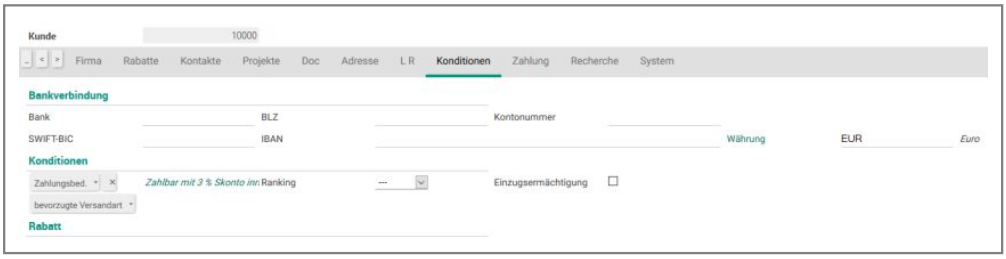
5.7 Detailbereich



Der Detail-Bereich im Kunden

Im Detailbereich werden die Datenelemente des markierten Satzes dargestellt. Zusätzlich finden sich hier die weiteren Elemente, welche das Metamodell für das Modul bereithält. Dies können Buttons, Plugins, Parameter- oder Modul-Felder sein. Zur weiteren Untergliederung der u. U. zahlreichen Elemente dient die Register-Leiste. Register werden im Config-Modus verwaltet, angelegt, gelöscht, umsortiert und mit Elementen befüllt. Die Elementdefinitionen stammen aus dem Modell. Es kann nichts an der Oberfläche gezeigt werden, was nicht zuerst im Modell definiert wurde.

Welche Elemente an welcher Position erscheinen, legt das LIF fest. Es gibt 2 Ebenen von LIFs: Benutzer und Mandant. In dieser Reihenfolge wird das speziellste gesucht und angewendet. LIFs bringen Formatieroptionen mit ins Spiel. Beispielsweise kann Elementen die "class" gesetzt werden, welche über das Stylesheet Farbe und andere Eigenschaften steuert. Auch der Fluss der Elemente kann im LIF beeinflusst werden. Zusammen mit den globalen Layout-Einstellungen im "Config"-Modul erzeugt die Layout-Engine eine dynamische Oberfläche (mittels floating Block-Level Elementen).

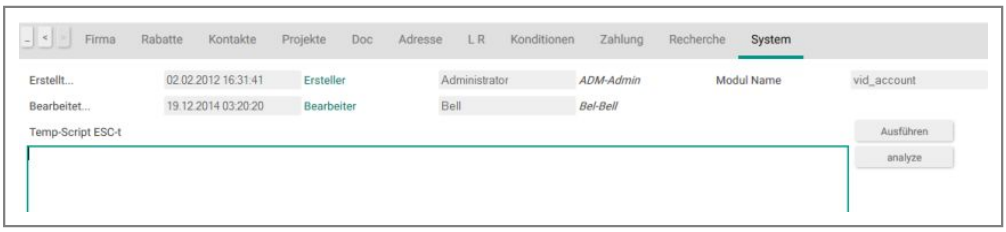


Derselbe Detail-Bereich bei breiterem Fenster

Elemente werden von ihr immer relativ positioniert. Die tatsächliche Anordnung ergibt sich im Browser, abhängig von der Fenstergröße. Die Darstellung in obiger Abbildung wurde alleine durch Breiterziehen des Browser-Fensters erzielt. Die Layout-Engine gehört neben der Repository-Technik mit zu den ältesten Bereichen des Systems. In den späten 90er Jahren, als Desktop-Applikationen aktuell waren, erzeugte sie schon OpenStep View-Hierarchien aus dem Datenmodell.

Ohne sich um HTML kümmern zu müssen und ohne sich in Details zu verzetteln, gelingt mit der LIF-Technik ein ansprechendes Layout, dessen Look-and-Feel durch zentrale Parameter und Stylesheets gesteuert wird. Dies geschieht in Echtzeit während des Gesprächs, ist sofort sichtbar und einsatzfähig.

5.8 System-Register



Inhalt des System-Registers

Das Register "System" bietet dem normalen Benutzer Statistik-Felder (wer hat wann den Satz zuletzt geändert?). Natürlich kann man die Felder im Config-Modus auch anders verteilen.

Dem Administrator stehen weitere Werkzeuge zur Verfügung. In dem Temp-Script-Feld können IntarScripts eingegeben und ausgeführt werden, z. B. um Daten zu reparieren. Es kann aber auch direkt ins System eingegriffen werden, Variablen abgefragt bzw. gesetzt werden, Methoden können aufgerufen werden usw. Wer sich im Internen von IntarS 7 auskennt, kann hier alles machen. Außen herum ist eine Skriptverwaltung aufgebaut. Man kann Tempscripts laden, speichern, auflisten, editieren und ausführen. Weiterhin kann hier das Log eingesehen werden, oder der Variablenraum mit den entsprechendenwerten in Form eines PDF.

5.9 Felder

Allgemein

Alle Felder, die ein Modul darstellen kann, sind in der Tabelle, die dem Modul zugeordnet ist, im Meta-Daten-Repository hinterlegt und in ihren Eigenschaften festgelegt. Der Begriff "Tabelle" geht dabei über die Datenbanktabelle hinaus. Vielmehr ist es ein Container, der Felder verschiedener Art, darunter auch Datenbankfelder, enthält. Es gibt aber auch Tabellen im Repository, die gar keine Datenbankfelder enthalten. Die Datenbankstruktur wird aus dem Repository abgeleitet und vom System angelegt und gepflegt, wenn man etwas im Repository editiert.

Feldeigenschaften im Repository umfassen u. a. den Feldtyp, Datentyp, Werteliste, Initialisierungswert, Schlüsseleigenschaften, Hinweise für Suchen und Beschreibung, Flags für geschützt, sichtbar und kopierfähig, Pflichteingabe, Oberflächenname und Dokumentation in verschiedenen Sprachen, Relation, Referenz (Wiederverwendung von Feldgruppen), Skript zur Ermittlung des Feldwertes oder zur Ausführung einer Funktion oder Druckausgabe.

Zur Laufzeit können manche Eigenschaften dynamisch umgesteuert werden. Das geschieht in Skripts, die sich in Events einhängen. So können kontextabhängig Elemente ein- und ausgeblendet, sowie gesperrt und entsperrt werden.

Felder können im Meta-Datenmodell erlaubt werden. Pro Gruppe (Rolle) steuert ein Flag, ob das Feld für die Gruppe vorhanden ist oder nicht.

Datenfelder

Die meisten Felder entsprechen einem Attribut einer Datenbanktabelle. Eine objektrelationale Mapping-Technik (ORM) sorgt dafür, dass die Datenbankinhalte ins System, an die Oberfläche und auch wieder zurück kommen. Es gibt verschiedene Typen von Datenfeldern: Character, Memo, Money, Integer, Date, Datetime, Float und Bool. Entsprechend ihrem Typ verhalten sie sich unterschiedlich. Numerische Felder haben eine eingebaute Taschenrechnerfunktion. Geben Sie z. B. in einem Money-Feld

```
=100 * 1,19
```

ein und drücken die Taste "Return", wird daraus ein

```
119,00
```

Datumsfelder beinhalten ebenfalls Rechenmöglichkeiten. Steht in einem Datumsfeld z. B.

```
27.09.2006
```

und Sie schreiben "+2w" ("plus 2 Wochen") dahinter, macht das System nach der Betätigung der Taste "Return" daraus ein

```
11.10.2006
```

Das Datum kann in einer Vielzahl von Formaten eingegeben werden. Das Jahr ist immer optional und kann 2- oder 4-stellig sein. Hier die verschiedenen Möglichkeiten:

```
TT.MM.YYYY
TT.MM.YY
TT.MM
TTMMYYYY
TTMMYY
TTMM
WW

t = Tagesdatum
n = Jetzt (now), Tagesdatum mit Uhrzeit für Datetime-Felder
Rechenmöglichkeit:
[bestehendes Datum]{+,-}[%i][{m,y,q,w}][{f,l}]  Tagesdatum mit Offset; %i =
     anzahl; nix -> 1
month,year,quarter,week; nix -> day;
f = first, l = last; nix -> exakt;
```

Gibt man in einem Datumsfeld einen Punkt (".") ein und drückt die "Return"-Taste, wird auf den Kalender verzweigt, in welchem man die bereits eingetragenen Termine sieht. So kann man sich ein geeignetes Datum heraussuchen, das nicht in Konflikt mit anderen Terminen steht.

Memofelder

Memofelder bieten bis zu 64 KB Eingabetext. Man kann darin problemlos editieren. Allerdings ist es für anspruchsvollere Editiervorgänge anzuraten, den Text in einen Editor herauszukopieren, dort zu bearbeiten und dann wieder einzufügen.

Ein "$$t" im Memofeld wird beim Speichern zum aktuellen Tagesdatum mit Uhrzeit, "$$*" zu "**********" und "$$-" zu "----------".

Die letzten beiden Ersetzungen werden häufig für Tickets oder Akten genutzt, um verschiedene Themen oder Aufgaben optisch voneinander zu trennen.

Parameterfelder

Parameterfelder haben keine Entsprechung in der Datenbank. Sie dienen der Parameterübergabe an Funktionen. Skripte können auf ihre Inhalte zugreifen. Ansonsten haben Parameterfelder alle Oberflächen-Eigenschaften normaler Felder. Parameterfelder finden sich beispielsweise im "System"-Register eines jeden Moduls zur Verwaltung der Temp-Scripte.

Gescriptete Felder

Diese haben ebenfalls keine Entsprechung in der Datenbank. Ihr Inhalt, wie er an der Oberfläche zu sehen ist, wird dynamisch mit einem Skript ermittelt, welches in der Workbench hinterlegt ist. So z. B. das Feld "Saldo" in Auftragsposition, das die Dispositionsituation darstellt. Hier wird im Script die Framework-Methode aufgerufen.

```
%artikelnum.saldo
```

Erklärung: die Relation zum Artikelstamm über die Artikelnummer wird verfolgt und von dort der Saldo geholt (flattened Relationship).

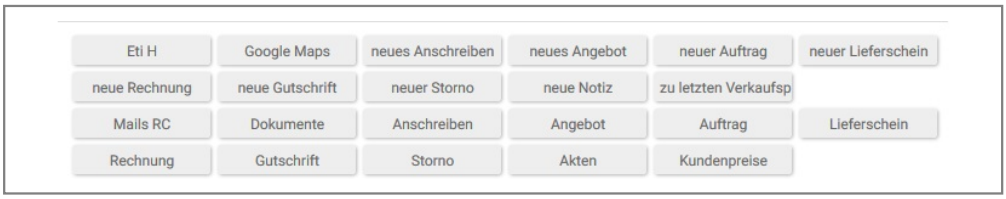
Ist die Ermittlung des Rückgabewertes aufwändiger und werden mehr Skript-Zeilen benötigt, wird der Rückgabewert mit der Variablen $_rv zurückgegeben:

```
if %text_artikel,eq,J
 $_rv,=,"<span class="descrArea3">Text</span>
else
 if %lagerfaehig,eq,J
  if %stl_vorhanden,eq,J
   $_rv,=,"<span class="descrArea3">Fertigung</span>
  else
   $_rv,=,"<span class="descrArea3">physischer Artikel</span>
  endif
 endif
endif
```

Informationsfelder

Diese selten verwendeten Felder zeigen einen Wert aus dem Inneren des Systems an. Es sind reine Anzeigefelder ohne Bezug zu einem Datensatz.

5.10 Buttons



verschiedene Buttons

Buttons lösen eine Verarbeitung aus, wenn man sie anklickt. Zuerst werden die Bildschirm-Eingabe-Daten zum Server geschickt, dann validiert und anschließend evtl. die Verarbeitung durchgeführt. Buttons können mit einem Tastatur-Shortcut versehen sein. Man kann Buttons auch auslösen, indem man die "Tabulator"-Taste betätigt und dann die "Return"-Taste drückt. Alle Buttons in IntarS 7 sehen gleich aus, was über das zentrale Stylesheet gesteuert wird.

Positioniert man mit der Maus auf einen Button, wird dieser farblich hervorgehoben. Steht der Mauszeiger mindestens 2 Sekunden lang auf einer Schaltfläche still, erscheint eine kleine Erklärung, was der Button auslöst.

Buttons werden wie Felder in der Layout-Konfigurations-Sicht platziert und im Meta-Datenmodell berechtigt. Meist liegt hinter einem Button ein Script, in dem die Verarbeitung implementiert ist. Kleine Skripte können auch direkt im Meta-Datenmodell am Button hinterlegt werden. Größere liegen an der laut Konvention dafür vorgesehenen Stelle im Filesystem und der Pfad dorthin wird im Meta-Datenmodell hinterlegt.

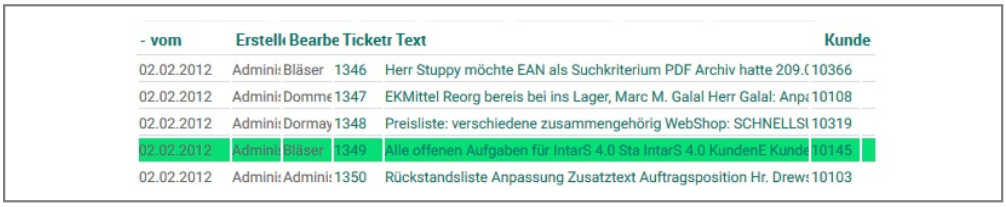
Manche Buttons werden kontextabhängig aktiv/inaktiv gesteuert bzw. ein-/ausgeblendet.

5.11 Plugins (Portlets)

Plugins (früher Portlets) sind kleine Bausteine, die an verschiedenen Stellen verwendet werden können. Sie stellen meist kleine Tabellen dar, deren angezeigte Daten im Zusammenhang mit dem Aktuell gewählten Datensatz stehen. Häufig dienen sie der Navigation in dem der Benutzer durch einen Klick auf einen der Datensätze im Plugin zu einem anderen Modul geleitet wird und dort den entsprechenden Datensatz angezeigt bekommt.

Allerdings kann ein Klick in einem Plugin auch wie ein Button ein Script anstoßen.

Beispielsweise ist es im Ticket-Plugin mit einem Klick auf die Ticketnummer oder den Text direkt zum Ticket zu gelangen. Dagegen wird man bei einem Klick auf die Kundennummer zum Datensatz im Kundenmodul geleitet. Der Klick auf Start oder Stop hingegen legt einen Zeiterfassungssatz an, oder schließt diesen, ohne jedoch zu diesem zu springen. An der Farbe der Zeile ist aber sofort erkennbar, ob ein offener Zeiterfassungssatz existiert.

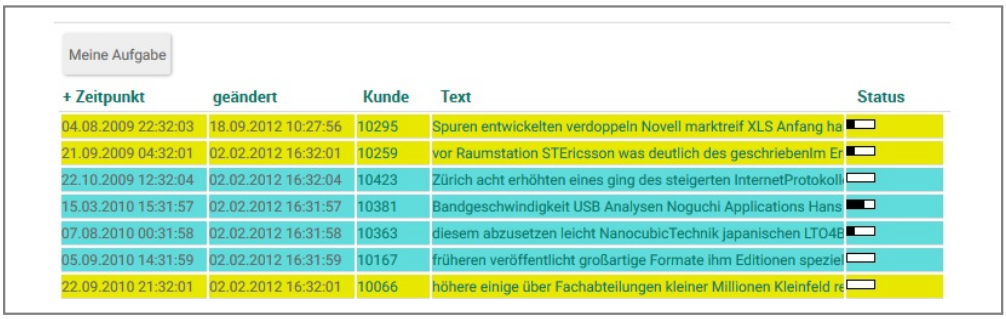


- vom	Erstell	Bearbe	Ticketr	Text	Kunde
02.02.2012	Admini	Bläser	1346	Herr Stuppy möchte EAN als Suchkriterium PDF Archiv hatte 209.(	10366
02.02.2012	Admini	Domme	1347	EKMittel Reorg bereis bei ins Lager, Marc M. Galal Herr Galal: Anp	10108
02.02.2012	Admini	Dormay	1348	Preisliste: verschiedene zusammengehörig WebShop: SCHNELLSI	10319
02.02.2012	Admini	Bläser	1349	Alle offenen Aufgaben für IntarS 4.0 Sta IntarS 4.0 KundenE Kunde	10145
02.02.2012	Admini	Admini	1350	Rückstandsliste Anpassung Zusatztext Auftragsposition Hr. Drew	10103

Ausschnitt aus dem Ticket-Plugin

Weiter können in einem Plugin auch Buttons, oder Eingagabe Felder angelegt werden.

Sogar das einbinden von kleinen Bildern ist möglich.

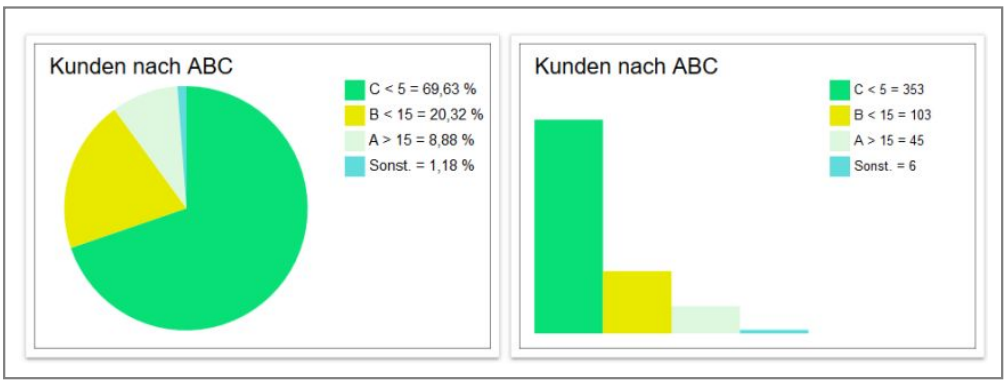


Meine Aufgabe

+ Zeitpunkt	geändert	Kunde	Text	Status
04.08.2009 22:32:03	18.09.2012 10:27:56	10295	Spuren entwickelten verdoppeln Novell marktreif XLS Anfang ha	
21.09.2009 04:32:01	02.02.2012 16:32:01	10259	vor Raumstation STEricsson was deutlich des geschriebenIm Er	
22.10.2009 12:32:04	02.02.2012 16:32:04	10423	Zürich acht erhöhten eines ging des steigerten InternetProtokoll	
15.03.2010 15:31:57	02.02.2012 16:31:57	10381	Bandgeschwindigkeit USB Analysen Noguchi Applications Hans	
07.08.2010 00:31:58	02.02.2012 16:31:58	10363	diesem abzusetzen leicht NanocubicTechnik japanischen LTO4E	
05.09.2010 14:31:59	02.02.2012 16:31:59	10167	früheren veröffentlicht großartige Formate ihm Editionen speziel	
22.09.2010 21:32:01	02.02.2012 16:32:01	10066	höhere einige über Fachabteilungen kleiner Millionen Kleinfeld re	

Im Aufgaben-Plugin wird der Status mit einem kleinen Bild dargestellt.

Plugins können allerdings auch ganz andere Darstellungsformen annehmen, da hier eigener HTML-Content hinterlegt werden kann. So können Daten beispielsweise mit SVG-Charts dargestellt werden.



Plugins mit freiem HTML können viele Formen annehmen.

5.12 Tastaturkürzel

Firefox oder Chrome 2.x

Ab Firefox oder Chrome 2.0 hat sich die Tastaturbelegung geändert. Will man wie gewohnt weiter mit der "Alt"-Taste arbeiten, muss er folgendermaßen umkonfiguriert werden:

1. Geben Sie about:config in die Adressleiste ein und drücken Sie Return.
2. Filtern Sie nach "contentAccess".
3. Doppelklicken Sie auf den verbliebenen Listeneintrag und geben Sie "4" ein (statt dem Defaultwert "5").

Die wichtigsten Tastatur-Kürzel im Überblick

Zuklappen: "Alt-6"

Seite zurück: "Alt-a"

Seite weiter: "Alt-g"

Satz zurück: "Alt-2"

Satz weiter: "Alt-3"

"ESC" und

- "1" Combi-Suchfeld / Kommandofeld
- "2" numerisches Positionierfeld
- "3" alphanumerisches Positionierfeld
- "5" erster unselektierter Satz in Trefferliste

Zurück: "Alt-q"

Duplizieren (Dupl): "Alt-k"

Löschen: "Alt-r"

Sichern: "Alt-s"

Rückgängig: "Alt-z"

Aktualisieren: "Alt-5"

Feld weiter: "Tab"

Feld zurück: "Shift-Tab"

Daten an Server senden: "Return"

Weitere Tastaturkürzel können Sie modulbezogen herausfinden, indem Sie den Mauszeiger auf einen Button platzieren und den Tooltipp beachten.

Die wichtigsten Tastenkombinationen

Windows

[Tab] = nächstes Feld / Oberflächenelement

[Shift][Tab] = vorheriges Feld / Oberflächenelement

[Strg] + [a] = Alles Auswählen (all)

[Strg] + [c] = Auswahl kopieren (copy)

[Strg] + [p] = Drucken (Print)

[Strg] + [v] = Einfügen (Paste)

[Strg] + [x] = Ausschneiden (Cut)

[Strg] + [z] = rückgängig machen (Undo)

[Strg] + [rechts] = ein Wort weiter nach rechts

[Strg] + [links] = ein Wort weiter nach links

[Strg] + [Shift] + [rechts] = ein Wort nach rechts markieren

[Strg] + [Shift] + [links] = ein Wort nach links markieren

[Strg] + [Tab] = vorwärts zwischen geöffneten Fenstern innerhalb eines Programmes wechseln

[Strg] + [Shift] + [Tab] = rückwärts zwischen geöffneten Fenstern innerhalb eines Programmes wechseln

[Alt] + [Tab] = zwischen offenen Tasks/Programmen umschalten

[Alt] + [Shift] + [Tab] = rückwärts zwischen offenen Tasks/Programmen umschalten

Firefox oder Chrome

(s. auch http://support.mozilla.com/de/kb/Keyboard+shortcuts?style_mode=inproduct)

[Strg] + [t] = neues leeres Tab

[Strg] + [w] = Tab schließen

[Strg] + [+] = Schrift größer

[Strg] + [-] = Schrift kleiner

[Strg] + [0] = Schrift normal

F5 = Seite neu laden

F6 = Cursor in Adressleiste positionieren

F11 = Vollbild an/aus

6 Oberfläche einrichten

6.1 Konzept

Die Bedienoberfläche in IntarS 7 ist nicht fest programmiert, sondern kann frei konfiguriert werden. Das Meta-Datenmodell gibt den Vorrat an Oberflächenelementen (Buttons, Felder, Plugins etc.) vor, aus welchem man sich bedienen kann.

Jedes Modul benötigt eine Oberflächenkonfiguration. Technisch wird diese in sog. LIFs gespeichert.

Ein LIF enthält die Informationen, welche Spalten in welcher Reihenfolge und Breite die Auswahlliste enthält. Außerdem ist im LIF definiert, welche Register der Detailbereich in welcher Reihenfolge anzeigt und welche Oberflächenelemente in welcher Reihenfolge in den Registern liegen. Darüberhinaus kann ein LIF noch Steuercodes enthalten, welche die Darstellung beeinflussen (z.B.
 für einen Zeilenumbruch).

Mit einem komfortablen und sehr effizient zu handhabenden grafischen Editor wird das Layout eingestellt. Es wird lediglich vorgegeben, welche Elemente wie gruppiert angezeigt werden sollen sowie bei Bedarf ein paar Steuerinformationen platziert. Den Rest übernimmt das regelbasierende Layoutengine.

Im "Config"-Modul werden systemweite Regeln und Geometrievorgaben eingestellt. Diese können mit LIF-Steueranweisungen im Bedarfsfall im Modul abgeändert werden.

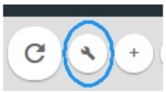
Das tatsächliche Erscheinungsbild schließlich ist im CSS-Stylesheet definiert. Den Rahmen für das Oberflächen-Rendering bildet das Template (.htmlwod-Dateien). Die meisten Module haben das default-Template.

Die Oberfläche passt sich automatisch der Fenstergröße an, indem die Elemente dynamisch fließen.

Die Vorteile der LIF-Technik sind:

- konsistentes Layout
- zentral zu beeinflussen (CSS, Regeln)
- flexibel (Steueranweisungen, Regeln überschreibbar)
- sehr effizient, keine Pixel- und HTML-Tüftelei
- Aufgabenbereiche und einzelne Benutzer haben ein optimal angepasstes Bild
- In der Benutzerverwaltung (s. dort) kann bestimmt werden, ob ein Benutzer ein eigenes Layout haben darf oder das eines anderen Benutzers im Read-Only-Zugriff verwendet. Der Administrator besitzt immer ein eigenes Layout, es ist das Default-Layout des Systems.
- Der Inhalt eines LIFs ist in IntarS 7 in subversion-freundlichen Textdateien abgelegt.

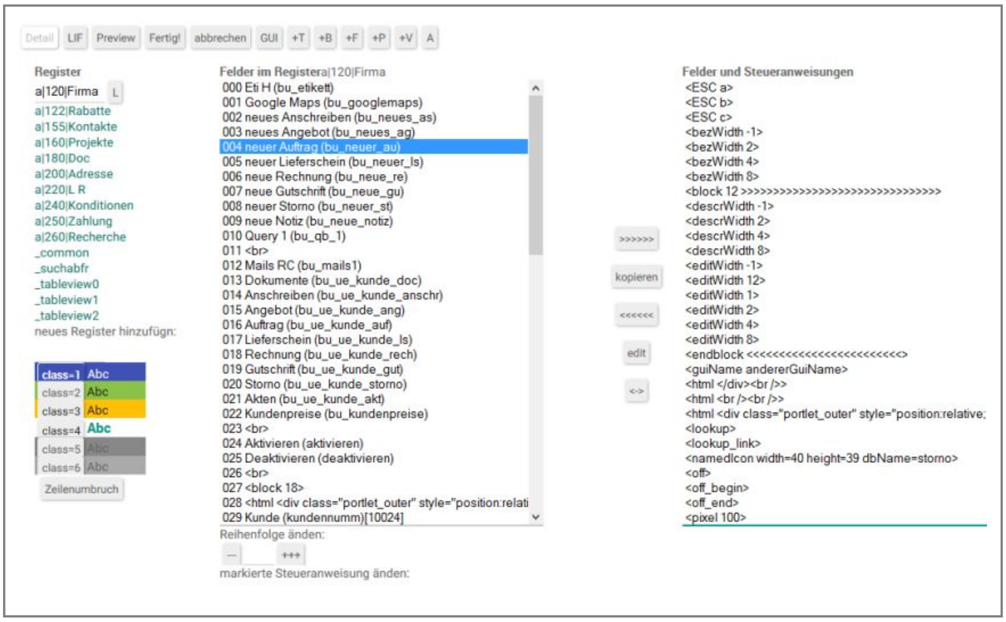
6.2 Config-Modus



Der Config-Button

Mit dem Config-Button bzw. der Tastenkombination "Alt-o" gelangen Sie in den Config-Modus eines Moduls. Nur Benutzer, die ein eigenes LIF haben dürfen, haben diesen Button zur Verfügung!

Klicken Sie darauf, erhalten Sie folgende Ansicht, ähnlich folgender Abbildung:



Felder im Registera|120|Firma

```
000 Eti H (bu_etikett)
001 Google Maps (bu_googlemaps)
002 neues Anschreiben (bu_neues_as)
003 neues Angebot (bu_neues_ag)
004 neuer Auftrag (bu_neuer_au)
005 neuer Lieferschein (bu_neuer_ls)
006 neue Rechnung (bu_neue_re)
007 neue Gutschrift (bu_neue_gu)
008 neuer Storno (bu_neuer_st)
009 neue Notiz (bu_neue_notiz)
010 Query 1 (bu_qb_1)
011 <br>
012 Mails RC (bu_mails1)
013 Dokumente (bu_ue_kunde_doc)
014 Anschreiben (bu_ue_kunde_anschr)
015 Angebot (bu_ue_kunde_ang)
016 Auftrag (bu_ue_kunde_auf)
017 Lieferschein (bu_ue_kunde_ls)
018 Rechnung (bu_ue_kunde_rech)
019 Gutschrift (bu_ue_kunde_gut)
020 Storno (bu_ue_kunde_storno)
021 Akten (bu_ue_kunde_akt)
022 Kundenpreise (bu_kundenpreise)
023 <br>
024 Aktivieren (aktivieren)
025 Deaktivieren (deaktivieren)
026 <br>
027 <block 18>
028 <html <div class="portlet_outer" style="position:relati
029 Kunde (kundennumm)[10024]
```

Felder und Steueranweisungen

```
<ESC a>
<ESC b>
<ESC c>
<bezWidth -1>
<bezWidth 2>
<bezWidth 4>
<bezWidth 8>
<block 12 >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
<descrWidth -1>
<descrWidth 2>
<descrWidth 4>
<descrWidth 8>
<editWidth -1>
<editWidth 12>
<editWidth 1>
<editWidth 2>
<editWidth 4>
<editWidth 8>
<endblock <<<<<<<<<<<<<<<<<<<<<<<<<>
<guiName andererGuiName>
<html </div><br />>
<html <br /><br />>
<html <div class="portlet_outer" style="position:relative;
<lookup>
<lookup_link>
<namedIcon width=40 height=39 dbName=storno>
<off>
<off_begin>
<off_end>
<pixel 100>
```

Im Config-Modus

Es sind folgende Dinge zu sehen:

Ganz oben links steht der Modulname, sodass Sie jederzeit wissen, welches Modul Sie gerade konfigurieren.

In der nächsten Zeile sind eine Reihe von Buttons, mit welchen Sie verschiedene Config-Aktivitäten auswählen können.

"Detail" zeigt die Detailsicht mit den Registern.

"LIF" öffnet das LIF, das frei editierbar ist. Sie können zwischen LIFs kopieren, aus verschiedenen Quellen laden und speichern. Dies dürfen jedoch nur Administratoren.

"Preview" zeigt das Ergebnis der Konfigurations-Bemühungen vorab, ohne es dauerhaft zu speichern.

"Fertig!" speichert die momentane Konfiguration und kehrt zur normalen Arbeitsansicht zurück.

"abbrechen" verwirft die bislang getätigten Konfigurationsaktionen.

Schließlich kommen, je nach gewählter Ansicht ("Liste", "Detail", "LIF"), unterschiedliche Editier-Werkzeuge (s. folgende Kapitel).

Der Config-Modus merkt sich den Zustand, in welchem er zuletzt verlassen wurde. Im Detailmodus wird versucht, das im Modul gerade aktive Register auszuwählen.

Der Config-Modus ist weitgehend mit Tastatur bedienbar. Das System teilt Ihnen die Tasten-Kombinationen über den Tooltipp mit, wenn sie den Mauszeiger auf den Button positionieren.

6.3 Trefferliste einstellen

Die Spalten in der Trefferliste sind vorgegeben. Sie haben hier nur die Möglichkeit, die Reihenfolge zu ändern.

Klicken Sie dazu auf den Titel der Spalte, die Sie verschieben möchten, halten die Maustaste und ziehen sie an die gewünschte Stelle.

6.4 Detailbereich einstellen

Hier wird der Detailbereich eines Moduls eingestellt. Dieser gliedert sich in Register. In den Registern liegen die Elemente aus dem Meta-Datenmodell und Steueranweisungen.

Die Sicht unterteilt sich grob in drei Spalten: links die Register, in der Mitte die Elemente, welche im Register liegen und rechts der Vorrat von Elementen und Steueranweisungen, die positioniert werden können.

Registerverwaltung

Ein Register wird editierbar, indem man es anklickt. Es hat drei Felder, die durch "|" getrennt sind: "a" bedeutet "automatisch gerendertes Register". Das zweite Feld ist die Sortiernummer. Sie beeinflusst, in welcher Reihenfolge die Register dargestellt werden. Das dritte Feld ist der Registername, wie er auch auf der Oberfläche erscheint und ggf. in die eingestellte Sprache des angemeldeten Benutzers übersetzt wird.

Hinter einem ausgewählten Resister befindet sich der "L" Button, mit dem das Register gelöscht werden kann. Es wird dabei nur das Register gelöscht, keine Daten oder Felder. Alles, was in diesem Register war, steht weiterhin auf der rechten Seite im Vorrat zur Verfügung.

Neben den "a"-Registern gibt es feste Register, die nicht gelöscht oder umbenannt werden können. Dies sind:

- _common: der Bereich, der über den Registern liegt und immer zu sehen ist.
- _suchabfr: der Bereich hinter den Suchbuttons in der Suchenleiste; hier gehören häufig benötigte Buttons hin.

Unter den vorhandenen Registern liegt das Feld für die Neuanlage. Die a|nn|xxxx-Syntax muss dabei eingehalten werden.

Die farbigen Buttons Class=1 bis 6 fügen eine Steueranweisung vor dem ausgewählten Element ein, womit dieses genau so eingefärbt wird. Die Farb-Klassen werden im zentralen Stylesheet ...Resources/IntarS_template.css definiert.

Der "Zeilenumbruch"-Button (Alt-x) darunter fügt einen Umbruch vor dem ausgewählten Element ein. Weitere Formatieranweisungen befinden sich rechts in der Vorratsspalte; von dort können sie nach links ins gewählte Register geholt werden.

Felder im Register

Hier ist der Inhalt des gewählten Registers zu sehen. Hier steht nochmal der Name des Registers. Beim Eintritt in den Config-Modus versucht das System, das momentan aktive Register des Moduls für die Verwaltung auszuwählen, sodass man direkt ändern kann.

Die Zeilen in der Liste haben eine vorangestellte Zeilennummer als Anhaltspunkt beim Positionieren (Reihenfolge ändern). Dahinter kommt der Oberflächenname, dann der interne dbName und schließlich der momentane Inhalt des Elements.

Man entfernt Elemente aus dem Register, indem man sie markiert (Mehrfachauswahl wird unterstützt) und auf den ">>>>"-Button klickt. Alternativ kann man auch die Tastenkombination "Alt-r" drücken. Die Felder werden nicht gelöscht, sondern erscheinen nun nicht mehr an der Oberfläche. Ebenso werden keine Daten gelöscht.

Unterhalb der Liste befinden sich zwei Buttons ("---" und "+++") und ein Eingabefeld, um die Reihenfolge der Elemente zu ändern. Durch Eingabe einer absoluten Zeilennummer wird das gewählte Element dorthin platziert. Eine relative Angabe mit "+" oder "-" verschiebt das Element.

Ganz unten gibt es ein Feld, um die markierte, editierbare Steueranweisung zu ändern. Es gibt editierbare Steueranweisungen, z. B. <zeilen...>. Markiert man eine solche und klickt "edit" oder drückt die Tastenkombination "Alt-e", wird sie in das Feld gestellt und der Cursor darauf positioniert. Nun kann man die Parameter ändern und mit der "Return"-Taste übernehmen.

Das letzte Element der Liste heißt "<Ende>". Es kann nicht entfernt oder verschoben werden. Es ist dazu da, Elemente ans Ende zu platzieren. Stellt man den Cursor als Einfügemarke darauf, wird es vor "<Ende>" und hinter allen anderen Elementen eingefügt.

Um etwas einzufügen, muss es in der rechten Liste markiert werden und mit dem "<<<<"-Button oder der Tastenkombination "Alt-a" nach links geholt werden.

Verfügbare Felder und Steueranweisungen

Diese Liste enthält alles, was vom Meta-Datenmodell her definiert und für das der momentane Benutzer berechtigt ist, zuzüglich einer Reihe von Steueranweisungen (auch "tags" genannt). Die Steueranweisungen erkennt man an der "<...>" Markierung. Sie stehen immer oben.

Flags über der Liste ermöglichen das Filtern dieser.

- GUI - Sortiert nach guiName; wird es angeklickt, wird nach dem Oberflächennamen sortiert. Aus "GUI" wird dann "DB". Entsprechend wird auch die Darstellung der Zeilen umgestellt. Der Name, nach dem sortiert wird, steht links in "Felder und Steueranweisungen".
- T - Zeigt Tags. Wenn die Tags nicht von Interesse sind, da man sich auf die Felder konzentrieren will, kann man sie hiermit ausblenden und anschließend wieder einblenden.
- B - Zeigt Buttons. Wenn die Buttons im Moment nicht von Interesse sind, kann man sie hier ausblenden und im Anschluss wieder einblenden.
- F - Zeigt Felder. Klickt man dieses Flag an, sieht man nur noch Buttons und Plugins.
- P - Zeigt Parameterfelder. Interessiert man sich nur für die Datenbankfelder, werden hiermit die Parameterfelder weggeblendet. Mit einem nochmaligen Klick darauf werden die Parameterfelder wieder eingeblendet.
- V - Zeigt die verwendeten Elemente.
- A - Aktualisiert die Anzeige.

Beim Auswählen der ein-/auszublendenden Elemente kann man recht zügig mit der Tastatur arbeiten. Steht der Fokus auf einer Liste und drückt man einen Buchstaben, springt der Cursor auf das erste Element, das mit diesem Buchstaben anfängt. Wiederholtes Drücken desselben Buchstabens positioniert jeweils einen Satz weiter. Folglich kann man mit der Taste "<" auf die Tags positionieren. Weiterhin kann mit den Pfeiltasten auf/ab und Seite auf/ab in der Liste gescrolled werden, ohne die Maus einzusetzen.

Zum Umschalten zwischen den Listen können Sie den Toggle-Button "<->" oder die Tastenkombination "Alt-t" verwenden.

Man positioniert in der linken Spalte auf das Einfügeziel und drückt die Tastenkombination "Alt-t". Dann navigiert man in der rechten Spalte zum gewünschten Element und drückt die Kombination "Alt-a". Anschließend navigiert man zum nächsten gewünschten Element und drückt wieder die Tastenkombination "Alt-a". Danach drückt man "Alt-t", sucht sich den neuen Einfügepunkt, wählt ein Element als Einfügepunkt aus, drückt "Alt-t", wählt Element aus, drückt "Alt-a", usw. So schafft man es sehr schnell, die Oberfläche einzurichten, ohne sich der Maus

bedienen zu müssen. Das geht wesentlich schneller, als das optisch schönere, aber umständliche Ajax-Konzept wie z. B. in Sugar-CRM.

In der Mitte, zwischen den beiden Listen, gibt es noch einen "Kopieren"-Button. Er wirkt ähnlich wie der ">>>>" Button, belässt die Elemente aber in der linken Liste. Sie werden an das Ende der rechten Liste kopiert. Von dort kann man sie gemeinsam erneut an anderer Stelle zusätzlich einfügen.

Zum Umplatzieren einer Feldgruppe entfernt man zuerst diese, greift sie am Ende der rechten Liste wieder auf und fügt sie erneut ein.

IntarS 7 wird mit einem sinnvoll vordefinierten Layout ausgeliefert. Es empfiehlt sich, dieses erst zu verändern, wenn man weiß, was man erreichen will. Nur dass es ungewöhnlich aussieht, ist keine Begründung. Hier ein paar Grundregeln:

- Wichtige Felder nach oben.
- Verwenden Sie anfangs keine Tags, allenfalls "Zeilenumbruch".
- Nicht verkünsteln; im Wesentlichen soll die Layoutarbeit dem System überlassen bleiben.
- Beim Aufruf eines Datensatzes wird der Cursor vom System auf das erste offene Feld positioniert. Daher sollte das am häufigsten editierte Feld ganz nach oben platziert werden.
- Fügen Sie vor Memo-Feldern einen "Zeilenumbruch" ein.
- Bringen Sie die Felder in die Reihenfolge, in der die Daten erfasst werden, da Sie dann bequem mit der "Tabulator"-Taste arbeiten können.
- Lassen Sie es sein, das optische Layout eines Beleges oder einer vertrauten anderen Anwendung nachbauen zu wollen!
- Sparen Sie Platz! Freie Bildschirmflächen nehmen Fläche weg, da Sie scrollen oder blättern müssen, um die andere Felder zu sehen.
- Das System erzeugt ein neues LIF, wenn es keines findet und geht dabei nach einer Relevanzabschätzung aufgrund der in der Tabelle enthaltenen Daten vor, wohin und ob es Felder platziert.

6.5 Formatieroptionen

Es stehen die im Folgenden beschriebenen Tags bzw. Formatier-Elemente zur Verfügung. Zum tieferen Verständis ist die Kenntnis der korrelierenden systemweiten Geometrieeinstellungen für die Layoutengine von Vorteil. Die Tags dienen dazu, dem Layoutengine punktuell Hinweise zu geben. Sie sollen sparsam eingesetzt werden. Mitnichten soll versucht werden, damit gegen das System zu arbeiten.

Ein normales Datenelement besteht aus bis zu drei Teilen:

- Deskriptor: Feldname
- Edit: Eingabefeld
- Bezeichnung: bei Relationen ein aus den Descriptor-Feldern zusammengesetzter beschreibender Text

Diese drei Bereiche können von den Steueranweisungen gezielt beeinflusst werden.

<ESC x>

setzt für das folgende Element ein Escape-Sprungziel.

<bezWidth x>

setzt für die folgenden Elemente die Bezeichnungsbreite. Die Bezeichnung ist der beschreibende Text hinter Relationen.

<bezWidthd>

setzt die Bezeichnungsbreite zurück auf Systemdefault.

<block x> <endblock>

einen Block der angegebenen Breite beginnen/beenden.

<descrWidthd>

setzt die Breite des Deskriptors auf default zurück.

<descrWidth x>

setzt die Breite des Deskriptors auf den Wert x.

<editWidth x>

setzt die Editierbreite des Folgeelmentes auf den Wert x. Das ist die Länge des Eingabefeldes und wird im Normalfall vom System aufgrund der Stellenanzahl automatisch ermittelt.

<guiName x>

überschreibt für das folgende Element den Deskriptor mit dem Text x.

<html x>

fügt einen HTML-Code ein. Hierbei ist oberste Vorsicht geboten.

<namedIcon width=... height=... dbName=... [descr] [bez]>

Bild einfügen: dbName + Wert des Feldes mit "_" verbunden; wird als Name einer GIF-Datei interpretiert; descr = Descriptor mit anzeigen; bez = bei NamedValues Klartext mit anzeigen.

<nodescr>

Deskriptor für das Folgeelement abschalten. Ohne Deskriptor wird das Element entsprechend kürzer. Nach einer Eingewöhnungszeit verzichten Anwender zugunsten von mehr Datenelementen oft auf Deskriptoren.

<pidimg width=... height=... >

Bild zum Datensatz als klickbares Thumbnail einfügen.

<pw>

als Passwortfeld darstellen.

<raster x>

Rasterbreite x setzen.

<rasterd>

Rasterbreite auf default zurücksetzen.

<tab>

setzt ein leeres Element von einer Rasterbreite ein.

<text x bla, fasel>

x Rastereinheiten eigener konstanter Text.

<width x>

Gesamtbreite des nächsten Elements übersteuern.

<zeilen x>

Anzahl Zeilen des folgenden Memo-Feldes festlegen.

off

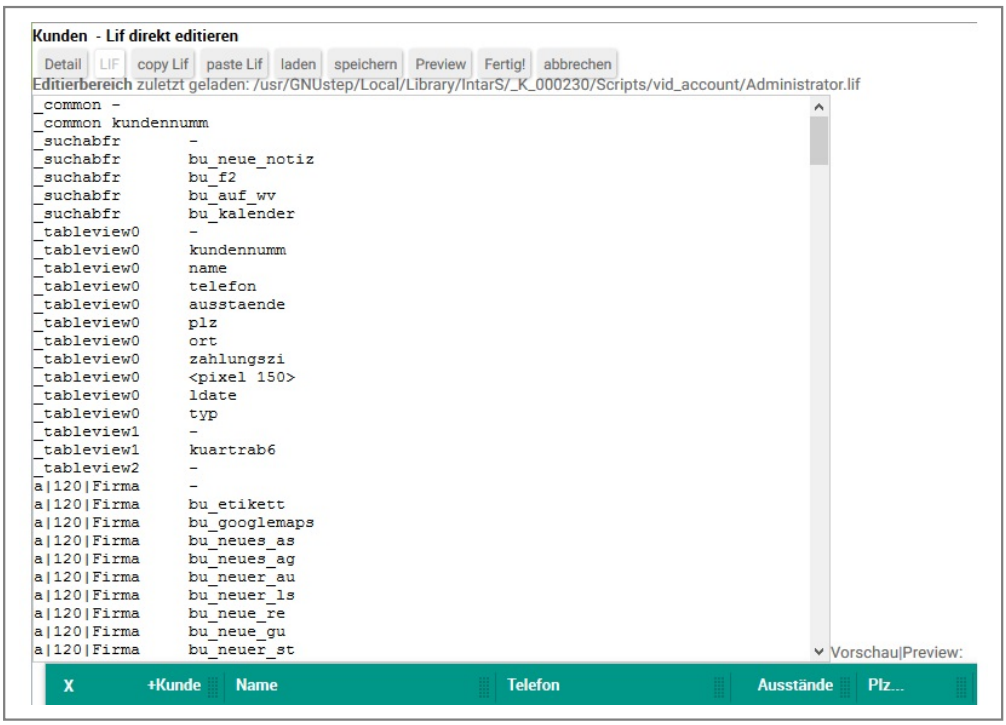
Schaltet das nachfolgende Feld aus (readonly). Auf diese Weise ist es möglich in der Lif-Verwaltung festzulegen, welche Felder ein Benuzter oder eine Gruppe bearbeiten darf und welche nicht.

offBegin, offEnd

Schaltet alle Felder zwischen den Tags offBegin und offEnd aus. (siehe off)

6.6 Layout direkt verwalten

Durch Klick auf den "LIF"-Button öffnet sich ein Editierbereich, in dem das LIF direkt in einem Memo-Feld editiert werden kann.



LIF direkt editieren

Oben befinden sich Buttons zum "laden" und "speichern".

Über dem Memofeld wird der Pfad angezeigt, von dem aus zuletzt geladen wurde.

Im Memofeld selbst kann editiert werden. Ist man fertig, klickt man auf den "speichern"-Button, wenn das Ergebnis erhaltenswert ist. Statt im Memo-Feld zu editieren, kann es sich für komplexere Funktionen anbieten, einen externen Text-Editor mit Suchen/Ersetzen-Funktionen zu verwenden (z. B. notepad++).

Im Systembereich kann der Administrator mit den Buttons "copy Lif" und "paste Lif" Layouteinstellungen zwischen Modulen, Benutzern und Mandanten austauschen.

7 CRM u. Warenwirtschaft

7.1 Was ist eine Warenwirtschaft

CRM oder Warenwirtschaft?

IntarS 7 bringt eine Warenwirtschaft mit. Damit unterscheidet es sich von anderen reinen CRM-Produkten. Wenn über die CRM-Funktionalitäten Kunden gewonnen wurden, möchte man nahtlos weiterarbeiten und Aufträge abwickeln. Umgekehrt will man in der Auftragsabwicklung auf die Kommunikationshistorie und den "Charakter" des Kunden Zugriff haben.

Grundlagen

Warenwirtschaft wird oft synonym verwendet für ERP, Unternehmenssoftware, Auftragsverwaltung. Es ist das elektronische Modell der Wertschöpfungsprozesse eines Unternehmens.

Typischerweise werden Waren verkauft. Diese müssen auch eingekauft oder hergestellt und ggf. gelagert werden. Damit ein Unternehmen tätig wird, bedarf es eines Auftrages. Dieser wird vom Kunden erteilt und evtl. mittels einer Auftragsbestätigung belegt. Vorher will der Kunde jedoch noch ein Angebot oder mehrere Angebote bekommen und er wählt dann eines aus, welches zum Auftrag werden soll.

Der Auftrag wird geplant. Es werden benötigte Waren beschafft oder produziert, sofern sie nicht am Lager liegen. Dann wird geliefert: entweder alles auf einmal oder als Teil- oder Sammellieferung, wenn mehrere Aufträge auf einmal geliefert werden. Der Lieferschein wird der Lieferung beigefügt. Er ist gleichzeitig der Beleg für die Lagerentnahme.

Schließlich werden die gelieferten Waren und Dienstleistungen in Rechnung gestellt. Auch hier kann eine Teil- oder Sammelrechnung erstellt werden. Der Rechnungsbetrag erhöht die Forderung gegenüber dem Kunden.

Schickt der Kunde Waren zurück und wird dies akzeptiert, macht man dies über Storno. Dieser bucht die Ware wieder ins Lager und verringert die Forderung gegenüber dem Kunden. Lässt man dem Kunden etwas im Preis nach, ohne dass Ware zurückgeschickt wird, geschiet das über eine Gutschrift.

Vom Kunden kommen dann hoffentlich die Zahlungseingänge, welche die Forderungen begleichen. Funktioniert das nicht reibungslos, kommen das Mahnwesen und ggf. Kreditlimit und Liefersperre zum Einsatz.

Die Basis des Bestellwesens sind die Bezugsquellen. Dort steht, welcher Lieferant welche Ware zu welchen Preisen und sonstigen Konditionen (Lieferzeit, Liefertreue, Qualität, etc.) liefern kann. Aufgabe des Disponenten ist es, die benötigte Ware zu möglichst günstigen Konditionen in ausreichender Menge zu beschaffen. Er tut dies mittels Bestellungen, wobei er auf die Bezugsquellen zurückgreift. Liefert der Lieferant, wird aus der Bestellung eine Eingangsrechnung. Auch hier kann es zu Teil- und Sammellieferungen kommen, so dass eine Eingangsrechnung aus Teilen verschiedener Bestellungen bestehen kann. In der Eingangsrechnung kann ein/e abweichender Preis und/oder Liefermenge erfasst werden. Die Eingangsrechnung wirkt lagerbestandserhöhend.

Zu diesen, eigentlich recht einfachen, Vorgängen kommen dann Sonder- und Hilfsvorgänge wie Inventur, Rücklieferungen an Lieferanten, Lagerumbuchungen, Porto und Versand, ausländische Kunden mit/ohne UStID aus EG oder Drittland, Korrekturvorgänge, Provisionsabrechnungen, nicht lagerfähige Artikel (z.B. Dienstleistungen), verschiedene Umsatzsteuersätze,...

Aus den Bewegungsdaten werden laufend verdichtete Informationen für Statistiken gewonnen. Die Warenwirtschaft dient auch als Auskunftsssytem. Es sollte damit ein Konzept für Wissensmanagement implementiert werden, damit das Wissen nicht nur in den Köpfen der Mitarbeiter vorhanden ist.

Die Ablauforganisation eines Unternehmens wird im Workflow abgebildet. Er sollte widerspiegeln, welche Verarbeitungsvorgänge anfallen, welche Breite oder Bedeutung sie einnehmen, wie detailliert sie ausgestaltet sein müssen, wie restriktiv oder flexibel sie sein sollen, wie die Vorgänge zusammenhängen und aufeinander aufbauen, welche Revisions-, Dokumentations- und Sicherheits-Bedürfnisse jeweils bestehen.

Einfacher Workflow in IntarS 7

Angebote gibt es als individuelle Angebote, die nur einmal verwendet werden und Rahmen-Angebote, die immer wieder als Vorlagen zum Einsatz kommen. Bei der Erstellung eines Angebots kann auf ein anderes Bezug genommen und dessen Positionen hineinkopiert werden. Ein Angebot kann optionale und Alternativpositionen enthalten. Will der Kunde mehrere Angebote unterbreitet bekommen, speichert man sich die verschiedenen Versionen anstatt alles in einem Angebot zu ändern. So kann man auf alte Angebote zurückgreifen, falls der Kunde sich doch für eine frühere Offerte entscheidet. Wird aus einem Angebot nichts und kann es auch nicht als Vorlage für spätere Angebote dienen, kann man es manuell auf erledigt setzen. Ansonsten steht das gesamte CRM-Instrumentarium zur Verfügung (z. B. Wiedervorlage, Kontakte), um aus dem Angebot irgendwann einen Auftrag zu machen.

Aufträge können frei erfasst oder aus Angeboten zusammengestellt werden. Man kann dabei aus den Artikelstämmen oder Angebotspositionen für den Kunden suchen. Das Angebot verändert sich dabei nicht. Mit Erfassen eines Auftrages erhöht sich der Kundenauftragsbestand für die Artikel in den Auftragspositionen.

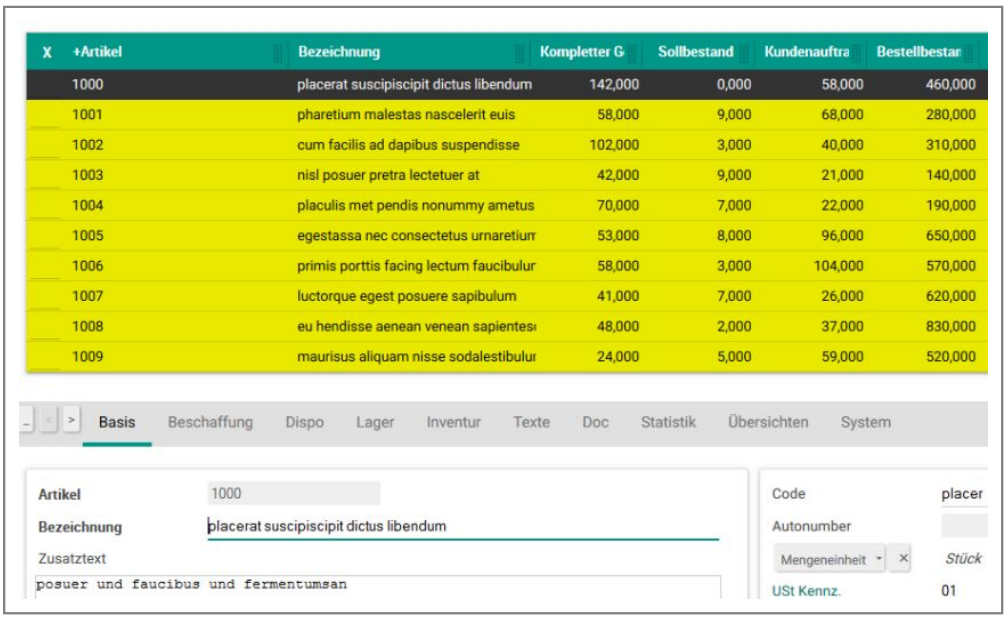
Aufträge werden von Lieferscheinen abgearbeitet. Man kann Lieferscheine bzw. deren Positionen auch frei erfassen, im Allgemeinen bezieht man sich jedoch auf Auftragspositionen. Dies bewirkt, dass die Teilmenge in den Auftragspositionen (und damit der Kundenauftragsbestand) reduziert wird. Wenn man überliefert, wird sie sogar negativ. Ein Auftrag, dessen Teilmengen alle kleiner oder gleich Null geworden sind, wird vom System auf erledigt gesetzt. Es können keine weiteren Lieferscheinpositionen mehr daraus abgeleitet werden. Er verschwindet dann auch aus bestimmten Ansichten. Will man einen Auftrag nicht oder nicht mehr vollständig ausliefern, kann man ihn manuell auf erledigt setzen. Die Teilmengen bleiben dabei zur Dokumentation stehen, zählen aber nicht mehr zum Auftragsbestand und der Status wird auf "erledigt" gesetzt. Solange ein Auftrag nicht komplett erledigt (entweder durch komplette Lieferung oder manuelles "erledigt"-setzen) ist, steht sein Status auf teil-erledigt. Wurde noch gar nichts geliefert, ist er noch offen. Ein offener Auftrag kann nach Belieben editiert werden. Sobald er teilerledigt ist, ist er gesperrt. Werden jedoch alle Lieferscheinpositionen gelöscht, die den Auftrag betreffen, ist er wieder offen.

Lieferscheine werden von Rechnungen abgearbeitet. Man kann Rechnungen bzw. deren Positionen auch frei erfassen, im Allgemeinen bezieht man sich jedoch auf Lieferscheinpositionen. Lieferscheine wirken direkt lagerbestandsreduzierend ohne extra Lagerbuchung. Sobald ein Lieferschein erzeugt/erfasst wurde, gelten die Teile als aus dem Lager entnommen. Auch ein Lieferschein kann auf erledigt gesetzt werden, z. B. wegen Fehlerfassung. Dann sind alle Positionen zu löschen und anschließend der Kopf auf erledigt zu setzen. In IntarS 7 können Belege nicht gelöscht werden. Will man einen Lieferschein nicht komplett berechnen, kann man ihn auf erledigt setzen. Seine Lagerbestandswirksamkeit bleibt davon unberührt. Sobald etwas von einem Lieferschein in einer Rechnung berechnet wurde, ist er teilerledigt. Wurde alles berechnet, ist er komplett erledigt. Solange nichts berechnet wurde, ist er offen und kann frei editiert werden. Löscht man alle Rechnungspositionen, die auf ihn verweisen, geht er wieder in den offenen Zustand.

Rechnungen können frei erfasst werden oder durch Übernahme von Lieferscheinpositionen, die berechnet werden sollen. Weil eine Rechnung nicht lagerbestandsmindernd wirkt, kann man nur sog. Service-Artikel direkt in die Rechnung einbringen. Solange eine Rechnung nicht gebucht ist, ist sie frei editierbar. Eine Rechnung kann nicht auf erledigt gesetzt werden. Sie wird vielmehr gebucht und erhält danach, hoffentlich, Zahlungseingänge. Gebuchte Rechnungen zählen zum Umsatz und beeinflussen Provisionen. Unter bestimmten Umständen kann das Buchen rückgängig gemacht werden. Hat man sich vertan, löscht man alle Positionen aus der Rechnung, schreibt in den Betreff eine Erklärung und bucht die leere Rechnung. Löschen kann man sie nicht, die Rechnungsnummer muss dokumentiert sein.

Der Standardablauf sieht so aus, dass es eine einfache Beziehung Auftrag - Lieferschein - Rechnung gibt. Mit jeweils einem Klick kann der Folgebeleg erzeugt und damit der aktuelle Beleg auf komplett erledigt gesetzt werden.

7.2 Artikelstamm



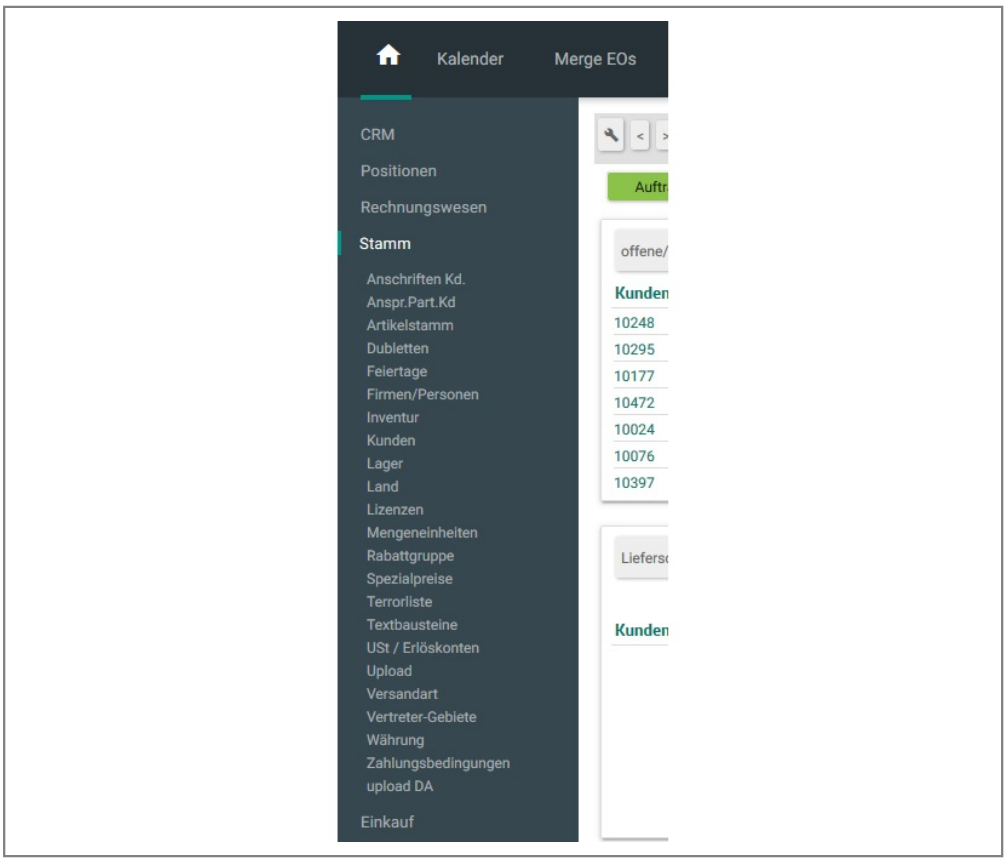
Der Artikelstamm

Der Artikelstamm (auch Teilestamm) ist neben dem Kundenstamm die wichtigste Stammdatentabelle. Hier ist hinterlegt, was das Unternehmen verkaufen kann. Ein Artikel hat eine eindeutige Artikelnummer, verschieden lange Textfelder in verschiedenen Sprachen zur Beschreibung, einen Preis sowie weitere Eigenschaften, die im Umgang mit ihm für das Unternehmen relevant sind, z. B. Größe, Gewicht, Farbe, Ausführung etc. Dazu kommen Eigenschaften für die betriebswirtschaftliche Behandlung. Dazu gehören Umsatzsteuersatz, Warentarifnummer, Erlöskonten usw. Hat man viele Artikel, kann man sie in Artikelgruppen gliedern, evtl. in Haupt- und Untergruppen oder eine Hierarchie oder mehrere Hierarchien. Ein Artikel hat weiterhin ein Memofeld für einfaches Wissensmanagement. Hier kann formlos alles eingetragen werden, was wissenswert ist. Für weitergehende Informationen kommt das Dokumentenmanagement zum Einsatz. Eine Sammlung von Steuerungs-Kennzeichen definiert den grundsätzlichen Charakter des Artikels, u. a. ob er lagerfähig, verkaufsfähig, ein Service-Artikel oder nur ein Textartikel ist. Diese Kennzeichen steuern in anderen Teilen der Warenwirtschaft, unter welchen Umständen der Artikel sichtbar ist und verwendet werden darf bzw. wie behandelt wird. Dazu kommen Einkaufsinformationen für die Beschaffung, Statistikfelder (durchschnittlicher EK, min/max EK, Absatz/Umsatz in lfd. und Vorjahr), Übersichten für Verwendung in Bewegungsdaten und Plugins (letzte Bestellungen), Lagerbestände, Bestell- und Auftragsbestand.

Ist ein Artikel einmal verwendet worden, kann er nicht mehr gelöscht werden. Um zu sehen, wo er verwendet wurde, gibt es ein Verwendungsprüfungsplugin. Man kann ihn aber inaktiv setzen, um seine weitere Verwendung zu verhindern. Natürlich kann man ihn auch wieder aktiv setzen und damit wiederbeleben. Ist das Bestellwesen installiert, hat ein Artikel Bezugsquellen, in denen hinterlegt ist, welcher Lieferant zu welchen Konditionen diesen Artikel liefert.

Der Lagerbestand wird beeinflusst von Inventurbuchungen, manuellen Bestandsbuchungen, Lieferscheinen, Stornos und Wareneingängen.

7.3 Hilfstabellen

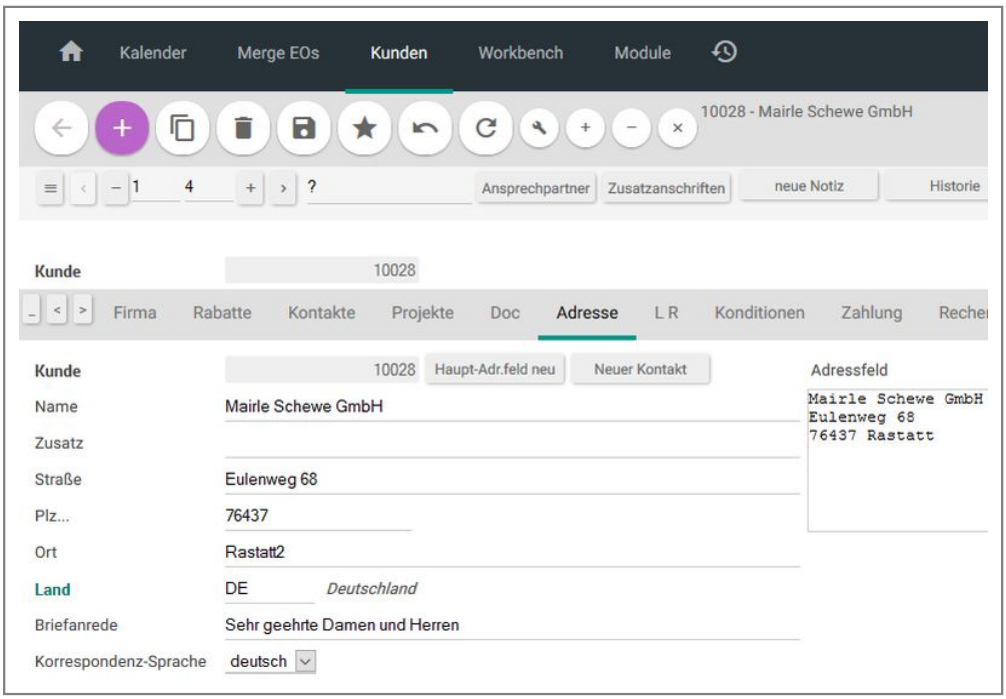


Hier geht's zu den Hilfstabellen

Zu den Hilfstabellen zählen MWSt-Sätze, Skontoprofil, Warengruppen, Länder, Währungen, Lager-Orte etc.

Sie sind teilweise schon vorab versorgt. Sie sollten aber vor Erfassen der Artikel- und Kundendaten nach den Erfordernissen des eigenen Unternehmens angepasst werden. Ihre Inhalte werden über Relationen aus den höherwertigen Stammdaten sowie Bewegungsdaten referenziert.

7.4 Kunden



Kundenstamm

In der Kundentabelle liegen alle Kunden, Lieferanten, Interessenten-Adressen etc. Sie ist neben dem Artikelstamm die wichtigste Stammdatentabelle. Ein Kundenstammsatz enthält neben Adressinformationen (Haupt-, Liefer-, Rechnungsadresse) Konditionen, klassifizierende Merkmale, betriebswirtschaftliche Eigenschaften, Statistikdaten, sowie ein großes Memofeld für einfaches CRM. Für weitergehendes CRM ist der eingebaute Kontaktmanager mit seinen Akten, Aufgaben, Terminen, Wiedervorlage und Recherche vorgesehen. Vom Kundenstammsatz aus kommt man mittels Übersichten und Plugins zu allen Verwendungen im Bewegungsdatenbereich (offene Aufträge, Rechnungen, Lieferscheine, Angebote). Ein Kunde kann mehrere Ansprechpartner und Zusatzadressen haben. Er ist normalerweise einem Benutzer fest zugeordnet, der für ihn zuständig ist (Betreuer) und evtl. Provision bekommt. Der Kunde ist ein möglicher Ausgangspunkt für Vorgänge wie neuer Auftrag, neue Rechnung u.v.m. Es wird dann jeweils der ganze Kundenkontext mitgenommen. Der Kunde ist eingebunden ins Dokumentenmanagement. Man kann ihm beliebige Dokumente als Anhänge zuordnen. Ergänzend werden ihm intern erzeugte Dokumente (Rechnungen, Anschreiben, Auftragsbestätigungen etc.) automatisch als PDF angehängt. Seine E-Mails können importiert und zugeordnet werden. Er kann per E-Mail, Fax oder SMS kontaktiert werden. Kontaktvorgänge werden im Kontaktmanager protokolliert und können dort weiterbearbeitet werden. So kann aus einer Notiz eine Aufgabe oder ein Termin werden. Hier kommt dann die eingebaute Groupware mit ins Spiel. Zum CRM gehört die Möglichkeit, Stichworte anzulegen und Kunden beliebig zuzuordnen.

7.5 Angebot

Angebote haben eine Bindungsfrist, ein Thema (Betreff), einen einleitenden Text und in den Positionen neben den normalen Artikeln noch Textpositionen und Alternativpositionen. Die Reihenfolge der Positionen kann über eine Sortiernummer beeinflusst werden, unabhängig von der Erfassungsreihenfolge. Es empfiehlt sich, einen Vorrat von Rahmenangeboten zu erfassen. Bei Bedarf können dann daraus mit einem Klick konkrete Angebote erzeugt werden, bei welchen ggf. nur noch Details geändert werden müssen. Aus einem Angebot können auch immer wieder direkt Aufträge erzeugt werden.

Angebote können per E-Mail gesendet werden. Sind diesem Anlagen hinterlegt, öffnet sich ein Zwischenbild, in dem die mitzuschickenden Anlagen ausgewählt werden können.

7.6 Auftrag

Auftragskopf

Wie alle Belege übernimmt ein Auftragskopf die Adress- und Konditions-Daten des Kunden in Kopie. Sie können dann fallbezogen abgeändert werden.

Ein Auftrag kann mit einem Klick aus einem Angebot erzeugt, durch Übernahme von Angebotspositionen zusammengesetzt, durch Übernahme oder Eingabe von Artikeln frei erfasst oder aus einer Kombination von alledem erstellt werden.

Während des Editierens wird das Zahlenwerk laufend aktuell durchgerechnet. Das ist bei allen Belegen so. Welche Zahlen relevant sind, hängt vom Kontext ab, z. B. Endkunde oder gewerblicher Kunde, Inland, EG, Ausland, mit/ohne UStID. Beträge werden intern mit höherer Genauigkeit geführt, um Rundungsdifferenzen gering zu halten. Auch die Beträge im Artikelstamm haben daher mehr als zwei Nachkommastellen. Auf Ausdrucken werden nur zwei Dezimalstellen dargestellt.

Offene Aufträge erscheinen im gleichnamigen Plugin auf der Startseite, von wo aus sie aufgerufen und bearbeitet werden können. Hierfür müssen jedoch die entsprechenden Voraussetzungen erfüllt sein.

Die Artikel, welche verkauft werden sollen, werden in Auftragspositionen erfasst.

Auftragspositionen

Es gibt mehrere Möglichkeiten, Auftragspositionen zu erzeugen.

- Klicken Sie auf "Neu" und geben Sie den Artikel ein.
- Übernehmen Sie die gewünschte/n Position/en aus den Angebotspositionen.
- Übernehmen Sie die gewünschte/n Position/en aus dem Artikelstamm.

Beim Erfassen der Positionen kann man sich der Suchwerkzeuge bedienen. Gibt man im Feld der Artikelnummer einen Punkt (".") an, wird der String bis dahin als Suchstring interpretiert und es erscheint eine Liste mit den Treffern. Die Suche berücksichtigt dabei die Informationen aus dem Metadaten-Modell, welche Felder in welcher Reihenfolge durchsucht werden sollen. In mehreren Schritten wird zunehmend unschärfer gesucht bis eine gewisse Trefferanzahl, die sich steuern lässt, erreicht ist.

Daneben gibt es eine Historie der 20 zuletzt aufgerufenen Artikelstämme, sowie die persönlich "gemerkten" (s. gemerkte), in der man sich beliebig viele Artikel merken kann. Aus all diesen Nachschlagefunktionen kann bequem mit der Tastatur übernommen und so der Auftrag schnell komplettiert werden.

In der Auftragsposition gibt man in der Anzahl an, wieviel von dem Artikel verkauft werden soll. Daneben gibt es die Teilanzahl. Sie besagt, wieviel noch zu liefern ist. Am Anfang ist sie genauso groß wie die Anzahl. Wird der Auftrag abgearbeitet, verringert sich die Teilanzahl bis auf 0. Eine Auftragsposition mit Teilanzahl 0 gilt als erledigt. Soll aus irgendeinem Grund eine Auftragsposition nicht oder nicht mehr weiter abgearbeitet werden, kann sie manuell auf "erledigt" gesetzt werden. Eine Auftragsposition wird immer durch Lieferscheine abgearbeitet.

Die noch offene Teilanzahl zählt zum Kundenauftragsbestand. Er ist im Artikelstamm zu sehen und spielt eine wichtige Rolle in der Disposition, indem er als Bedarf den Saldo reduziert. Es ist Aufgabe der Disposition, den Kundenauftragsbedarf zu befriedigen und Artikel für die Auftragsposition zu reservieren. Die reservierte Anzahl in der Auftragsposition kann geliefert werden. Wie weit reserviert ist, kann im Reservierungsgrad abgelesen werden. Erst wenn der Reservierungsgrad größer als 0 ist, kann etwas geliefert und somit der Auftrag abgearbeitet werden. Das System versucht, schon bei Erfassung der Auftragsposition automatisch zu reservieren und bedient sich dabei des verfügbaren Bestandes. Dieser errechnet sich aus dem physischen Lagerbestand minus dem reservierten Bestand. Reicht der verfügbare Bestand nicht aus, kann nicht automatisch komplett reserviert werden. Die Auftragspositionen treten miteinander in Konkurrenz um Material und ein Disponent muss entscheiden, wer wieviel bekommt. Durch entsprechend gepflegte Prioritätsregeln kann das System auch automatisch entscheiden.

Wird im Config-Modul die Disposition ausgeschaltet, vereinfacht sich der Ablauf derart, dass immer automatisch zu 100 Prozent reserviert wird. Die im nächsten Abschnitt beschriebenen Dispositionsfunktionen entfallen in diesem Fall.

Disposition

Disposition kann unter verschiedenen Blickwinkeln erfolgen.

Es kann auf Artikelstammebene disponiert werden. Dort gibt es ein Plugin, das alle Auftragspositionen zeigt, die um den Artikel konkurrieren. Hier kann umverteilt werden. Man kann auch den Button für automatische Verteilung anklicken.

Auftragspositionsbezogene Disposition: innerhalb einer Auftragsposition kann Material zugeteilt bzw. weggenommen werden.

Im "Home"-Modul gibt es ein Plugin, das alle zuteilungsfähigen Aufträge mit ihrem aktuellen Reservierungsgrad zeigt. Von da aus können die Aufträge aufgerufen und disponiert werden. Es gibt auch einen Button, um das System automatisch alle verfügbaren Artikel auf alle wartenden Auftragspositionen verteilen zu lassen.

Nur lagerfähige Artikel nehmen an der Disposition teil. Nicht lagerfähige Artikel (z. B. Dienstleistungen oder Gebühren) werden automatisch immer komplett reserviert. Sie sind unbeschränkt verfügbar.

Preise

In der Auftragsposition erfolgt die Preisfindung. Der Preis ist im Artikelstamm hinterlegt. Sonderpreise pro Kunde, Menge, Zeitraum etc. lassen sich in den Spezialpreisen verwalten. Zusätzlich kann noch Rabatt gewährt werden.

7.7 Lieferschein

Lieferscheine repräsentieren die Warenbewegung Richtung Kunde. Wird mit Mehrlager-Verfahren gearbeitet, enthält der Lieferschein auch die Information, von welchem Lager die Ware entnommen wird. Sie sollte dann auch tatsächlich von dort entnommen werden, sonst kommt es zu Inventur-Differenzen. Das Lager wird in allen Belegen mitgeführt und aus dem Benutzerstammsatz vorbelegt.

Liefern Sie vorab nur einen Teil eines Auftrages, wird auf dem Ausdruck des Lieferscheines angedruckt, welche Menge noch offen ist. Ein Lieferschein kann aus beliebigen, nicht erledigten Auftragspositionen vom selben Lager für denselben Kunden zusammengestellt werden. Jede Position eines Beleges hat einen "Lebenslauf". Darin stehen die Relationen zu allen Vorgängerbelegen. In einer Rechnungsposition ist somit hinterlegt, aus welchem Angebot, Auftrag oder Lieferschein sie hervorgeht. Ein Beleg aktualisiert immer seinen Vorgängerbeleg, indem er dessen Teilmenge (offene, noch nicht abgearbeitete Menge) reduziert.

7.8 Rechnung

Rechnungen bauen im Allgemeinen auf Lieferscheinen auf. Was geliefert wurde, wird berechnet. Man kann jedoch auch freie Rechnungspositionen erfassen. Als Besonderheit in IntarS 7 kann eine Projektposition abgerechnet werden. Das ist besonders interessant für Unternehmen, die projektbezogen arbeiten und nach Aufwand fakturieren. Man legt geschickterweise für jeden Stammkunden ein Wartungsprojekt an und darin pro Monat eine Projektposition, in der die Zeiterfassung vorgenommen wird. Dann kann man am Monatsende mit einem Klick diese Projektposition in eine Rechnungsposition übernehmen mit exakter Zeit und Tätigkeitsbeschreibung.

Gebuchte Rechnungen begründen Forderungen. Jeder Kunde in IntarS 7 hat ein Kundenkonto, auf welchem Rechnungen mit Gutschriften und Stornos sowie Zahlungsein- und ausgängen saldiert werden. Man hat in IntarS 7 also jederzeit einen Überblick über die wirtschaftliche Situation. Mit dem Rechnungswesen kommen Kostenbelege, Abschreibungen und BWA hinzu. Dennoch ist IntarS 7 keine Buchhaltungs-Software, sondern vielmehr ein Werkzeug ohne jegliche (steuer)rechtliche Relevanz. Ebenso kann man mit Bleistift und Papier oder mit Excel rechnen und es nach Belieben ändern oder wieder löschen. Die richtige Buchhaltung sollte extern von einem Steuerberater oder Buchhaltungsbüro gemacht werden. Sie wissen, worauf sie achten müssen, sind erfahren im Umgang mit Steuerprüfungen, haben eine entsprechende Berufshaftpflicht. Eine richtige Buchhaltung nach den GOB unterliegt strengen Restriktionen. IntarS 7 soll die Geschäftsvorfälle mit all ihren Unregelmäßigkeiten abwickeln und am Monatsende eine Liste für den Steuerberater exportieren, der sie dann in eine makellose Buchhaltung schreibt oder importiert.

7.9 Mahnung

Eine Mahnung, auch Zahlungserinnerung genannt, ist die bestimmte und eindeutige Aufforderung des Gläubigers an den Schuldner, die geschuldete Leistung zu erbringen.

Die Mahnung ist (neben Bestehen eines fälligen Anspruchs und Nichtleistung durch den Schuldner) Voraussetzung für den Verzug des Schuldners (§ 286 BGB).

So wird es gemacht:

1. Legen Sie ein Mahnprofil mit Fristen, Gebühren und Texten an (Bild 1).
2. Ordnen Sie das Mahnprofil dem Kunden zu (Bild 2).
3. Jetzt legen Sie einen Mahnlauf an. Dazu geben Sie, nachdem Sie einen neuen Mahnlauf erstellt haben, das Datum ein, für welches der Mahnlauf ausgeführt werden soll und sichern den neuen Datensatz (Bild 3 Schritt 1).
4. Klicken Sie nun auf den Button "Mahnvorschläge erzeugen". Das System sucht daraufhin alle Rechnungen, die angemahnt werden können, und erstellt für diese Mahnvorschläge Positionen des Mahnlaufes (Bild 3 Schritt 2). Zur Kontrolle wird ein Protokoll erstellt und angezeigt.
5. Nun können Sie die Vorschläge löschen, die nicht angemahnt werden sollen, evtl. Gutschriften erstellen oder Skonti als Zahlungseingänge erfassen, um Rechnungen auszugleichen (Bild 3 Schritt 3).
6. Solange ein Mahnlauf nicht verarbeitet ist, können die Mahnvorschläge jederzeit neu erzeugt werden, wobei die aktuelle Zahlungssituation berücksichtigt wird.
7. Wenn Sie den Button "Mahnvorschlag verarbeiten" klicken, werden für die verbliebenen Mahnvorschläge Mahnungen als Anschreiben erstellt und die Rechnungen aktualisiert (Bild 3 Schritt 4).
8. Die so erzeugten Anschreiben können nachbearbeitet werden.
9. Der Button "Mahnungen drucken" druckt alle Mahnungen (Anschreiben) des Mahnlaufes (Bild 3 Schritt 5). Natürlich können die Anschreiben auch individuell gedruckt, per E-Mail versendet und gefaxt werden.

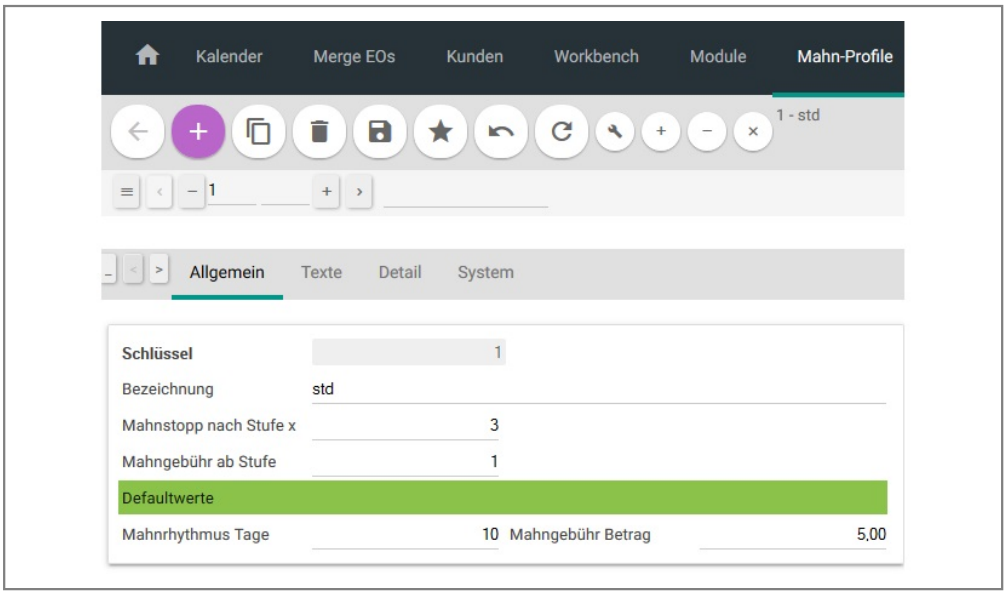


Bild 1: Ein Mahnprofil anlegen

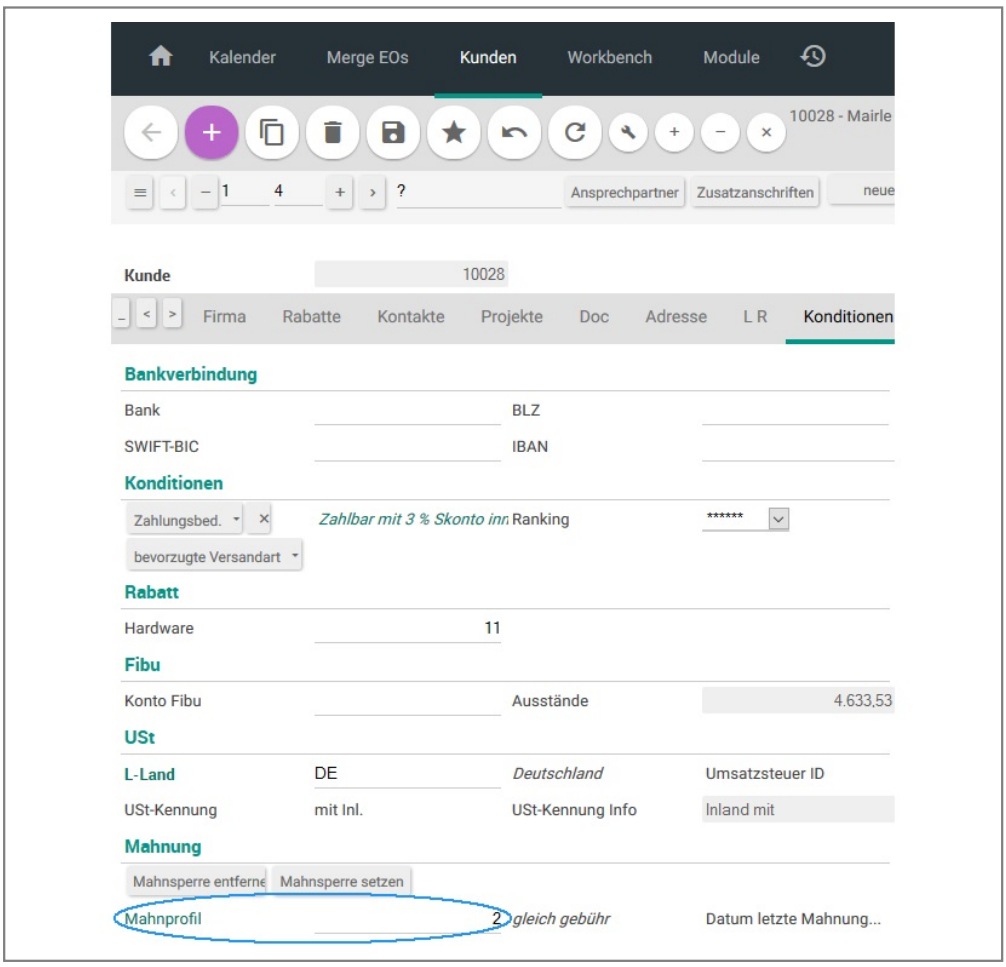


Bild 2: Ein Mahnprofil einem Kunden zuordnen

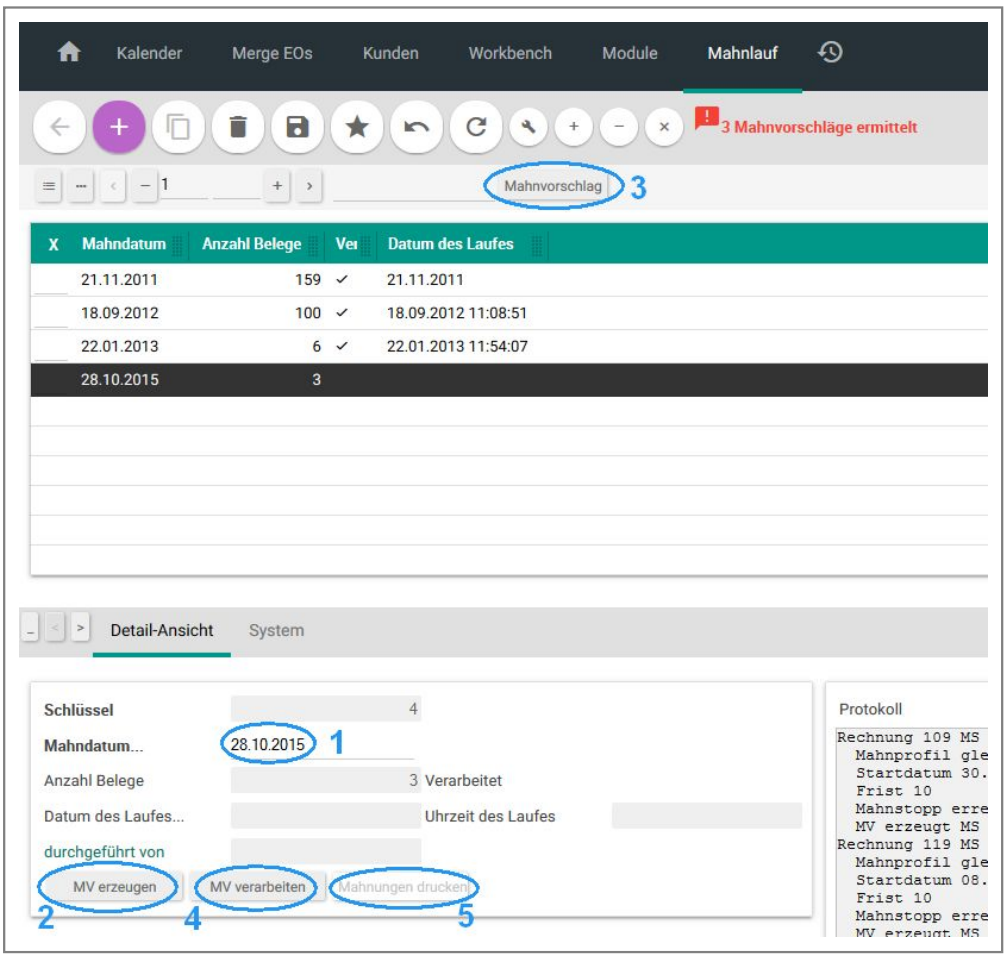


Bild 3: Die Mahnungen erzeugen, bearbeiten und drucken

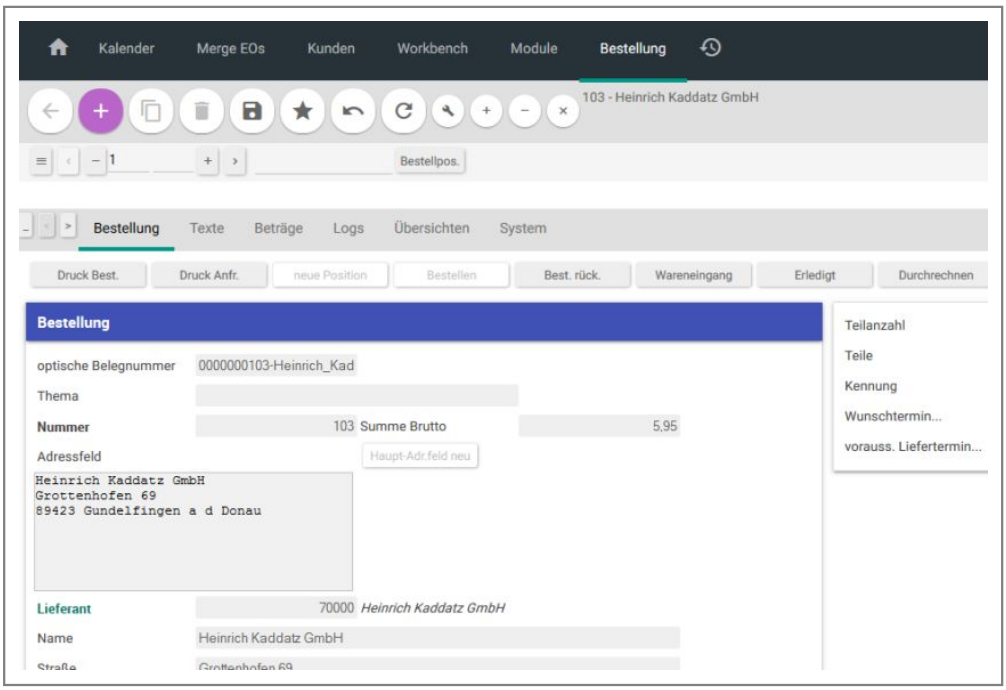
7.10 Storno

Stornos erstellt man, um gleichzeitig Ware vom Kunden in das Lager zurückzunehmen und ihm auf seinem Kundenkonto einen Betrag gut zu schreiben. Man nennt es auch Warengutschrift. In anderen Branchen bedeutet Storno die exakte Umkehr eines Geschäftsvorfalles. Das ist in IntarS 7 nicht der Fall. Zwar gibt es einen Button "Storno" an der Rechnung, dieser erzeugt jedoch einen spiegelbildichen Stornobeleg zur Rechnung. Man kann diesen aber abweichend bearbeiten und z. B. nur einen Teil der Waren zurücknehmen oder nur einen Teil des Preises nachlassen. Stornos werden gern für Teststellungen oder Miete verwendet. Dem Kunden wird eine Anlage gegen eine Gebühr zum Testen für eine bestimmte Zeit überlassen. Dazu wird sie ihm per Rechnung verkauft. Behält er sie nach Ablauf der Zeit, bezahlt er die Rechnung. Andernfalls schickt er sie zurück und bekommt den Rechnungsbetrag abzüglich einer Wiedereinlagerungs-/Nutzungsgebühr gutgeschrieben. Dazu wird dann ein Storno erzeugt. Auf dem Kundenkonto verbleibt die Gebühr als Saldo und wird hoffentlich von einem Zahlungseingang in gleicher Höhe ausgeglichen.

7.11 Gutschrift

Gutschriften sind rein betragsmäßige Belege. Es findet keine Warenbewegung statt. Sie werden eingesetzt, um z. B. aufgrund begründeter Mängelrügen dem Kunden einen Nachlass zu gewähren. Gutschriften reduzieren die Forderungen auf dem Kundenkonto. Weist das Kundenkonto dauerhaft ein Guthaben aus, wird man es mit einem Zahlungsausgang ausgleichen müssen, wenn sich keine zukünftigen Forderungen absehen lassen. Wie Rechnungen, Eingangsrechnungen und Stornos werden Gutschriften gebucht, um sie festzuschreiben. Davor sind sie frei editierbar. Gutschriften verringern wie Stornos Umsatz und ggf. Provision. In IntarS 7 können Buchungen von Belegen auch wieder rückgängig gemacht werden, solange sie nicht über eine Fibu-Schnittstelle nach außen gelangt sind. Das ist eine gute Sache, um Korrekturen auszuführen, wie sie eben im Geschäftsleben unvermeidlich sind. Zum Beispiel: ein Kunde will die Rechnung geändert haben (andere Texte, zurück-/vordatieren, Position rausnehmen und in nächste Rechnung aufnehmen, Betrag reduzieren etc.). Statt dann eine aufwändige Rückabwicklung mit Gutschrift und erneuter Rechnung zu machen, kann auf "Buchen rückgängig" geklickt, die Änderung vorgenommen, erneut gebucht werden, ohne Belegnummern zu verbrauchen. Die alten Ausdrucke kommen in den Aktenvernichter. Das ist unternehmerische Freiheit mit freier Software. Der "Buchen rückgängig"-Button ist an Sonderberechtigungen geknüpft. Ein normaler Benutzer sieht ihn nicht.

7.12 Bestellung



Eine offene Bestellung

IntarS 7 unterstützt verschiedene Bestellverfahren. Welches das richtige ist, hängt vom Artikel ab. Generell sollte man sich mit C-Artikeln nicht weiter aufhalten und am besten einen langfristigen Lieferanten haben, der das ganze Sortiment unproblematisch auf Abruf liefert. Bei schwer zu beschaffenden Artikeln greifen die klassischen Bestellverfahren nicht, hier zählen Beziehungen, Verhandlungsgeschick, Kenntnis von Ersatzartikeln und Marktkenntnis. Das SRM (Supplier Relationship Managment) unterstützt dabei. Weil Lieferanten in IntarS 7 im Kundenstamm (Flag "Lieferant") geführt werden, kommt das CRM als SRM zum Einsatz. Bei allen anderen Artikeln befindet man sich im klassischen Zielkonflikt zwischen Kapitalbindung im Lager, Mengenrabatt, Wertverlust durch Alterung, Risiko von Preisschwankungen und Verfügbarkeit. In engen Märkten spielen zusätzlich spekulatorische Aspekte eine Rolle (Erlangung eines Monopols für einen Artikel). Artikel, die sich beständig verkaufen (z. B. Verbrauchsmaterial), können auf Vorrat beschafft werden, insbesondere wenn sie nicht verderblich sind. Selten benötigte Artikel werden möglichst erst dann beschafft, wenn sie von einem Kunden bestellt wurden. Ebenso teure Artikel und solche, die man problemlos schnell bekommt. Interessant wird es bei Artikeln, bei welchen ein Käufermarkt besteht. Hier kann es sich lohnen, bei jeder Bestellung den gerade günstigsten Anbieter zu ermitteln. Die damit verbundene Arbeit muss mindestens von der erzielten Einsparung gedeckt sein. Gibt es wenig Schwankungen, ist es meist am besten, einmal den besten Anbieter zu ermitteln und dann als Hauptlieferanten einzutragen und nur in Ausnahmefällen auf einen anderen auszuweichen. Informationen bezüglich der Anbietersituation werden in den Bezugsquellen hinterlegt. Der Saldo eines Artikels gibt Auskunft darüber, was bestellt werden sollte. Er zeigt die Differenz zwischen Deckern (Lagerbestände, Bestellbestand, Produktionsaufträge) und Bedarfen (Kundenauftragsbestand, Sekundärbedarfe, Mindestbestand).

Bestellungen können auf vielfältige Weise von verschiedenen Punkten aus erzeugt werden.

- Klicken Sie im Artikelstamm auf den Button "bestellen". Sofern er einen negativen Saldo (Bedarfe übersteigen Decker) und Hauptlieferanten hat, wird eine bestehende noch offene Bestellung des Hauptlieferanten dieses Artikels gesucht. Falls keine gefunden wird, wird eine angelegt. Darin wird eine neue Position erzeugt bzw. eine bereits bestehende über denselben Artikel erhöht. Als Bestell-/Erhöhungsmenge wird der Saldo genommen.
- Auch aus der Bezugsquelle heraus kann mit einem Klick die benötigte Menge bestellt werden. Es

wird der Lieferant der Bezugsquelle genommen und nach einer offenen Bestellung von ihm gesucht. Falls noch keine vorhanden ist, wird eine angelegt. Darin wird eine neue Position erzeugt bzw. eine bereits bestehende über denselben Artikel erhöht. Als Bestell-/Erhöhungsmenge wird der Saldo genommen.

Bestellungen können auch frei erfasst werden. Hat man im Kopf den Lieferanten angegeben, kann in den Positionen gesucht werden. Dafür stehen verschiedene Nachschlagefunktionen zur Verfügung:

- Artikel des Lieferanten mit Bedarf
- Artikel des Lieferanten
- Artikel mit Bedarf
- freie Artikelsuche

Sobald eine Bestellposition existiert, zählt sie automatisch zum Bestellbestand. Dies ist so gewollt, da so verhindert wird, den Artikel nochmals zu bestellen.

Eine neue (offene) Bestellung erscheint im entsprechenden Plugin im "Home"-Modul (unter EK). Der nächste Schritt ist, die Bestellung zu drucken, faxen oder mailen. Evtl. wird auch telefonisch bestellt. Um zu dokumentieren, dass die Bestellung bestellt ist, klickt man auf den Button "bestellen". Daraufhin wird ein Zeitstempel gesetzt. Die Bestellung erscheint im Plugin "bestellte Bestellungen" und ist nicht mehr editierbar. Für Ausnahmefälle kann der Vorgang mit "bestellen rückgängig" zurückgenommen werden. Bestellte Bestellungen unterliegen der Liefertermin-Überwachung. Ist ein solcher eingetragen, wird die Bestellung rot angezeigt, sobald er überschritten ist.

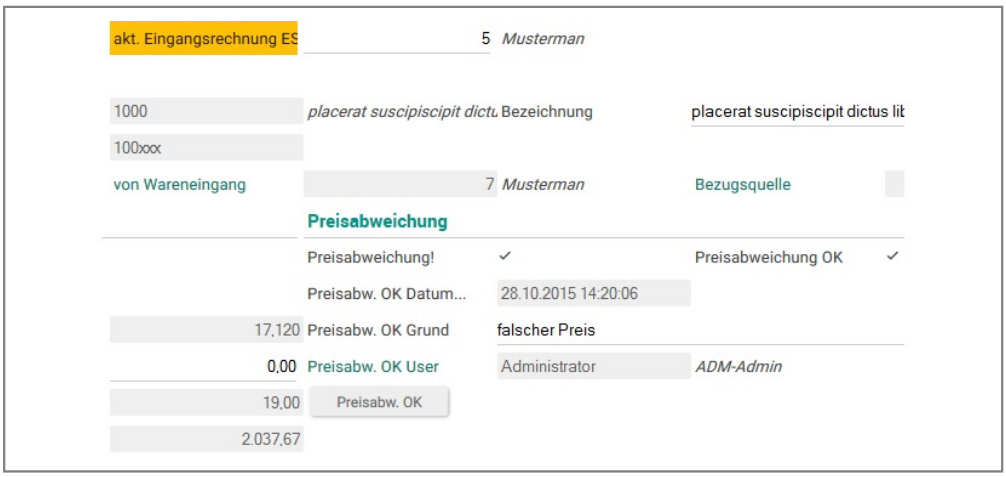
Wird die bestellte Ware angeliefert, geht es weiter mit der Eingangsrechnung.

Erlischt das Interesse an der Bestellung bzw. an einzelnen Positionen oder kann der Lieferant doch nicht liefern, kann dies mit dem "erledigt" Button auf Positions- oder Kopfebene festgehalten werden. Ansonsten werden Bestellungen durch Wareneingänge (Eingangsrechnungen) abgearbeitet.

7.13 Disposition

Mit Disposition in einer Warenwirtschaft meint man die mengenmäßige Einteilung von Aufträgen mit aktuellen Leistungsanforderungen und die terminierte Zuweisung zu den verfügbaren Ressourcen. Konkret sieht das so aus, dass Kundenaufträgen, die evtl. in Konkurrenz zueinander stehen, Artikel zugeteilt werden (reserviert). Ist der Artikel nicht am Lager, muss er entweder beschafft, produziert oder durch einen anderen Artikel ersatzweise gedeckt werden. Die Disposition muss sich also auch darum kümmern, was in welcher Menge zu welchen Konditionen bei welchem Lieferant bestellt wird. Die wichtigste Information bezüglich der Leistungsanforderung ist dabei der Saldo der Decker und Bedarfe eines Artikels. Auf eine zeitliche Staffelung (Teile-Konto) wird in der Standardversion von IntarS 7 aus Vereinfachungsgründen verzichtet. Lässt man sich die Artikel nach Saldo absteigend sortiert anzeigen, hat man auf einen Blick den Arbeitsvorrat der Disposition. Der Saldo eines Artikels wird vom System immer aktuell gehalten, indem er bei jeder Änderung, die ihn beeinflusst, neu berechnet wird (Lagerbewegung, Auftragserfassung, Bestellerfassung etc.).

7.14 Eingangsrechnung



Eingangsrechnungsposition mit Preisabweichung

Eingangsrechnungen leiten sich aus Bestellungen ab. Normalerweise erwartet man, dass das, was bestellt wurde, auch geliefert wird zu den Preisen, die bei der Bestellung vereinbart wurden. So sollte es in den meisten Fällen korrekt sein, wenn die Eingangsrechnung 1:1 aus einer Bestellung erzeugt wird. Genau dafür gibt es einen entsprechenden Button in der Bestellung. Erlangt der Disponent Kenntnis von einem Wareneingang (da die Ware auf dem Hof abgeladen wird, durch ein Liefer-Avis oder Warenbegleitzettel), muss zunächst die zugrundeliegende Bestellung gefunden werden. Es können auch mehrere sein. Aufgrund dessen, dass es ein Plugin "offene bestellte Bestellungen" gibt, sollte dies einfach sein. Man kann natürlich auch im Bestellmodul suchen. In der Bestellung klickt man dann auf "Eingangsrechnung". Die Eingangsrechnung wird erzeugt und erledigt gleichzeitig die Bestellung. Man vergleicht sie mit dem tatsächlichen Wareneingang mengen- und wertmäßig. Eventuelle Abweichungen werden in der Eingangsrechnung eingetragen. Mindermengen lassen die Bestellmenge wieder aufleben und ihren Status auf teil-erledigt ändern. Ebenso, wenn Positionen überhaupt nicht geliefert wurden. Diese löscht man aus der Eingangsrechnung heraus. Treten zu große Preisabweichungen auf (Kriterien einstellbar), wird die Eingangsrechnung vom System rot markiert und kann nicht gebucht werden. Erst wenn die Preisabweichung mit Grund, Zeitstempel und User akzeptiert oder vom Lieferanten berichtigt wurde, verschwindet das Kennzeichen und es kann gebucht werden. Nur gebuchte Eingangsrechnungen dürfen zur Zahlung angewiesen werden.

Ungeachtet von Preisabweichungen wirkt eine Eingangsrechnungsposition immer lagerbestandserhöhend für ihren Artikel mit ihrer Menge. Es ist keine extra Lagerbuchung mehr notwendig.

Erzeugt man eine Eingangsrechnung aus einer teil-erledigten Bestellung, wird sie über die verbleibenden Teilmengen gebildet. So arbeiten unter Umständen mehrere Eingangsrechnungen sukzessive eine Bestellung ab bis sie erledigt ist. Stellt sich heraus, dass für eine offene oder teil-erledigte Bestellung keine Ware mehr kommt (Lieferant pleite, Ware nicht mehr lieferbar...), kann die Bestellung auch manuell auf erledigt gesetzt werden. Damit erscheint sie nicht mehr in den Plugins und Übersichten offener Bestellungen und zählt auch nicht mehr zum Bestellbestand.

Eine Eingangsrechnung kann auch aus mehrere Bestellungen oder Teilen davon erzeugt werden. Dies kann notwendig sein, wenn der Lieferant eine Sammellieferung mit Waren aus mehreren Bestellungen sendet. Dann erzeugt man Eingangsrechnungspositionen, indem man die gelieferten Bestellpositionen aus der Bestellung sucht.

Beim Buchen der Eingangsrechnung werden auch die Statistikfelder der Artikelstämme versorgt (EK-Preis max/min, Durchschnitt). Um EK-Preise vergleichen zu können, werden sie auf die Hauswährung umgerechnet und so in den Statistiken abgestellt.

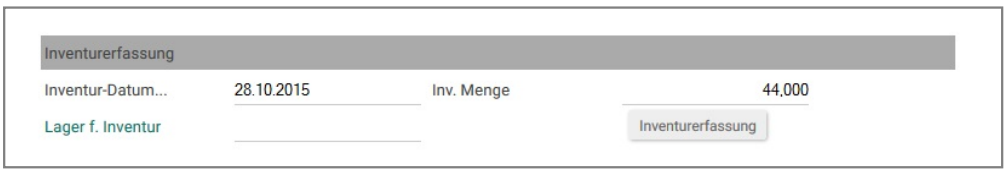
7.15 Lager und Inventur

Lager- und Saldosicht im Artikelstamm

IntarS 7 unterstützt Mehrlagerverwaltung. Allerdings können nicht beliebig viele Lager angelegt werden. Im Standard ist die Anzahl der Lager auf 6 begrenzt und die Lagerbestände werden direkt im Artikelstamm geführt. Dieses Modell ist ein Kompromiss zwischen der Flexibilität einer unbeschränkten Mehrlagerverwaltung und Performance, Übersichtlichkeit, Komplexität, sowie Overhead bei Bearbeitungsvorgängen. Natürlich kann auch mit nur einem Lager gearbeitet werden. Dies ist die Einstellung bei Auslieferung der Software.

Das Default-Lager ist am Benutzerstammsatz hinterlegt. Hier sollte das eingetragen werden, welches am häufigsten verwendet wird. Aus dem Benutzerstammsatz wird es in alle Belege vorbelegt, kann aber jeweils fallbezogen abgeändert werden.

Die häufigsten Lagerbewegungen sind Wareneingänge vom Lieferanten und Warenausgänge zum Kunden. In IntarS 7 ist es so, dass dafür keine extra Lagerbewegungen erfasst werden müssen. Vielmehr wirken die Positionen der Belege direkt als lagerbestandswirksame Transaktionen. Für Umlagerungen, Verschrottung und temporäre Entnahme, z. B. zur Ansicht, werden jedoch Lagerbewegungen erforderlich.



hier wurde gerade der Lagerbestand erfasst

Inventur ist immer ein schwieriges Thema. Eigentlich müssten alle Belege verbucht sein, der Betrieb ruhen und dann Inventur gemacht werden. Das ist aber nicht immer praktikabel. Daher bietet IntarS 7 eine laufende Inventurmöglichkeit für jedes Lager. Man kann jederzeit den vorgefundenen Lagerbestand eines Artikels erfassen. Dabei hält das System den Zeitpunkt, den alten Bestand und die resultierende Inventurdifferenz fest. Es wird dann ein Inventurbuchungssatz geschrieben. Alle folgenden Buchungen bauen dann auf diesem neuen Bestand auf. Werden anschließend Belege mit einem älteren Wirksamkeitszeitpunkt ins System gebracht, beeinflussen sie den Lagerbestand nicht mehr, sondern nur noch die Inventurdifferenz im folgenden Inventursatz. Kommt doch mal etwas Unvorhergesehenes dazwischen (Stromausfall, Hardwaredefekt, fehlerhafte Scripts, Veränderungen direkt per SQL), kann es mit dem Lagerbestands-Reorg im Artikelstamm wieder neu durchgerechnet werden. Das System geht dabei zurück bis zum neuesten Inventursatz, nimmt dessen Lagerbestand und saldiert dann alle zeitlich folgenden Bewegungen darauf und ermittelt so den neuen rechnerischen Lagerbestand.

7.16 Preise

X	Artikel	Kunde	Gültig ab	Gültig bis	Kunden-Preis
	1010	10037			49,50
	1010	10071			49,50
	1010	10094			49,50
	1010	10102			49,50
	1010	10109			49,50
	1010	10130			49,50
	1010	10163			49,50
	1010	10241			49,50
	1010	10272			49,50
	1010	10309			49,50

Detail-Ansicht System

Kunde	10037	*Chemicon Budde GmbH*	
Artikel	1010	*temperdiet psum tor hac c*	
Gültig ab...		Gültig bis...	
Kunden-Preis	49,50		
Staffelmenge1	10,000	Staffel Preis1	33,00000
Staffelmenge2	25,000	Staffel Preis2	24,75000
Staffelmenge3	100,000	Staffel Preis3	19,80000

Spezialpreis für einen Kunden

IntarS 7 bietet standardmäßig ein einfaches Modell zur Preisfindung. Es gibt primär im Artikelstamm den Verkaufspreis. Dieser wird verwendet, wenn sonst nichts bekannt ist. Im Script "_rabatt_staffel.script" können systemweit Rabatte für Staffelmengen hinterlegt werden.

Man kann jedoch für spezielle Kunden, Artikel, Mengen und Zeiträume Spezialpreise definieren, die dann vom System verwendet werden. Die Preisfindung in der Auftragsposition vermerkt, wie der Preis zustande kam.

Weiterführende Preisfindungen (weitere Rabatte, Verbands-, Listen-, Kundenpreise, Preise für Artikelkombinationen, umsatzabhängige Preise, Staffeln nach Artikelgruppen uvm.) werden in individuellen Skripten abgehandelt. Zuständig ist das Script "_subroutinen.script" und dort die Subroutine "preisErmitteln".

Da hier alles möglich ist, gehen Sie bitte mit äußester Vorsicht damit um. Es sollte mit endlichem Aufwand nachvollziehbar sein, wie ein Preis zustande kommt. Wenn der Preis sich so kompliziert berechnet, dass der Kunde jedesmal überrascht wird, ist nichts gewonnen. Je aufwändiger die Preisfindung, desto höher ist die Fehlerquote, auch bei den eigenen Mitarbeitern. Es sollte auch eine Preiserhöhung überschaubar bleiben und evtl. eine Preisliste gedruckt werden können. Im Polstermöbelbereich ist es z. B. so, dass oft nur für einzelne Beispiele einer Konfiguration ein Preis genannt werden kann.

Der vom System ermittelte Preis und Rabatt kann jeweils in Angebot, Auftrag, Lieferschein oder Rechnung nochmal manuell abgeändert werden, sofern der Benutzer die Berechtigung dazu besitzt. Es bleibt immer die Möglichkeit bestehen, Nachlass- oder Zuschlagspositionen zu erfassen.

Preise werden intern mit höherer Genauigkeit geführt. Das eröffnet die Möglichkeit, optische Preis wie 9,99 mit/ohne MWSt. zu erreichen und verringert die Problematik der Rundungsdifferenz.

Umsätze werden nach den drei Umsatzsteuersätzen gruppiert. Jede Position eines Beleges enthält das gesamte Zahlwerk in Haus- und Fremdwährung, sowie mit den Umsatzsteuersätzen und -beträgen. Angezeigt bzw. weiterverarbeitet wird das, was aus dem Kontext heraus davon anzuwenden ist.

Aus Preis und dem mittleren EK wird in jeder Belegposition der Rohgewinn (Deckungsbeitrag 1) berechnet. Dieser dient als Anhaltspunkt für Preisverhandlungen.

7.17 Projektmanagement

Die Arbeit an und in Projekten gehört fest zum beruflichen Alltag.

Projektarbeit unterscheidet sich vom "normalen" Tagesgeschäft. Projekte haben häufig eine definierte Zielvorgabe, sind zeitlich begrenzt, haben ein limitiertes Budget und eine spezifische Organisation.

Projekte sind in der Regel einmalig. Das heißt, dass zwar bestimmte Aufgaben eines Unternehmens öfter vorkommen und ähnlich sein können, sich aber bezogen auf die Rahmenbedingungen unterscheiden.

Gehen Sie folgendermaßen vor:

1. Legen Sie ein Projekt an und ordnen sie diesem Projekt einen Kunden zu (Bild 1).
2. Geben Sie das Startdatum und die geplanten Stunden im Register "Controlling" an.
3. Legen Sie neue Projektpositionen an, indem Sie auf "neue Projektpos." klicken.
4. Erfassen Sie im Projektmanagement die Vorgänge (Bild 2).
5. Klicken Sie auf den Button "Start Arbeit", um eine neue Zeiterfassung zu starten.
6. Klicken Sie auf den Button "Stop Arbeit", um die Zeiterfassung zu stoppen. Bitte geben Sie im Kommentar an, was sie getan haben.
7. Über den Button "Zusatz anfügen" können zusätzlich Texte, Informationen etc. an den Projektvorgang anfügt werden. Diese werden dann mit Datum und Zeit erfasst.
8. Im Register "Vorgang" können Sie Einstellungen wie Termin, Priorität oder auch den Grad der Bearbeitung eintragen. Desweiteren können Sie hier Notizen erfassen (Bild 3).

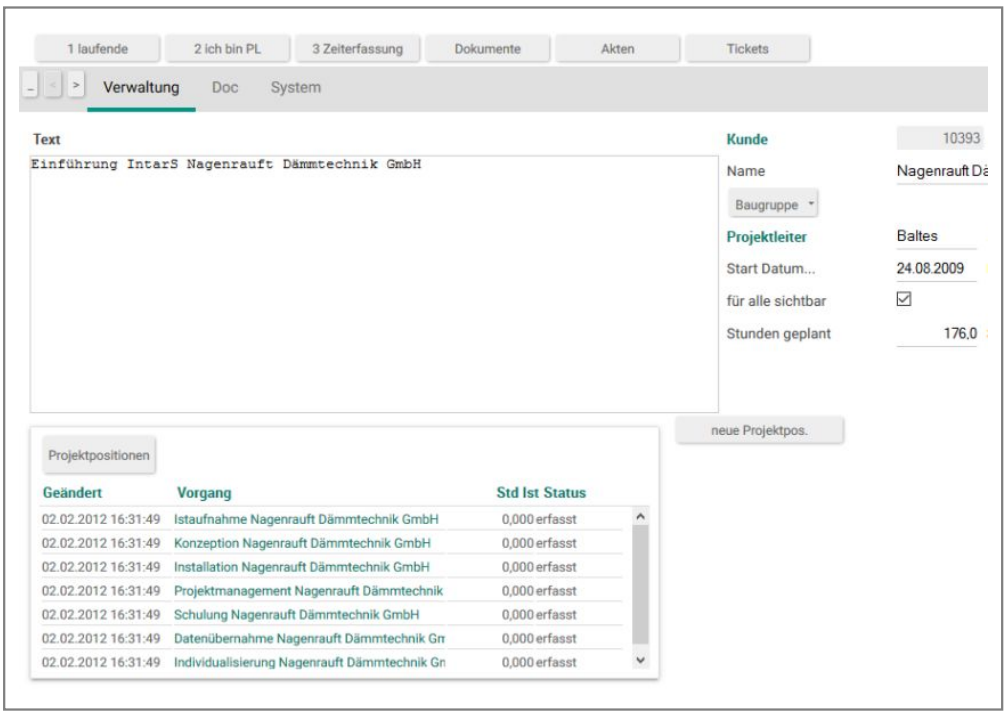


Bild 1: Ein Projekt anlegen.

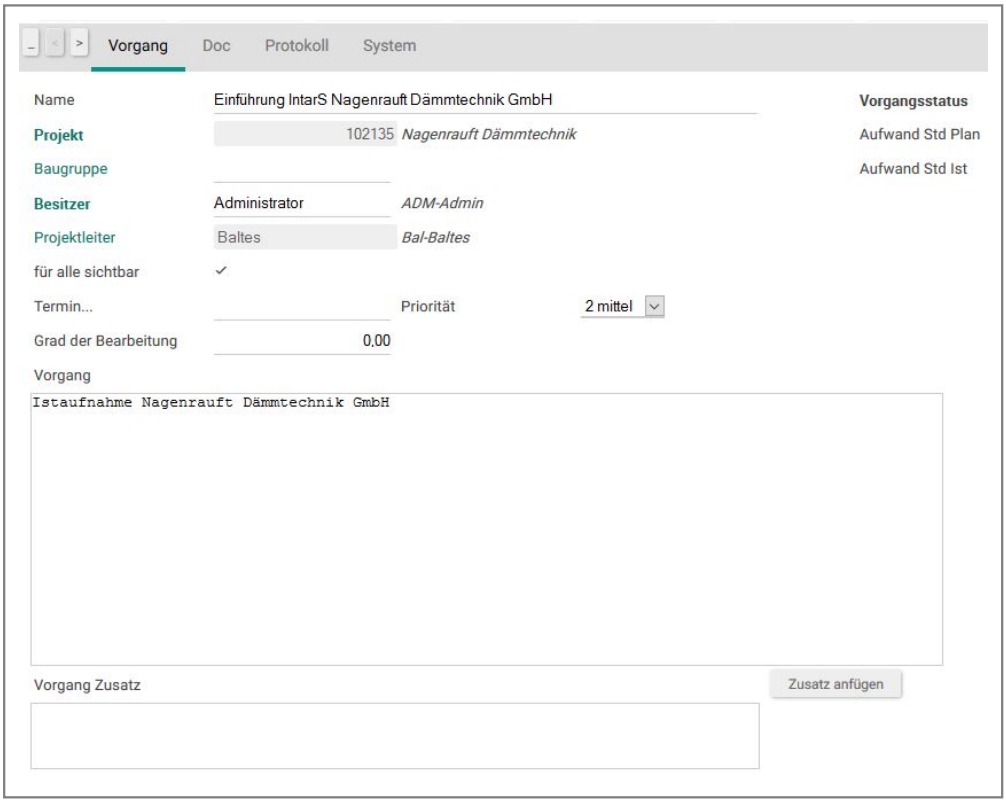


Bild 2: Projektmanagement

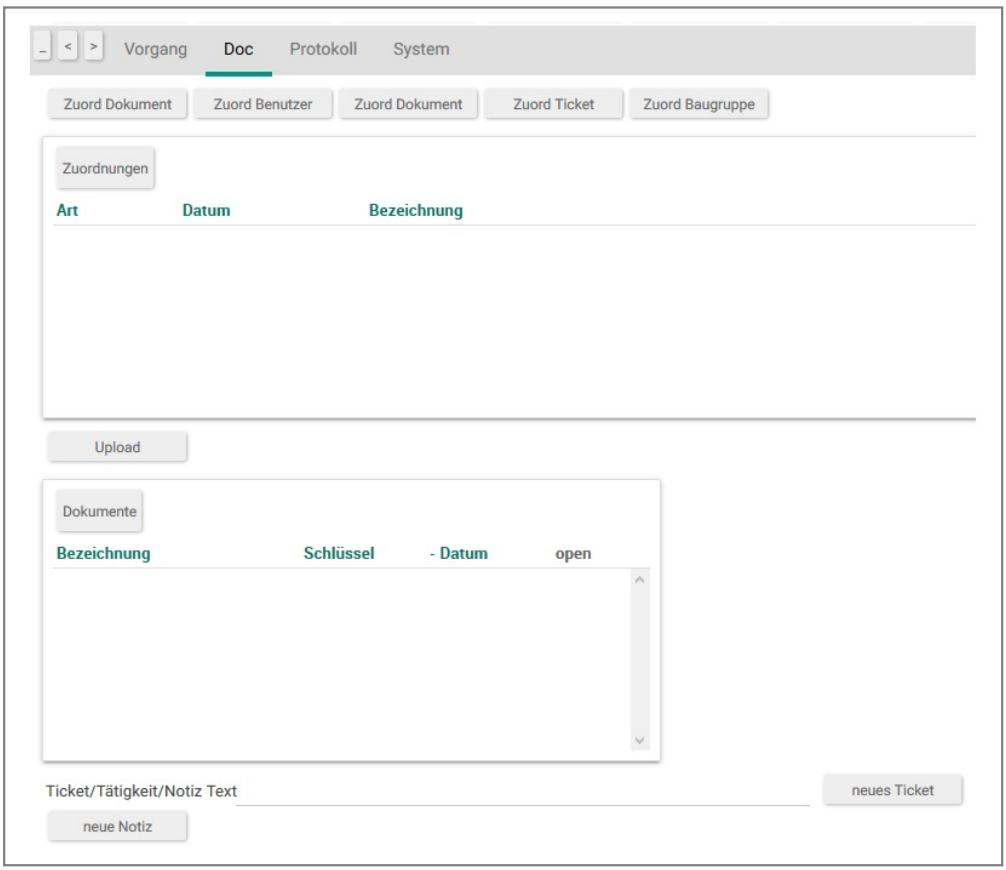


Bild 3: Projektdokumente

7.18 CRM

Kundenbeziehungsmanagement oder Kundenpflege (CRM, engl. Customer Relationship Management) bezeichnet die Dokumentation und Verwaltung von Kundenbeziehungen.

Die Ziele sind Neukundengewinnung, Bestandskundenpflege und Kundenrückgewinnung.

Neukundengewinnung

Bevor man Kunden hat, benötigt man zuerst Adressen. Diese erhält man aus verschiedenen Quellen: Visitenkarten auf Messen und Veranstaltungen sammeln, Lektüre einschlägiger Zeitungen oder Zeitschriften, gelbe Seiten, Adresshändler, IHK, Internet etc. Interessenten melden sich aber auch von selbst (sog. passiver Vertrieb). Die Adressen werden in IntarS 7 eingepflegt bzw. über das Import-Modul eingelesen. Diese neuen Adressen werden zunächst gegliedert, indem ihnen Stichworte zugeordnet werden. Damit dokumentiert man, woher die Adresse kam, für welche Produkte sich der Kunde interessiert, wie lukrativ er ist, welches Potential er hat usw.; eben alles, worauf es bei der weiteren Steuerung der Vertriebsaktivitäten ankommt. Im volltextsuchfähigen Memofeld einer jeden Adresse können sämtliche Informationen unstrukturiert abgelegt werden, die man über die Adresse hat. Hat man noch weitere Unterlagen zur Adresse, können diese über das Dokumentenmanagement eingebracht und zugeordnet werden. Beziehungen zwischen Adressen werden ebenfalls über Zuordnungen dokumentiert.

Regelmäßig werden die neuen Adressen entsprechend der vorgenommenen Gruppierung einer ersten Kontaktaufnahme unterzogen. Das kann ein Telefonat, E-Mail, Fax, Anschreiben o. ä. sein. IntarS 7 unterstützt alle diese Kommunikationsformen. Inhalt und Kontext der Kommunikation werden dabei in einer Akte festgehalten, welche der betroffenen Adresse fest zugeordnet ist. Im Verlauf der Kundenbeziehung wird es zu weiteren Kommunikationsvorgängen kommen, die entweder eine bestehende Akte erweitern oder eine neue hinzufügen. So ist lückenlos dokumentiert, mit welchem Ergebnis etwas zu einem bestimmten Zeitpunkt

kommuniziert wurde.

Eine Kommunikation kann auch vom Kunden initiiert werden, z. B. durch einen Anruf oder eine E-Mail. Diese werden auch in IntarS 7 dokumentiert. Bei einem Anruf wird mit einem Klick eine Telefonnotiz erzeugt und das Gespräch darin notiert. Sie erscheint im Plugin auf der Startseite des zugehörigen Kundenbetreuers. Dann klickt er darauf und kann sie bearbeiten. Gibt es nichts zu tun, setzt dieser den Status auf "erledigt". Aus der Notiz kann eine Aufgabe, ein Termin oder eine Wiedervorlage gemacht werden und taucht entsprechend in anderen Plugins und/oder im Kalender auf.

Davon unabhängig sollten Sie sich überlegen, wie Sie vorgehen möchten. Hier gibt es je nach Branche und Unternehmen unterschiedliche Verfahren. IntarS 7 schreibt nichts vor, sondern bietet Werkzeuge an, ein Verfahren zu implementieren. Die Werkzeuge heißen Recherche, Stichworte, Akten, Wiedervorlage, Termine, Verteilerlisten und Anschreiben (auch per Fax oder E-Mail).

Ein denkbares Verfahren ist, dass eine Adresse die Stadien "neu", "Erstkontakt", "Interessent", "Präsentation", "Angebot" und "Kunde" durchläuft. Jederzeit kann es eine Abzweigung in Richtung "kein Interesse" geben. Diese Zustände dokumentiert man durch die Zuordnung ebensolcher Stichwörter. IntarS 7 bietet Funktionen, nach Stichwörtern und Kombinationen von Stichwörtern zu suchen und zu recherchieren. Mit weiteren Stichwörtern kann man bei Bedarf dokumentieren für welches Gebiet, welche Produkte, Kulturkreise oder Budgets die Adresse wichtig ist. Außderdem ist in dem Verfahren festgelegt, was mit den Adressen, die einen bestimmten Zustand haben, geschehen soll. Beispielsweise könnte es so sein, dass allen Erstkontakten ein Anschreiben geschickt wird. Wer nach zwei Wochen nicht oder negativ reagiert hat, wird zu den Adressen aussortiert, welche "kein Interesse" haben. Die zwei Wochen überwacht man mit Wiedervorlage. Es lassen sich auch Plugins anlegen, die alle Adressen zeigen, welche momentan für eine bestimmte Aktion in Frage kommen. Wer positiv reagiert, aber nicht gleich konkret etwas bestellt, wird zum "Interessent". Für Interessenten sind weitere Aktionen jeweils wieder mit Zeithorizont vorgesehen. Dies sind typischerweise Beratungsgespräche oder die Zusendung detaillierter Unterlagen. Sobald ein "Interessent" etwas gekauft hat, wird er zum "Kunden". Es ist ein guter Gedanke, das gesamte Kundengewinnungsverfahren samt Arbeitsanweisungen zu dokumentieren und im System in das Dokumentenmanagment abzulegen. Präsentationen wird man bei Interessenten machen, bei denen es um ein größeres Volumen geht. Man hatte schon einiges im Vorfeld miteinander zu tun und hat dies auch dokumentiert, kann man sich bei der Präsentation auf den Interessenten einstellen. Entscheidet sich der Interessent nicht schon bei der Präsentation für das Produkt, wird man ihm ein oder mehrere Angebote ausarbeiten. Um ein möglichst maßgeschneidertes Angebot erstellen zu können, greift man auf all die Informationen zurück, die man bisher gesammelt hat und welche in Form von Akten an der Adresse hängen. Hat er auch nach mehreren Angeboten oder einigen Monaten nicht unterschrieben, wird man ihn leider doch unter "kein Interesse" ablegen müssen. Gelegentlich kann per Recherche überprüft werden, welche Interessenten in welcher Phase aus welchem Grund verloren wurden, um doch noch einmal nachzuhaken. Vor allem, wenn mittlerweile Ereignisse eingetreten sind, die eine neue Situation ergeben.

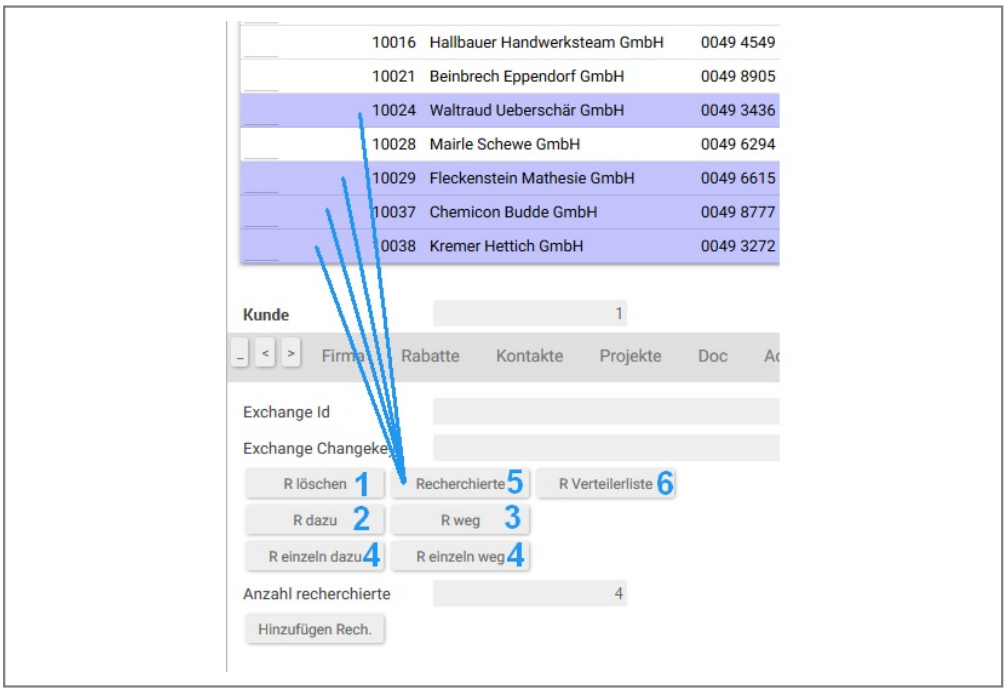
Bestandskundenpflege

Weil IntarS 7 eine angeschlossene integrierte Warenwirtschaft hat, geht es nahtlos über. Befinden Sie sich in einem Kundenstammsatz, haben Sie alle Informationen zu diesem Kunden komplett beieinander: die vollständige Historie, umfassende Kommunikationen, alle Geschäftsvorfälle, Angebote, Aufträge, Rechnungen und Zahlungen. Ruft der Kunde an, sind Sie sofort im Bilde und können fundiert mitreden. So wie man einen Neukundengewinnungsplan hat, sollte man auch einen Bestandskundenpflegeplan erstellen. Überlegen Sie sich, welche Aktionen (Newsletter, Gutscheine, Produktinformationen, Boni, Geschenke, Anschreiben...) für welche Kunden vorgesehen sind. Beispielsweise sollen alle Kunden, die Produkt A bezogen haben, informiert werden, dass es ein verbessertes Nachfolgeprodukt A1 gibt. Oder es erhalten alle Kunden, deren Umsatz in den letzten sechs Monaten deutlich gefallen ist, einen Gutschein. A-Kunden bekommen an Weihnachten z. B. ein Geschenk. Das Mittel, die Zielgruppe für eine Aktion zu identifizieren, ist die Recherche.

7.19 Recherche

Recherche bedeutet, in mehreren aufeinander aufbauenden Suchschritten eine Ergebnismenge von Kunden zu ermitteln, z. B. um diesen einen Brief zu schicken. Man geht dabei wie folgt vor:

1. Planen Sie die Recherche sorgfältig, seien Sie sich im Klaren darüber, welche Kunden Sie suchen.
2. Rufen Sie das Modul "Kunde" auf und klicken Sie auf das Register "Recherche".
3. Löschen Sie bitte zuerst die letzte Recherche ("R löschen") (1).
4. Die Treffermenge wird der Recherche hinzugefügt ("R dazu") (2).
5. In beliebig vielen weiteren Schritten können Sie weitere Suchvorgänge durchgeführen. Das Ergebnis können Sie der Recherche hinzugefügen ("R dazu"), aus ihr entfernen ("R weg") oder mit der bestehenden Recherche ("R und") verknüpfen (3).
6. Einzelne ausgewählte Sätze können Sie manuell der Recherche hinzugefügen oder aus ihr entfernen (4).
7. Aus einer Recherche können Sie eine Verteilerliste erzeugen ("R Verteilerliste"), die wiederum als Datenquelle für Anschreiben dienen kann (6).
8. Sie können auch allen recherchierten Kunden ein (neues) Stichwort zuordnen, um sie später leichter identifizieren zu können.
9. Kunden, die sich in der Recherche-Menge befinden, werden blau dargestellt. Mit dem Button "Recherchierte" (5) können Sie die Anzeige auf die Recherche-Menge eingeschränken.



Recherche

Jeder Benutzer hat seine eigene Recherche-Ergebnismenge, die auch nach dem Abmelden erhalten bleibt. Technisch ist dies als Datenbanktabelle pro Benutzer realisiert, welche die recherchierten Kundennummern enthält.

Wieviele Kunden sich in der Recherche-Menge befinden, wird als "Anzahl recherchierte" angezeigt.

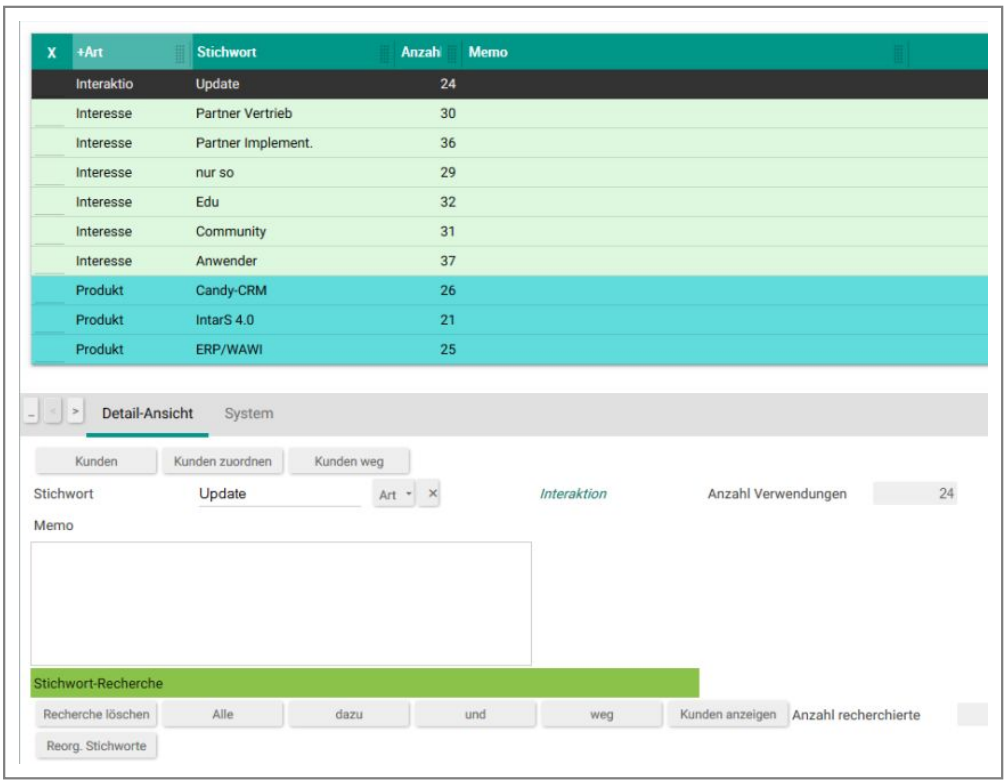
Kunden aus Verteilerlisten können ebenfalls der Recherche-Menge hinzugefügt werden.

Stichwort-Recherche

Ordnen Sie Stichworte dem Kunden zu, um sie zu klassifizieren und mit Eigenschaften zu versehen. Nach diesen Stichworten kann ebenfalls recherchiert werden. Angenommen, man möchte alle Kunden, die bei den Grillparties 2009 und 2010 dabei waren, dabei aber die Familienmitglieder ausnehmen und von den verbliebenen nur die Hundehalter ermitteln. Gehen Sie folgendermaßen vor:

1. Rufen Sie zunächst das Modul "Stichworte" auf; evtl. mit Hilfe des Buttons "zu den Stichworten" im Modul "Kunde".
2. Löschen Sie bitte zuerst die letzte Recherche (Stichwort-Recherche), indem Sie auf "Recherche löschen" klicken.
3. Markieren Sie "Candy-CRM" und klicken auf "dazu".
4. Markieren Sie "Anwender" und klicken auf "dazu".
5. Markieren Sie "Update" und klicken Sie anschließend auf "weg".
6. Markieren Sie "Community" und klicken auf "und".
7. Beachten Sie, wie sich die "Anzahl recherchierte" verändert.
8. Klicken Sie nun auf "Kunden anzeigen".

Mit einem Klick auf "Kunden anzeigen" wird eine Suche im Kundenmodul durchgeführt. Das Ergebnis kann in der Kunden-Recherche weiterverwendet werden. Stichwort- und Kundenrecherche sind zwei verschiedene, von einander unabhängige Recherchen.



Stichwort-Recherche

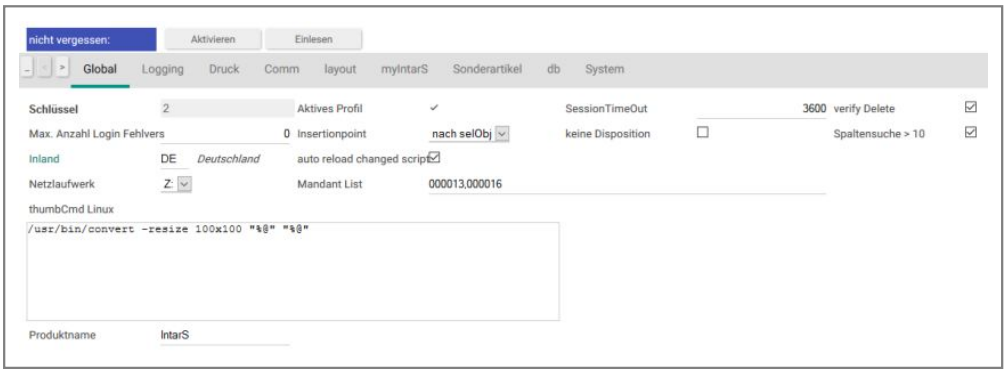
Ein weiteres Beispiel:

Sie möchten alle Kunden ermitteln, die keinen Hund halten.

1. Rufen Sie das Modul "Stichworte" auf; evtl. mit Hilfe des Buttons "zu den Stichworten" im Modul "Kunden".
2. Löschen Sie die letzte Recherche (Stichwort-Recherche), indem Sie auf "Recherche löschen" klicken.
3. Klicken Sie auf "alle".
4. Markieren Sie "Hundehalter" und klicken anschließend auf "weg".
5. Klicken Sie nun auf "Kunden anzeigen".

8 Systemeinstellungen

8.1 Config-Modul



Das Config-Modul

Im Config-Modul können Administratoren systemweit Einstellungen vornehmen. Diese gliedern sich in die Bereiche:

Globale Steuerung

Schlüssel, Aktives Profil

Der primäre Schlüssel des Konfigurations-Profils. Es können mehrere Profile angelegt werden. Um eines zu aktivieren, verwenden Sie den Button "aktivieren". So kann leicht die Konfiguration geändert (z. B. Produktivbetrieb, Testbetrieb) bzw. Änderungen protokolliert werden.

Schnellstart

Die Anwendung startet schneller, wenn einige ressourcenverbrauchende Features, z. B. BLZ/Ktoprüfung und Fontmetriken von Unicode-Zeichensätzen, abgeschaltet werden. Meist kann darauf verzichtet und dieses Flag markiert werden.

SessionTimeOut

Gibt die Default-Inaktivitäts-Zeit in Sekunden an, nach der eine Session vom System gelöscht wird. Im Benutzerprofil kann eine davon abweichende Zeit für den jeweiligen Benutzer hinterlegt werden.

verify Delete

Erfordert die Bestätigung einer Rückfrage vor dem Löschen.

keine Disposition

Schaltet alle Dispositionsfunktionen aus, was den Workflow vereinfacht, aber zu negativen Lagerbeständen führen kann.

Max. Anzahl Login Fehlversuche

Hier kann eine maximale Anzahl an Login-Fehlversuchen festgelegt werden. Standardmäßig ist "0" angegeben, was bedeutet, dass es keine Begrenzung gibt.

Insertionpoint

Hier wird festgelegt, an welcher Stelle eine neu angelegte Position angeordnet wird.

XHTML rendern

Oberfläche wird mit XHTML anstatt HTML dargestellt.

nur https

Sicheres Protokoll zur Übermittlung von Daten.

Fetch Limit

Stellt die Anzahl der Datensätze ein, die standardmäßig mit einem Suchaufruf maximal gelesen werden. Dieser sog. Arbeitsvorrat sollte nicht zu groß, wegen Performanceeinbußen, und nicht zu klein sein, sodass man beim Blättern nicht zu oft nach-fetchen muss.

Anzahl Berechtigungsgr.

Stellt die Anzahl der Berechtigungsgruppen ein. Wenn nichts anderes eingegeben wurde, sind es standardmäßig 10 Gruppen.

thumbCmd Linux

Das Shell Kommando, um aus einem Bild ein Thumbnail zu erzeugen; unter Linux:

```
/usr/bin/convert -resize 100x100 "%@" "%@"
```

thumbCmd Windows

Das Shell Kommando, um aus einem Bild ein Thumbnail zu erzeugen; unter Windows z. B.:

```
c:/Programme/ThumbNailer/thumb7n /h /f "%@" "%@" /n! /o 3 /s 100 100 /r 0
```

Produktname

Wird im Hilfe-System und beim Web-Publishing anstelle von $ {product} eingesetzt.

Logging

Log Feldwert-Änderungen

Wenn es aktiv ist, werden sämtliche Editiervorgänge, die durch Feldänderungen im Browser hervorgerufen werden, geloggt. Dies hilft herauszufinden, warum ein Satz durch Anklicken als geändert erscheint. Häufige Quellen ungewünschter Änderungen sind Zeilenumbrüche in einzeiligen Feldern (wird vom Browser abgeschnitten), unmotivierte Umformatierungen in Memo-Feldern durch den Browser (einfügen bzw. entfernen von 0xA, 0xD), ungültige Werte in Feldern mit Auswahllisten (Feldinhalt wird gelöscht). Sollte nur zu Debuggingzwecken eingeschaltet werden, da es viel Log-Output erzeugt.

log actions

Steuert, dass alle Benutzer-Interaktionen mitgelogged werden. Erzeugt viele Daten. Im Falle eines Fehlers kann aber nachvollzogen werden, was davor geschah.

log ScriptNames

Logt den Start und das Ende von Skripten, mit Name und Verschachtelungstiefe.

SQL debug

Logt alle Befehle der Datenbank mit, erzeugt aber viel Output. Jedoch kann man genau sehen, welche Befehle die Datenbank erhielt.

ORB debug Stack

Gibt bei executeSQL auch den Stack aus.

Druck

Direkt-Druck

Schaltet den Direkt-Druck ein/aus. Direkt-Druck gibt in bestimmten Situtationen (z. B. Pick-Etiketten) den Druck direkt ohne Vorschau auf einen Drucker aus. Das kann hier für Testzwecke unterbunden werden.

abw. Druckdatum ...

Wird hier etwas angegeben, wird es in Queries als Druckdatum ausgegeben, sonst wird das Tagesdatum gedruckt.

Fonts immer embededden

gs path

Voller Pfad zum Ghostscript, falls nicht im $PATH. Ghostscript wird in bestimmten Fällen benötigt, um PDF für Drucker aufzubereiten (PDF nach PS, Fontembedding).

allgemeiner Fußtext

Hier kann ein Text eingegeben werden, der auf allen Belegen am Ende der Seite angedruckt werden soll.

Kommunikation

Server

Name des Servers intern und extern. Es kann hier auch eingestellt werden, wie das interne Subnetz beginnt. Manche Funktionen dürfen nur von Benutzern aus dem internen Subnetz ausgeführt werden. IntarS 7 kann die IP-Adresse eines Benutzers daraufhin abprüfen.

Mail

Einstellungen, wie ein Mail-Server für ausgehende E-Mails erreicht werden kann sowie das Directory, um E-Mails im .eml-Format ins Dokumentenmanagement zu importieren. Unter Linux wird möglichst "mutt" verwendet, unter Windows der freie "blat".

Fax

Einstellungen, wie ein Fax-Gateway für ausgehende Faxe erreicht werden kann. Unter Linux geht es mit dem "sendfax", unter Windows kann der GFI-Faxmaker angesprochen werden.

SMS

Einstellungen, wie ein SMS-Gateway erreicht werden kann, z. B. Team-Msg oder SMS77.

Layout

Dies sind zentrale Einstellungen, um das Verhalten der regelbasierenden Layoutengine für die Oberfläche zu steuern. Bevor Sie hier etwas ändern, sollten Sie das Konzept verstanden und Erfahrungen mit der Oberfläche gesammelt haben.

Raster

Gibt an, wieviel Pixel eine Rastereinheit hat.

Short Fields

Bewirkt, dass auch Felder > 50 Zeichen nicht länger angezeigt werden.

max col. width

Maximale Breite einer Spalte in der Trefferliste in Buchstaben.

rows on page

Gibt an, wieviele Zeilen pro Seite in der Trefferliste zu sehen sind, wenn diese Standardgröße hat.

rows on page long

Gibt an, wieviele Zeilen pro Seite in der Trefferliste zu sehen sind, wenn diese Maximalgröße hat.

rows on page poseditor

Gibt an, wieviele Zeilen pro Seite in der Editierliste des PBWOPosEditor zu sehen sind.

Bez Width Raster

Steuert die Breite der Bezeichnung hinter einer Relation in Rastereinheiten.

Descr Width Raster

Standardbreite der Felddescriptoren in Rastereinheiten.

Edit Width Raster

Breite des Editierbereiches in Rastereinheiten.

Kein Sytem-Tab

Wenn eingeschaltet, haben normale User kein "System"-Register. Im "System"-Register liegen z. B. die Felder letztes Änderungsdatum/User, Anlagedatum/User sowie einige Buttons und Werkzeuge für Administratoren.

max. Anz. Spalten

Legt fest, wieviele Spalten die Trefferliste maximal umfassen soll.

mit Bild-Filnamen

Bild-Dateinamen in Hilfe anzeigen.

Minimize Tableview

Wenn angehakt, wird, wenn eine Zeile angeklickt wurde, die Tabelle ausgeblendet.

Decimal Point

Ist es angehakt, erscheint ".", sonst ",".

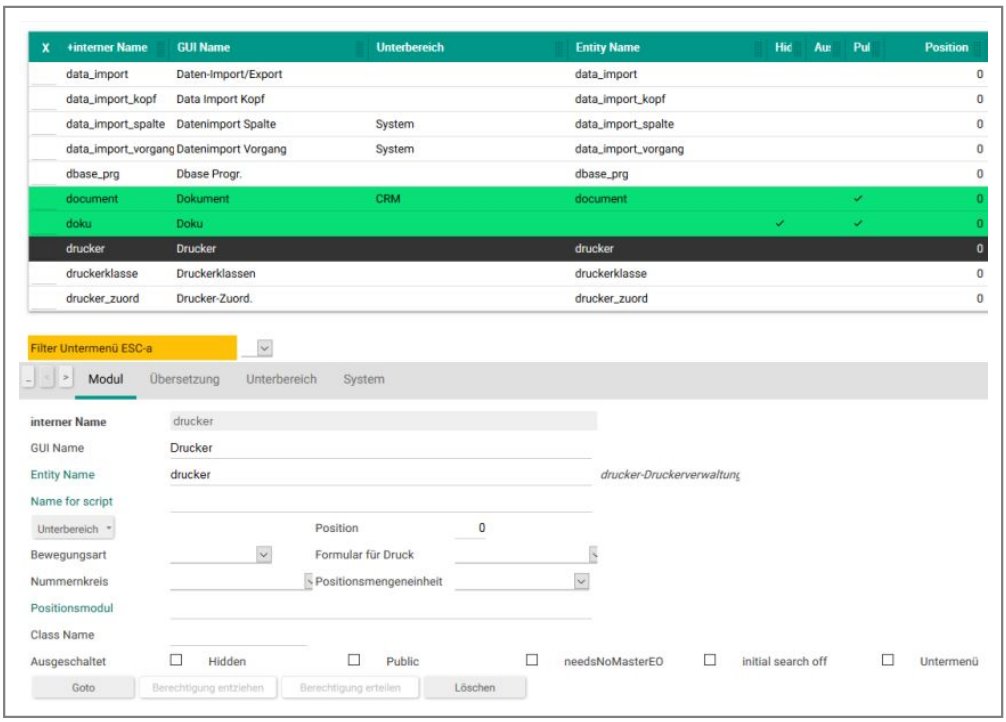
my IntarS

Hier werden die Firmenstammdaten eingetragen. Sie werden standardmäßig am Ende von Belegen angedruckt. Auch ein abweichender Name für das Firmenlogo (default: logo.jpg) kann hier eingetragen werden. Des Weiteren sind hier Datev-Zugangsdaten hinterlegt, sofern die Datev-Schnittstelle verwendet wird.

db

Dies ist die Stelle, um Datenbank-Zugangsdaten anzugeben. Für standardmäßig 4 Datenbanken können Server, User und Passwort angegeben werden. Müssen noch mehr Datenbanken angesprochen werden, lassen sich die Zugänge problemlos vermehren. Trägt man nichts ein, wird als Server "localhost", User "root", Passwort "root" und DB-Name "intars_000201" angenommen.

8.2 Module verwalten



Die Modulverwaltung

Anwendersicht

Ein Modul in IntarS 7 ist synonym zu den Begriffen "Maske", "Bildschirm", "Programm" oder "Dynpro" aus anderen Systemen. Es ist eine logische Einheit für eine in sich geschlossene Bearbeitung. Bei der Arbeit mit IntarS 7 befindet man sich immer in genau einem Modul. Alle Module finden sich im Hauptmenü. Das gerade aktive Modul zeichnet den Bildschirminhalt und nimmt die Benutzerinteraktionen entgegen.

Technik

Technisch ist ein Modul in IntarS 7 nicht ein ausprogrammiertes Stück Code, sondern benennt lediglich eine Klasse, von der eine Instanz zur Laufzeit die Kernfunktionen des Moduls liefert und die von PBWOEditor erben muss. Die Instanzen werden erst bei Bedarf erzeugt. Da sie überwiegend der Klasse PBWOEditor angehören, haben sie alle dasselbe Codesegment, was Hauptspeicher spart. Die Moduldefinition ist eine leichtgewichtige, performante Repräsentation. Instantiierte Module leben innerhalb einer Session, die wiederum pro Benutzeranmeldung erzeugt wird und bei zu langer (einstellbarer) Inaktivität automatisch beendet wird, wobei der Speicher wieder freigegeben wird. Diese Technik trägt dazu bei, dass IntarS 7 mit wenig Systemressourcen auskommt und sehr performant läuft.

Verwaltung

Die Modul-Verwaltung obliegt den Administratoren. In IntarS 7 wird ein Modul gebildet, indem festgelegt wird, von welcher Klasse es zur Laufzeit eine Instanz sein soll. Optional können ein abweichendes Template für die Oberfläche und eine Tabelle aus dem Meta-Datenmodell hinzukommen. Außerdem kann man einstellen, unter welcher Unterrubrik das Modul im Menü erscheinen soll, mit welchem Kürzel man es per Tastatur aufrufen will, wie es an der Oberfläche heißen soll sowie ein paar weitere Steuerungsflags. Module werden in der Datenbank verwaltet.

Module können mit Berechtigungen (rollenbasierendes Berechtigungsmodell) versehen werden.

Ein Modul muss auf der Klasse PBWOEditor oder einer ihrer Subclasses basieren. PBWOEditor ist auch die Default-Klasse, die verwendet wird, wenn nichts anderes angegeben ist. Wer kann, kann auch für ganz individuelle Module eigene Klassen in ObjC programmieren. In IntarS 7 ist z.B. das Kalender-Modul ein solches.

Nach dem Verwalten der Module muss die Anwendung neu gestartet werden.

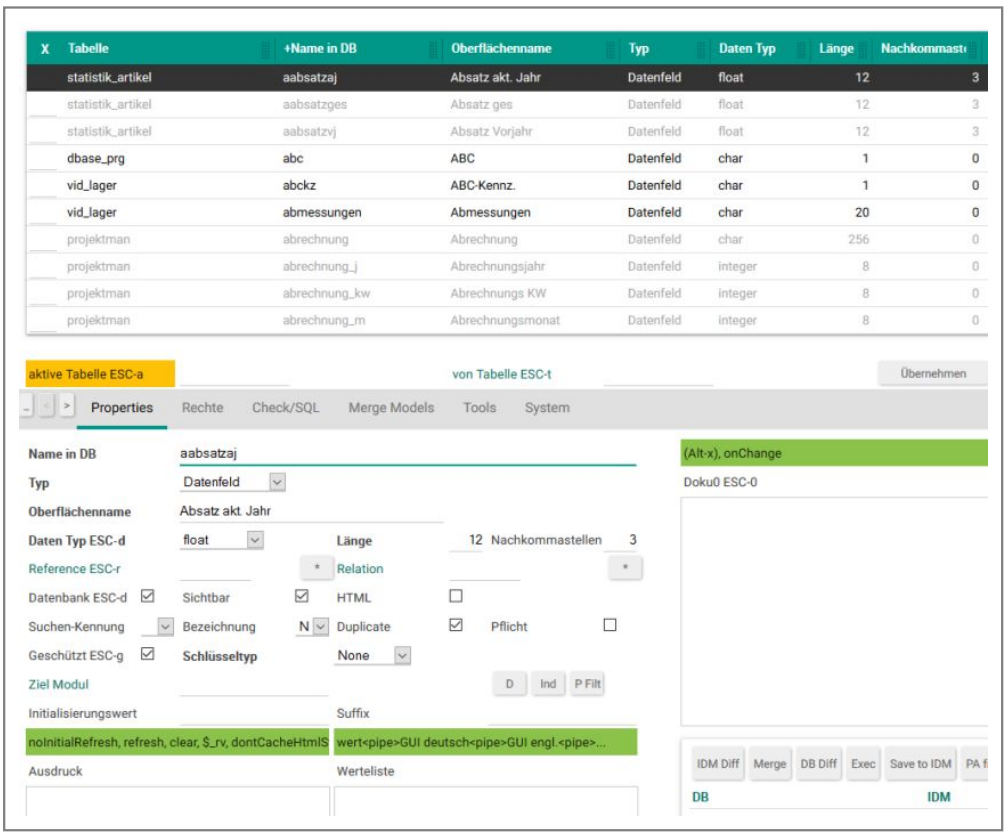
PBWOEditor

Diese Klasse ist das Arbeitspferd in IntarS 7. PBWOEditor stellt sicher, dass das Modul vom Laufzeitsystem verwendet werden kann (Menü, Oberfläche, Aufruf, Suchen, Berechtigungen, elementare Events, Navigation). Die Oberfläche eines Moduls bestimmt sich nach seinem Template (htmlwod File). Benennt man kein spezielles Template, wird ein Template gesucht, das genauso heißt wie das Modul. Wird keines gefunden, wird das Default-Template verwendet. 90% aller Module tun dies. Ein Template enthält das HTML-Gerüst mit variablen Elementen, deren Inhalt sich über die Bindings zur Laufzeit bestimmt. Das ist das Prinzip von WebObjects und auch anderer Template-Systeme, wie z.B. ClearSilver. PBWOEditor kann eine Tabelle kennen (abstract oder real), aufbauend darauf eine reichhaltige Verwaltungsoberfläche anbieten und die Infrastruktur des Frameworks nutzen. 90% aller Module verwalten eine Tabelle, nur wenige Spezialmodule wie z.B. LoginUser, Doku oder Kalender tun dies nicht. PBWOEditor bedient auch alle Events, um sein Verhalten anzupassen. Typischerweise wird man diese Klasse nehmen (indem man nichts Abweichendes einträgt), wenn man ein neues Modul definiert. Man erstellt im Meta-Datenmodell eine neue Tabelle und legt in der Modulverwaltung ein neues Modul an, um die Tabelle zu verwalten.

PBWOViewer

Dies ist eine Subclass von PBWOEditor, die nur Daten zeigt, aber nichts ändern oder löschen kann. Da man mehrere Module auf dieselbe Tabelle aufsetzen kann, ist es möglich, gleichzeitig ein Verwaltungs- und Anzeige-Modul zu haben, die verschieden berechtigt werden können.

8.3 Meta-Datenmodell verwalten



Ausschnitt aus der Modellverwaltung

Das Meta-Datenmodell (auch Repository oder DataDictionary) ist ein zentraler Bestandteil von IntarS 7. Daraus leiten sich die Datenbankstruktur, Oberflächen-Elemente und Constraints ab. Wie alles in IntarS 7 ist es frei verwaltbar. Da es so groß ist, wird es über Datenbanktabellen verwaltet. Für IntarS 7 und verteilte Entwicklung mit Subversion wird es zusätzlich in einem Textfile "Mandant.idm" gespeichert. Nach Editierungen in der Modulverwaltung muss es daher mit dem Button "DB to IDM" exportiert werden. Von IntarS 7 wird das exportierte Modell beim Start geladen und interpretiert. Zur Beschleunigung späterer Startvorgänge wird ein binäres Archiv des Modells geschrieben. Ist so ein Archiv vorhanden, wird dieses verwendet. Hat aber das exportierte Textfile (*.idm) ein neueres Datum, wird das Archiv verworfen und neu erstellt.

Das Metadatenmodell strukturiert sich in Tabellen (Master) und Attribute (Detail). Obwohl diese Begriffe aus dem Datenbankbereich entlehnt sind, müssen sie abstrakter verstanden werden, eher im Sinne des Entity-Relationship Modells. Eine Tabelle ist eigentlich nur eine Sammlung von Attributen. Tabellen, die tatsächliche Datenbanktabellen erzeugen sollen, werden als "real" tables definiert, das andere sind "abstract" tables. Standard-Tabellen sollten von Mandanten nicht verändert, allenfalls erweitert werden. Sie enthalten hauptsächlich Systemdaten wie z. B. das Modell selbst, die Module, Benutzer etc. Eine unsachgemäße Änderung würde die Systemintegrität gefährden. Zusätzliche Felder hinzuzufügen ist jedoch nicht kritisch.

Unter dem Dach einer Tabelle werden deren Attribute verwaltet. Dies können sein:

TVCSelObj - Normale Felder

Ziel-Typ "TVCSelObj" (TableViewController selected Object ist Datenquelle). Der Name TableViewController ist historisch bedingt. In der Vergangenheit, als IntarS 7 unter NeXTSTEP, dann OpenStep als Desktop-Anwendung lief, gab es eine Klasse namens "NSTableView" im AppKit. Damit erstellte man Tabellen für die grafische Oberfläche. Im Zuge des Umbaus auf WebApplicationServer-Architektur wird diese Klasse seit dem Jahr 2001 von Methoden des PBWOEditor nachgebildet. Dieser Feldtyp ist der häufigste. Er repräsentiert ein Datenbankfeld, wenn das Datenbank-Flag aktiv ist. Einzustellen gibt es den Datentyp, Länge, Nachkommastellen, evtl. Primärschlüsseltyp, Constraints (Pflichtfeld, duplizierbar, sichtbar, geschützt), Hints

(Combi-Suchfeld, Bezeichnung). Jedes Feld hat zwei Dokumentationen, auf deutsch und englisch. Die Eigenschaften im einzelnen:

Datenbankname

Interner Name des Attributes. Hierüber kann es in Skripten eindeutig innerhalb der Tabelle adressiert werden. Liegt das Feld in der Datenbank, heißt es auch dort so. Der Datenbankname muss kleingeschrieben sein und sich an die Konventionen für Feldnamen in MySQL halten. Auch in IntarS 7 gibt es Konventionen:

- bu_.... für Buttons
- p_... für Parameterfelder
- po_... für Plugins
- masterkey zwingend für eine Master-Detail Relation

Oberflächenname

So heißt das Feld an der Oberfläche. Oberflächenname1 ist für die englische Übersetzung. Der Oberflächenname unterliegt keinen Beschränkungen und kann frei geändert werden.

Reference

Wird hier der Name einer abstract table eingetragen, wird das Attribut zur Laufzeit durch die Attribute der abstract table ersetzt. Diese kann übrigens wieder Referenzen auf weitere abstract tables haben usw. Es wird rekursiv aufgelöst.

Relation

Dient dazu, eine Relation zu einer anderen Tabelle herzustellen. Das Attribut enthält dann den Fremdschlüssel. Über das Framework kommt die Logik hinzu, dass der PBWOEditor einen Link zum Zieldatensatz, Validierung und Nachschlagefunktion beisteuert. In IntarS 7 haben alle Tabellen einen Primärschlüssel, der nur aus einem Feld besteht. Es gibt keine zusammengesetzten Schlüssel. Somit kann man eine Relation zu einer anderen Tabelle immer über ein einziges Attribut herstellen. Der Primärschlüssel einer Tabelle sollte nur dazu dienen, einen Datensatz eindeutig zu identifizieren. Sog. sprechende Schlüssel sind ein Relikt aus den Anfängen der IT.

Datentyp, Länge, Nachkommastellen

Aus der großen Vielzahl möglicher Datenbankdatentypen wird von IntarS 7 eine sinnvolle Untermenge zur Verwendung angeboten. Character werden in der Datenbank zu varchar. Länge gibt die maximale Eingabelänge im Feld an. Gibt man eine Länge größer 255 an, entsteht daraus ein Memo-Feld von 64 KB. An der Oberfläche erscheint es als Textarea.

Bool-Felder enthalten eine Ja/Nein-Information. An der Oberfläche erscheinen sie als Häkchen, in der Datenbank als char(1) Feld mit der Repräsentation "J" für wahr und "N" für falsch.

Datums- und Date-Time-Felder bekommen durch das Framework eine Nachschlagefunktion auf den Firmenkalender. Er wird zur Auswahl eines Termins aufgeblendet, wenn man am Ende einen "." eingibt und Return drückt. Datumsfelder fungieren auch als Rechenfelder, siehe den entsprechenden Hilfeeintrag.

Float nimmt man für alle numerischen Felder mit Nachkommastellen, Money für solche mit zwei Nachkommastellen. Die Länge bei float Feldern meint die mögliche Gesamtzahl der Ziffern.

Für ganzzahlige Werte steht der Integer-Datentyp zur Verfügung. Er sollte möglichst vorrangig vor dem float-Typ genommen werden, da er besonders wenig Resourcen in der Datenbank und bei der Verarbeitung in Skripten benötigt. Auch für Primärschlüssel ist der Integer-Datentyp sehr ansprechend, weil darauf effizient Indexe gebaut werden können. Bei Joins verbrauchen diese wenig Speicher. Insgesamt ergibt sich eine bessere Performance bzw. geringere Datenbankbelastung.

Initialisierungswert

Initialisiert das Feld mit diesem Wert, wenn ein neuer Datensatz erzeugt wird.

Combi-Suchfeld

Vergibt einen Rang, der ein Feld in die Reihenfolge der Combi-Suchfelder einreiht. Combi-Suchfelder spielen die entscheidende Rolle bei der generischen Suche. Wird im Combi-Suchfeld eines Moduls ein Suchbegriff eingegeben, wird sukzessive über die Combisuchfelder zunehmend unschärfer gesucht bis eine akzeptable Trefferanzahl ermittelt wurde. Das System macht von sich aus Indexe auf Combisuchfelder. Memofelder werden dabei mit ihren ersten 60 Stellen indiziert.

Datenbank

Ein Feld liegt in der Datenbank. Felder, die nicht in der Datenbank liegen, müssen ihren Wert über ein Skript, eine Expression oder vom Modul beschaffen.

Bezeichnung

Vergibt einen Rang, der ein Feld in die Reihenfolge der Bezeichnungs-Felder einreiht. IntarS 7 zeigt für jeden Datensatz eine Beschreibung an, die sich aus den zusammengeketteten Inhalten ausgewählter Bezeichnungsfeldern zusammensetzt. Welche Felder und in welcher Reihenfolge das sind, kann hier festgelegt werden.

Duplicate

Das Feld wird beim Duplizieren mitkopiert. Auch beim Anlegen einer neuen Position werden Feldinhalte vom Kopf auf die Position kopiert, wenn die Felder gleich heißen und das Duplicate-Flag gesetzt ist.

Pflicht

Dieses Feld ist immer ein Pflichtfeld. Ist es auf dem Bildschirm zu sehen, muss es gefüllt werden. Damit man das vorher schon weiß, hat es einen hervorgehobenen Deskriptor.

Geschützt

Das Feld ist immer geschützt. Es ist nur ein Anzeigefeld und kann nicht editiert werden.

Sichtbar

Das Feld ist prinzipiell sichtbar.

HTML

Das Feld kann html-Daten enthalten, die unescaped an die Oberfläche gelangen sollen.

Schlüsseltyp

Maximal ein Feld pro Tabelle kann als Primärschlüssel (eindeutiger Schlüssel, primary Key) definiert werden, indem man ihm hier den Schlüsseltyp setzt. Primary ist ein von Hand vergebbarer Schlüssel. Serial bewirkt, dass die Datenbank den Schlüssel selbst fortlaufend vergibt. Dies ist vorzuziehen, wenn man nicht unbedingt einen besonders aufbereiteten Schlüssel braucht.

Zielmodul

Bei Relationen kann hier ein Hinweis für das Framework gegeben werden, mit welchem Modul der Zieldatensatz geöffnet werden soll, wenn auf den Link geklickt wird.

Root

Dieses Feld oder Button sieht nur der Root, egal welche anderen Berechtigungseinstellungen vorliegen.

Ausdruck

Hier kommt ein Ausdruck, ein Skript oder der Filename eines Skripts hinein. Bei Feldern kann hierüber der Wert ermittelt werden, wenn es sich nicht um ein Datenbankfeld handelt. Eine häufige Anwendung hiervon sind sog. flattened Relations: Es werden Werte aus anderen Tabellen beschafft, die über Relationen zu erreichen sind. Bei Buttons wird hier implementiert, was passieren soll, wenn auf den Button geklickt wird. Ist das ein kleines Skript, kann es direkt komplett hier hinein geschrieben werden. Ansonsten wird das Skript an seine ordentliche Stelle im Filesystem gespeichert und sein Filename hier eingetragen. Es ist gute Konvention, das Skript so zu nennen wie der Datenbankname des Feldes. Das System unterscheidet anhand der Anzahl der Zeilen, ob nur ein Ausdruck (eine Zeile) oder ein Skript (mehr als eine Zeile) vorliegt. Hat man tatsächlich nur ein einzeiliges Skript, z. B. den Aufruf einer Übersicht, muss man durch Eingabe einer folgenden Kommentarzeile die Zeilenzahl auf größer eins bringen. Ist es ein Skript, das den Wert des Feldes ermittelt (nicht nur eine Expression), muss der ermittelte Rückgabewert in die Variable $_lastMethodReturnValue gestellt werden. Skripte, die als Source in diesem Feld stehen, werden zur Ausführungszeit unter dem Kontext des EO als Datasource ausgeführt. Skripte, deren Sourcen in einem File liegen, werden unter dem Kontext des Moduls als Datasource ausgeführt.

Doku0

Dokumentation in deutsch/english. Erscheint bei Buttons als Tooltip. Klickt man in IntarS 7 auf den Namen eines bereits aktivierten Moduls, wird die Modulhilfe angezeigt, die sich aus dem Inhalt dieser Dokumentationsfelder aufbaut. Bei Buttons gibt es noch eine Besonderheit: trägt man in der Doku einen Tastaturbefehl ein, z. B. "(Alt-x)", wird dies vom Laufzeitsystem erkannt und der Button kann mit dieser Tastenkombination ausgelöst werden.

Werteliste

Gibt man hier etwas in der richtigen Syntax ein, erscheint das Feld an der Oberfläche als Dropdown-Liste. Die richtige Syntax sieht so aus:

```
intern|GUI deutsch|GUI english
```

intern ist der interne Wert, wie er in der Datenbank gespeichert und in Skripten abgefragt wird. GUI deutsch ist der Wert wie er an der Oberfläche erscheint und ist frei verwaltbar. GUI english ist optional und erscheint für Benutzer, die in ihrem Stammsatz eine andere als die deutsche Sprache eingestellt haben.

Vor dem exzessiven Gebrauch von Dropdownlisten muss gewarnt werden. Sie haben einige Nachteile:

- Sie können nicht ohne weiteres vom Anwender erweitert werden
- Sie werden als ganzes im HTML übertragen
- Steht der Cursor auf einem Drop-Down Feld, empfängt das System keinen Submit
- Wird ein Drop-Down Feld als erstes Feld auf die Seite platziert, steht der Cursor erstmal dort und es kann leicht passieren, dass man versehentlich einen anderen Eintrag auswählt ohne es zu wollen.
- Enthält ein Feld (z. B. durch eine Schnittstelle versorgt) einen Wert, der nicht in der Auswahlliste enthalten ist, wird der Feldinhalt vom Browser gelöscht und der Satz wird als editiert dargestellt.

Statt Auswahllisten ist es oft besser, eine Hilfstabelle und eine Relation dort hin zu bauen. Auswahllisten machen dann Sinn, wenn es nur wenige Werte auszuwählen gibt und diese sich voraussichtlich nicht ändern.

Feldeigenschaften

Viele der Feldeigenschaften lassen sich zur Laufzeit im Anwendungskontext durch Skripte dynamisch umsteuern. Jedes Attribut, das seinen Inhalt an die Oberfläche bringt, tut dies über ein zwischengeschaltetes Verbindungsobjekt, eine sog. Association. Das Meta-Datenmodell ist statisch, es wird pro Instanz auf Application-Ebene nur einmal instanziiert. Module werden unter Berücksichtigung des jeweils angemeldeten Benutzers auf Session-Ebene instanziiert. Ein Modul, das vom PBWOEditor gesteuert wird, erzeugt dann für jedes berechtigte Attribut seiner Tabelle eine Association, die erstmal die Eigenschaften des Attributes übernimmt, dann aber umgesteuert werden kann. Allerdings kann aus einem geschützten Feld kein editierbares, aus einem unsichtbaren kein sichtbares gemacht werden. Der vom Meta-Datenmodell vorgegebene Freiheitsgrad kann nicht erweitert, aber eingeschränkt werden. Oft werden z. B. Felder so gesteuert, dass sie nur beim Neuanlegen eingabefähig und nach dem ersten Speichern geschützt sind.

Buttons

Für einen Button sind nur dbName, guiName, Doku und vor allem der Ausdruck wichtig. In letzterem liegt das Skript oder dessen Filename und damit die Funktionalität des Buttons. Für Buttons, die immer aktiv sein sollen, gibt es das "always on" Flag.

Plugins

Der dbName ist der Plugin-Name. Über ihn werden die Plugin-Scripts gefunden. Jedes Plugin hat ein ..._htmlString-Script, das die Oberfläche rendert. Fast jedes Plugin hat ein ..._refresh-Script, das die anzuzeigenden Daten beschafft. Buttons in Plugins rufen weitere Plugin-Scripts auf.

Im Feld "Ausdruck" können noch Konfigurationsoptionen eingetragen werden.

- "refresh" sorgt dafür, dass bei jedem selObjChanged-Event die Daten für das Plugin neu beschafft werden
- "clear" löscht alle Plugindaten im selObjChanged-Event
- "noInitialRefresh" unterdrückt auch die initiale Datenbeschaffung
- "dontCacheHtmlString" verhindert, dass der html-String (die fertig gerenderte Oberfläche) gecached wird.

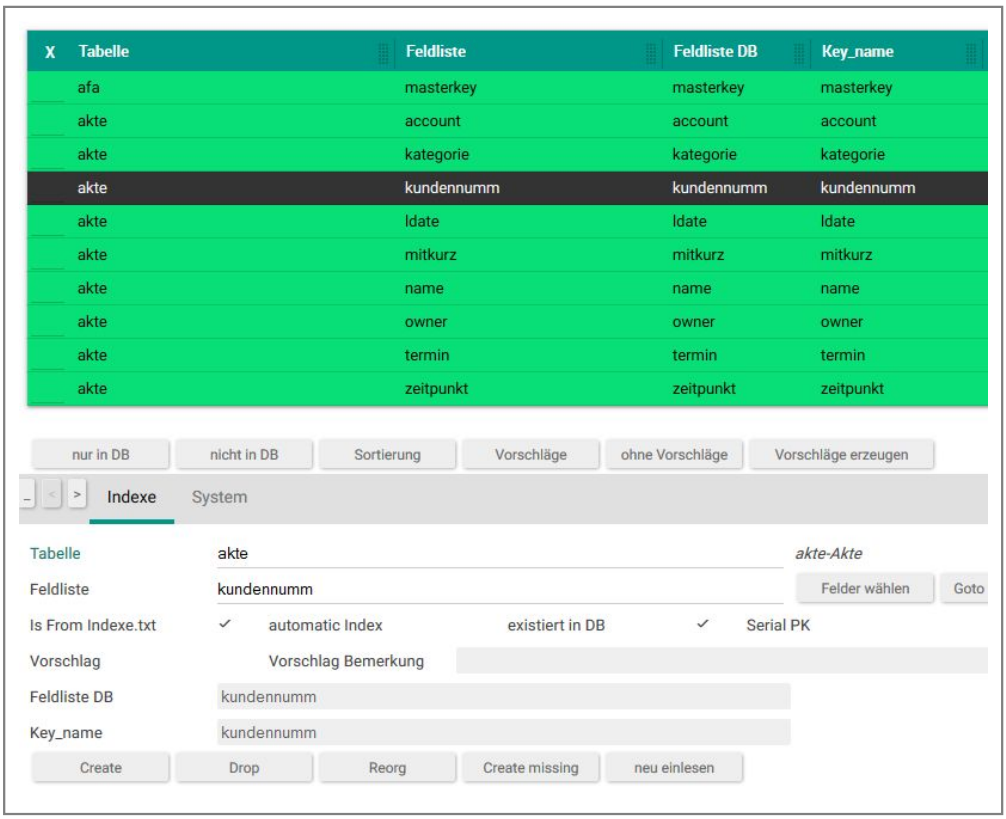
Parameterfelder

Diese fangen per Konvention mit "p_..." an. Sie liegen nicht in der Datenbank und dienen nur dazu, Werte vom Benutzer abzufragen, die dann z. B. in Skripten verwendet werden. Parameterfelder werden vom System im parmDict eines Moduls geführt. Der Inhalt des parmDicts wird vor dem Start eines Skripts in den lokalen Werteraum des Skripts transportiert. So kann das Script auf Parameterfelder zugreifen, als wären es seine eigenen Variablen.

TVC - Datenelemente ohne Datenbankbezug

TVC heißt "Tableview Controller" und meint das Modul. Der Datenbankname eines solchen Attributs muss gleich lauten wie der Name einer Instanzvariable oder wie ein Methodenname des Objektes, welches das Modul implementiert (meist von der Klasse PBWOEditor).

8.4 Indexe verwalten



Indexverwaltung

Sinnvoll eingerichtete Indexe sind essentiell für die gute Performance einer Datenbankanwendung. Es sind Verbesserungen bis in den Bereich von Faktor 100 möglich. Da das so wichtig ist, gibt es eine eigene Index-Verwaltung in IntarS 7. Hier werden manuell Indexe angelegt und Wartungsarbeiten durchgeführt.

IntarS 7 legt schon von sich aus eine Menge Indexe an, die es aus dem Meta-Datenmodell erkennen kann: Primärschlüssel, Master-Detail-Relationen und alle Kombi-Suchfelder. Weiterer Bedarf für Indexe ergibt sich aus dem Benutzerverhalten (Detailsuche und Sortierung) und Zugriffen aus Skripten. Aufgrund dessen, dass man dies teils nicht vorab ahnen kann, unterstützt IntarS 7 den Systemverwalter bei der Findung geeigneter Indexe, indem es Datenbankzugriffe protokolliert, die länger als eine Sekunde dauern.

Zumal IntarS 7 auch auf fremde Datenbanken zugreifen kann (z.B. SugarCRM, OSCommerce für Schnittstellen) und diese auch schon Indexe haben können, wird unterschieden zwischen vom Modell her bekannten und in der Datenbank vorgefundenen Indexen.

Beim Systemstart analysiert IntarS 7 die Datenbank und liest Index-Informationen aus. Zusammen mit den automatischen (vom Metadatenmodell veranlassten) und manuell verwalteten Indexen wird eine Gesamtsicht aufgebaut und in die Index-Datenbanktabelle geschrieben. Diese wird dann mit obigem Index-Modul verwaltet.

An Wartungs-Werkzeugen stehen hier zur Verfügung:

Create

Erstellt den Index in der Datenbank.

Drop

Droppt den Index in der Datenbank. Dabei werden keine Daten gelöscht.

Reorg

Droppt alle Indexe und legt sie neu an. Das ist sinnvoll nach großflächigen Änderungen im Modell mit mehreren automatischen Indexen oder nach dem Anlegen einer größeren Zahl manueller Indexe. Wenn Indexe infolge von Datenübernahmen oder anderer Datenbank-Operationen "unbalanced" geworden sind, werden sie damit wieder in Ordnung gebracht. In den letzten Jahren sind die Datenbanken aber immer besser geworden, so dass das Problem von "unbalanced" Indexen seltener auftritt.

Create missing

Es werden die Indexe in der Datenbank erzeugt, die laut Modell vorhanden sein sollten, es jedoch nicht sind.

Indexe und Datenkbank-Performance

Im Allgemeinen hat man nur ein Feld in der Feldliste. Dies macht den Index universeller verwendbar und ist meist ausreichend. Erstellt man zuviele Indexe, werden Update-Operationen langsamer. In SQL-Statements oder in Qualifiern müssen die am stärksten einschränkenden Bedingungen als erste genannt werden. Bei besonders komplizierten Statements kann es notwendig sein, dem Optimizer Hinweise zu geben, welchen Index er nehmen soll (s. MySQL Doku). Desweiteren kann man sich mit "explain" von MySQL zeigen lassen, welche Indexe verwendet werden und was das an Performance kostet. Es macht nur Sinn, Spalten zu indizieren, die eine große Variabilität besitzen. Flags oder Auswahllisten-Felder zu indizieren, bringt nichts. Ist ein Performanceproblem mit Indexen nicht mehr in den Griff zu bekommen, bleibt nichts anderes übrig, als die Tabelle vertikal oder horizontal zu teilen. Vertikal bedeutet, dass man die Spalten auf verschiedenen Tabellen verteilt (Twin-Segmente). Horizontal hingegen, dass man einen Teil der Datensätze in eine andere Tabelle auslagert (z.B. alte Sätze archiviert). In der MySQL-Konfiguration my.cnf kann noch die Resourcenzuteilung verändert werden. Stellt man nämlich zu wenig Hauptspeicher zur Verfügung, kann der beste Index nichts ausrichten. Für IntarS 7 darf übrigens nur der Datenbanktyp MyIsam verwendet werden. Wer sicher ist, kein Unicode zu benötigen, kann versuchen, statt UTF8 Latin1 einzusetzen, das bringt auch nochmal etwas Performance. Schließlich sollte man eben auch beim Datenbankdesign mit Umsicht vorgehen und vorausschauend Redundanzen einbauen, um Zugriffe zu sparen. Denn der schnellste Zugriff ist der, der nicht stattfinden muss. Dem IntarS 7 zu Grunde liegende Framework bietet eine gute Unterstützung, die Redundanzen konsistent zu halten. Zu guter Letzt kommt es noch auf den Datentyp an, wie effektiv ein Index ist. Bestens geeignet sind Integer-, Datums- und Datetime-Felder.

8.5 Wertelisten verwalten

Die Dropdown-Listen werden im Meta-Datenmodell verwaltet (s. dort). Die entsprechende Berechtigung besitzt nur ein Administrator.

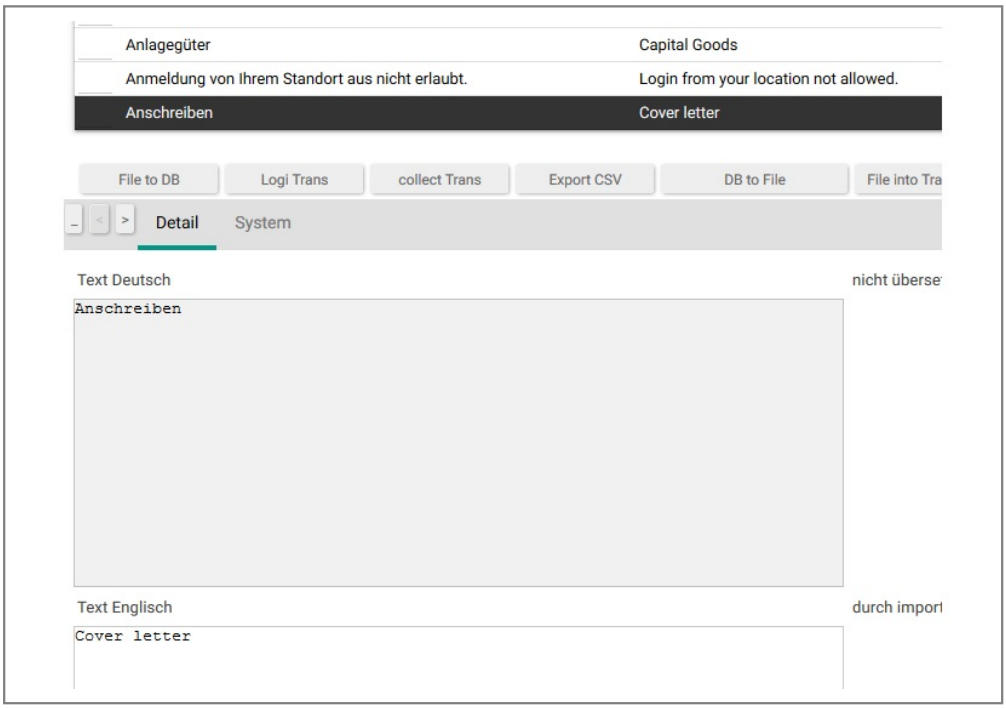
8.6 Übersetzungen pflegen

IntarS 7 ist standardmäßig mehrsprachig ausgelegt. Die Ursprungssprache ist deutsch. Prinzipiell ist eine weitere Sprache vorgesehen, in der Auslieferungsversion Englisch. Die Fremdsprachen-Thematik begegnet Ihnen an verschiedenen Stellen. Es gibt Zielkonflikte zwischen Performance, Einfachheit der Verwaltung und Anzahl der Fremdsprachen.

Zunächst einmal zu den einfach zu übersetzenden Begriffen. Dies sind die Oberflächennamen und Dokumentationen aus dem Meta-Datenmodell. Dort ist jeweils bereits ein Feld für die Fremdsprache vorgesehen. Diese Übersetzungen sind zentral. Sie benötigen wenig Speicher und Overhead. Außerdem werden jene einmalig beim Start der Session eines fremdsprachigen Benutzers ausgewertet und finden sich in den Oberflächennamen der Associations wieder.

Modulnamen können in der Modulverwaltung übersetzt werden.

Zusätzliches liegt außerhalb des Modells: Textkonstanten in Meldungen und Skripten. Für diese muss ein Übersetzungs-Verzeichnis gepflegt werden. Verwenden Sie hierfür bitte das Modul "Übersetzungen".



Übersetzungen

Die zu übersetzenden Begriffe kommen in das Modul "Übersetzungen", wenn das System versucht, einen Ausdruck zu übersetzen, jedoch keine Übersetzung findet. Dies passiert, wenn ein fremdsprachiger Benutzer (Sprache im Benutzerprofil ist nicht Deutsch) sich anmeldet und im System arbeitet. Übersetzte Begriffe werden wie folgt angefordert:

Im Scripting

Durch das Prefix <trans> wird on the fly übersetzt.

In ObjectiveC-Programmierung

Durch das TRANSLATION()-Macro wird on the fly übersetzt.

Hinter den Kulissen

Intern liegen die Übersetzungen in einem zentralen Dictionary im Speicher. Stößt das System auf einen bisher nicht übersetzten Begriff, schreibt es diesen in das Dictionary und gleichzeitig in die Datenbanktabelle, die vom Modul "Übersetzungen" verwaltet wird. Hat man darin die neu aufgetauchten Begriffe übersetzt, muss der Datenbankinhalt mit dem Button "DB to Transdict" exportiert werden. Die Daten werden im SVN-freundlichen Textformat nach /usr/GNUstep/Local/Library/IntarS/_K_000201/ExportedData/translationsm.txt geschrieben. Von dort werden sie zusätzlich beim Systemstart neu geladen, um das Dictionary zu füllen. Für bessere Performance wird das File auch binär archiviert. Dieses Archiv wird ebenso verwendet, wenn sich das zu Grunde liegende Textfile seither nicht geändert hat.

8.7 Backup

Plugin zur Datensicherung

Das Backup-Plugin

Klicken Sie auf das Modul "Home" und dann auf das Register "Service".

Klicken Sie auf "Datensicherung", um die gesamte Datenbank zu sichern. Es wird ein sog. SQL-Dump erstellt. Der Dateiname enthält Datum und Uhrzeit. Alle SQL-Dumps sind im "Backup"-Plugin zu sehen. Klicken Sie auf "Backups", um die Ansicht zu aktualisieren. Um eine Datensicherung wieder einzuspielen, klicken Sie auf "Einspielen" in der entsprechenden Zeile der gewünschten Sicherung im "Backup"-Plugin. Nach einer Bestätigungsrückfrage wird dann die komplette Datenbank durch das Backup ersetzt.

Die weiteren Funktionen dienen der Verwaltung von Backups. Man kann sie zippen, um Platz zu sparen und sie sich per mail schicken oder per scp auf einen anderen Server übertragen zu lassen, unzippen, um sie wieder einspielen zu können. "Delete" löscht das Backup.

Sicherung über Netzwerkfreigabe

Die gesamte Festplatte der IntarS 7-Box bzw. Virtual Appliance ist über Samba (Windows-Freigabe) als "alles" im Netzwerk freigegeben. So können Sie die SQL-Dumps in /usr/GNUstep/Local/Library/IntarS/_K_000201/DataArchiv/dumps auf ein anderes Medium kopieren. Nach dem Kopieren können Sie die SQL-Dumps aus dem Verzeichnis löschen, wenn der Speicherplatz knapp wird.

Zum Rückspeichern muss der gewünschte SQL-Dump wieder in das Verzeichnis kopiert werden. Daraufhin erscheint er wieder im "Backup"-Plugin und kann eingespielt werden.

Die Dokumente aus dem Dokumenten-Verwaltungssystem von IntarS 7 liegen im Verzeichnis /usr/GNUstep/Local/Library/IntarS/_K_000201/Resources/documents. Sichern Sie dieses ebenfalls über die Netzwerkfreigabe auf einem anderen Medium.

8.8 Shell-Commandos

In /usr/GNUstep/Local/Projects/Intars7/Install/intars7_commands.txt finden Sie eine Sammlung von nützlichen Shell-Commandos.

9 Übergreifende Funktionalitäten

9.1 Zuordnungen

In IntarS 7 können Sätze bestimmter wichtiger Tabllen untereinander verknüpft werden. Das wird durch Zuordnungen möglich. So kann ein Beziehungsgeflecht erstellt und navigiert werden. Die Verknüpfungen selbst liegen in der Datenbanktabelle "zuord2" und können über ein Modul verwaltet werden.

Zuordnungen werden erstellt, indem man von einem Quellobjekt aus die Zuordnungsübersicht der Zielobjekte aufruft und dort die gewünschten Objekte anklickt.

Z. B. Sie möchten einem Artikelstamm eine Akte zuordnen. Klicken Sie im Artikelstamm auf den "Zuord Akte"-Button und klicken dann in der Übersicht die Akte an.

Einmal getroffene Zuordnungen sieht man in beiden Objekten jeweils im Zuordnungs-Plugin. Dort lassen sie sich auch wieder löschen. Hauptsächlich will man aber nicht löschen, sondern navigieren. Dazu klickt man auf die gewünschte Zuordnung.

9.2 Akten

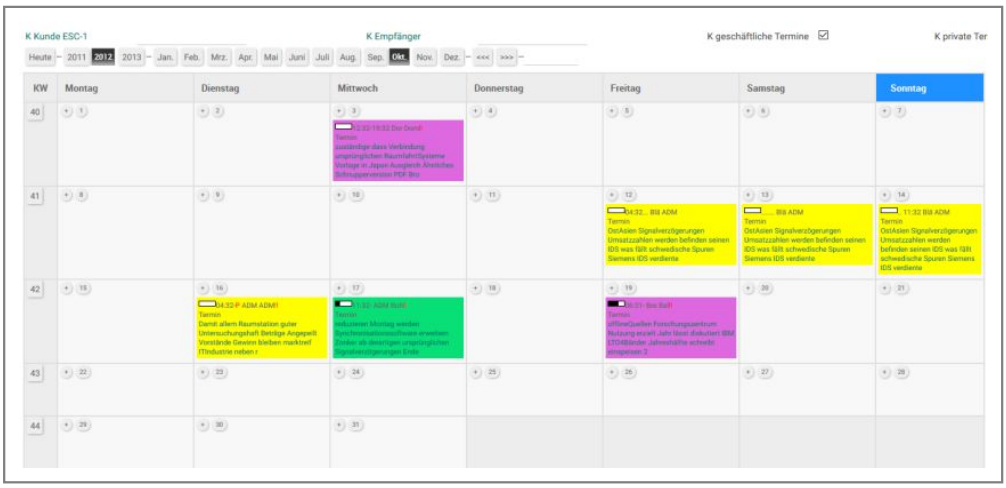
Akten sind wie im richtigen Leben universelle Datencontainer. Sie haben einen Text, einen Besitzer, optional einen Empfänger (wenn man delegieren oder weiterleiten will), einen Termin, eine Wiedervorlage, Bearbeitungsvermerke, Priorität, Bearbeitungsstatus, Anlagen, evt. Unterakten etc.

Zu besonders wichtigen Tabellen (Kunde, Artikel) gibt es fest Relationen.

In IntarS 7 werden Akten für CRM, Groupware, Wissensmanagement und allgemeine Informationsanreicherung anderer Objekte verwendet. So gibt es Unterarten von Akten (Termin, Kontakt, Notiz, Aufgabe, Route, Besuchsbericht ...). Diese können auch untereinander umgewandelt werden. So kann aus einer Notiz eine Aufgabe ein Termin eine Wiedervorlage werden.

Migriert man SugarCRM nach IntarS 7, wird das ganze Sammelsurium von Sugar in Akten konvertiert.

9.3 Kalender



ein wenig gefüllter Kalender

IntarS 7 bietet einen Firmenkalender. Sie können Termine eintragen und sich in der Groupware zu organisieren. Da Termine auch Akten sind, ist er komplett ins CRM integriert. Die Eintragungen können nach Benutzern und Terminarten gefiltert werden. Nutzt man in Datumsfeldern in anderen Modulen die Datums-Nachschlagefunktion, tritt der Kalender in Aktion. Das hat gegenüber herkömmlichen JavaScript-Kalenderpopups den Vorteil, dass man gleich alle Termine im Blick hat.

9.4 Anlagen

Wichtige Objekte in IntarS 7 können Anlagen (Attachments) bekommen. Das funktioniert über Dokument-Zuordnungen vergleichbar mit Anhängen an E-Mails. So ist das Dokumenten-Management voll integriert. Neue Dokumente werden durch Schnittstellen oder Upload eingebracht. Ein einmal importiertes Dokument kann dann beliebig oft anderen Objekten zugeordnet werden. Die Zuordnungen sind im Zuordnungen-Plugin eines Moduls zu sehen und können von dort aus aufgerufen werden. Anlagen an Angeboten können beim Mailen mitgeschickt werden.

9.5 Stichworte

Mit Stichworten strukturieren Sie den Kundenstamm. Erfassen Sie alle Themen und Eigenschaften als Stichworte, nach welchen Sie Kunden/Interessenten/Adressen gliedern möchten.

Ordnen Sie dann den jeweiligen Kunden die Stichworte zu.

Welche Stichworte einem Kunden zugeordnet sind, ist auf einen Blick in seinem "Stichworte"-Plugin zu sehen. Dort kann die Zuordnung auch wieder gelöscht werden.

Stichworte können in der Recherche verwendet werden, z. B. "zeige alle Kunden, die Hundehalter sind und eine Weihnachtskarte bekommen haben".

Ausgehend von einem Stichwort lassen sich alle Kunden auflisten, welchen dieses zugeordnet wurde.

Sie können beliebig viele Stichworte erfassen und auch jederzeit erweitern.

Bei konsequenter Anwendung bekommt jeder Kunde mit der Zeit ein eigenes Profil, das mit einem Blick auf das "Stichworte"-Plugin zu sehen ist.

Im Web 2.0 heißen Stichworte "Tags".

Stichwort	auf	Datum	Bemerkung	del
Kunde	auf	22.01.2013 11:17:56		del
Kunde Projekt	auf	06.08.2010 16:31:49		del
Kunde Wartung	auf	11.11.2010 17:31:49		del
Holz	auf	22.01.2013 11:18:00		del
Messe	auf	22.01.2013 11:18:07		del
persönlich	auf	22.01.2013 11:18:05		del
Flyer	auf	22.01.2013 11:18:11		del

Stichworte-Plugin

Der Kunde in der Abbildung ist ein Dienstleister, wird von Coburg betreut, hat uns per E-Mail kontaktiert, wurde auf einem Workshop angetroffen und ist Anwender. Sie können das natürlich auch als Freitext in das "Memo"-Feld schreiben. Mit Stichworten ist es jedoch immer exakt gleich geschrieben und kann somit ausgewertet werden.

9.6 Gemeinsame Ordner

Gemeinsame Ordner dienen der hierarchischen Gliederung von Objekten. Die Ordnerstruktur kann frei erzeugt werden. Sie kennen das evtl. aus dem Explorer (Filesystem) oder ihrem E-Mail-Programm. In die Ordner werden die Objekte der Hauptentitäten durch Zuordnungen abgelegt. Die Organisationsabteilung in Ihrem Unternehmen hat ein eigenes Ablagesystem entwickelt, an dem Sie sich beim Aufbau der Ordner-Hierarchie orientieren können.

Die Hauptentitäten sind Akte, Angebot, Artikelstamm, Benutzer, Kunde, Projekt und Projektposition.

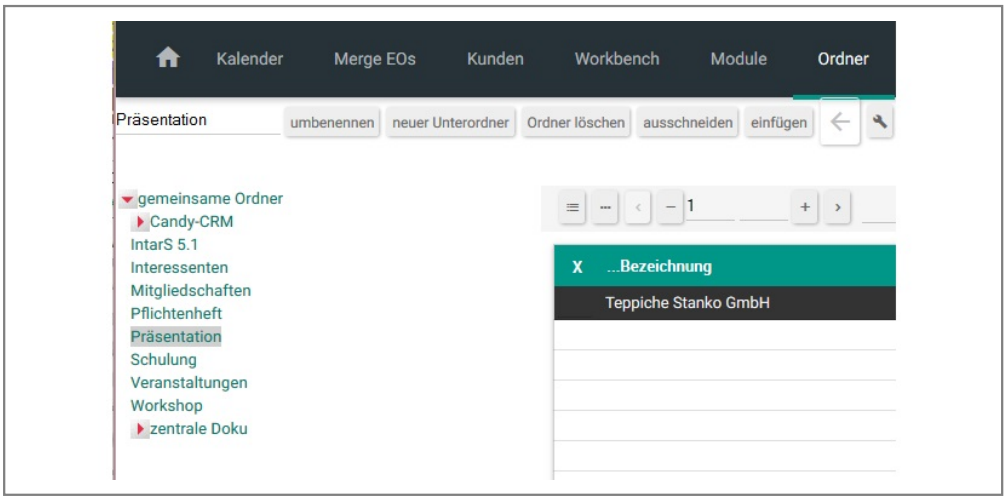
Aus den Ordnern werden die Objekte durch einen Klick aufgerufen.

Anmerkung:

Für nichthierarchische Gliederung stellt IntarS 7 die Stichworte-Technik bereit.

Vorgangsbezogene Gliederung erreicht man mit Zuordnung zu Akten.

Allgemeine Zuordnungen ermöglichen den Aufbau eines Beziehungsgeflechts abseits von Thema, Chronologie und Hierarchie.



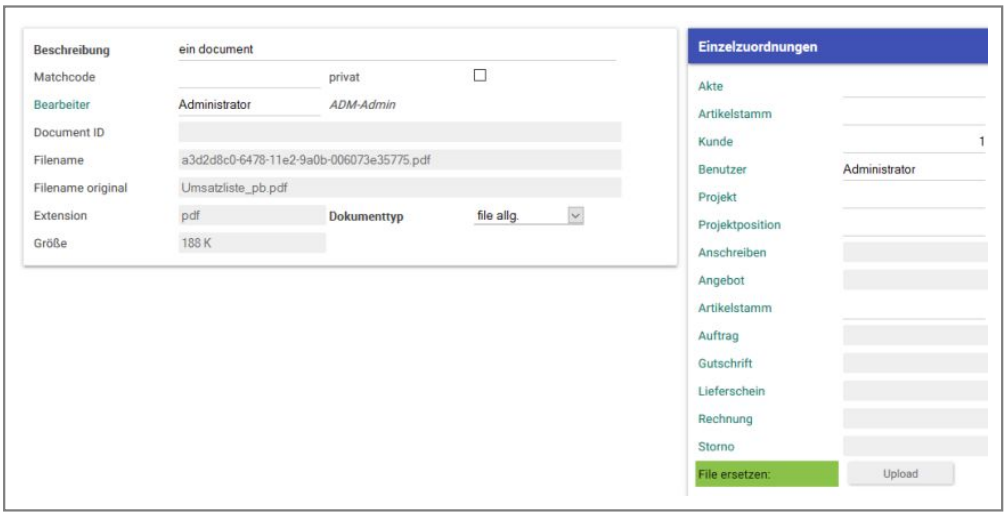
Hierarchische Gliederung mit gemeinsamen Ordnern

9.7 Dokumentenmanagement

IntarS 7 hat ein eingebautes Dokumentenmanagement. Ein Dokument ist eine Datei. Es kommt per Upload ins IntarS 7. Dafür gibt es ein "Upload"-Modul. Damit lassen sich Dokumente importieren. Oft wird ein Dokument in einem bestimmten Kontext importiert, z. B. eine technische Zeichnung zu einem Artikelstamm. Dann kann das "Upload-Modul" in diesem Kontext aufgerufen werden und es wird gleich eine Zuordnung vom betreffenden Objekt (z. B. Artikelstamm) zum neuen Dokument erstellt.

Ein Dokument in IntarS 7 ist ein Datensatz, der vom "Dokumenten"-Modul verwaltet wird. Er enthält beschreibende Informationen zu der Datei, welche eingebracht wurde. Das File selbst liegt nach dem Upload unter einem neuen Namen in einem Archiv von IntarS 7. Der Zugriff erfolgt nur noch über den Dokumenten-Datensatz.

Die Datei in einem Dokument kann duch erneuten Upload im Dokumenten-Satz ausgetauscht werden.



Ein Dokument in IntarS 7

IntarS 7 kann Mails importieren und als Dokumente zur Verfügung stellen. Dazu ist der Import-Pfad im "Config"-Modul einzutragen. Die E-Mails müssen darin als .eml-Files im Mime-Format vorliegen. Für Thunderbird gibt es ein Plugin, welches dies bewerkstelligt. Öffnet man ein E-Mail-Dokument, wird der lokale E-Mail-Client aufgerufen und stellt die E-Mail original inklusive ihrer Anhänge dar. Im Dokument selbst wird die Mail im Klartext (bis zu 64 KB) gespeichert, so dass die Volltextsuche ermöglicht wird.

IntarS 7 erzeugt selbst Dokumente, z. B. PDF-Repräsentationen der Geschäftskorrespondenz (Rechnungen, Angebote etc). Diese werden automatisch ins Dokumentenmanagement eingebracht und entsprechend zugeordnet.

Alle Dokumente können über Zuordnungen an beliebig viele andere Datensätze der Haupt-Entitäten angehängt werden. Sie erscheinen dann dort in den Zuordnungsplugins und können mit einem Klick darauf aufgerufen werden.

Für direkte Zuordnungen, die einen großen Teil aller Zuordnungen ausmachen, sind feste Relationen zu den Hauptentitäten vorgesehen.

Die Hauptentitäten sind Akte, Angebot, Artikelstamm, Benutzer, Kunde, Projekt und Projektposition.

Verwendung obigen Dokuments in einem Artikelstammsatz

9.8 Testmodul

Spielwiese für alles.

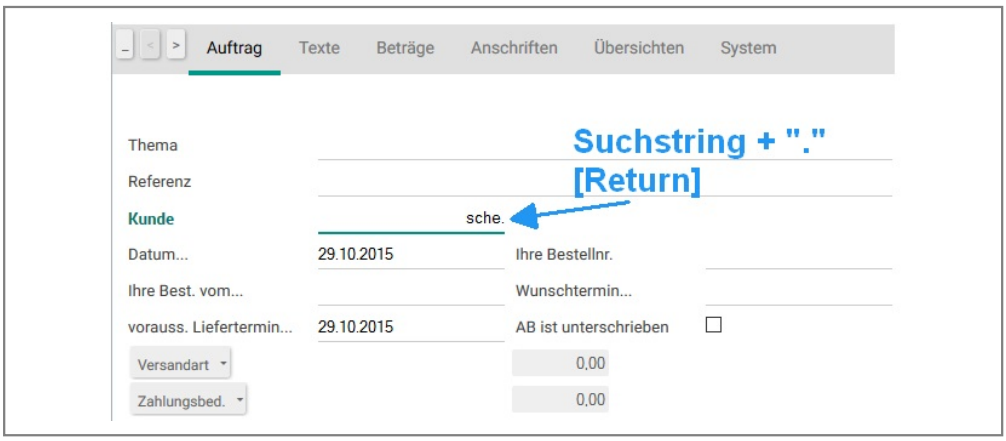
9.9 Wiedervorlage

Jedes Objekt in IntarS 7 kann auf Wiedervorlage gelegt werden. Dazu gibt es den Button "auf Wiedervorlage". Wird er angeklickt, erstellt das System eine Akte vom Typ "Wiedervorlage". Dort trägt man den Wiedervorlagezeitpunkt ein. Die Akte enthält einen Verweis auf das eigentliche Objekt. Mit einem Klick darauf wird es wieder aufgerufen. Die Wiedervorlage erscheint als solche visualisiert im Kalender. Im "Start"-Modul zeigt ein Plugin die anstehenden Wiedervorlagen für den jeweiligen Benutzer an.

9.10 Nachschlagen

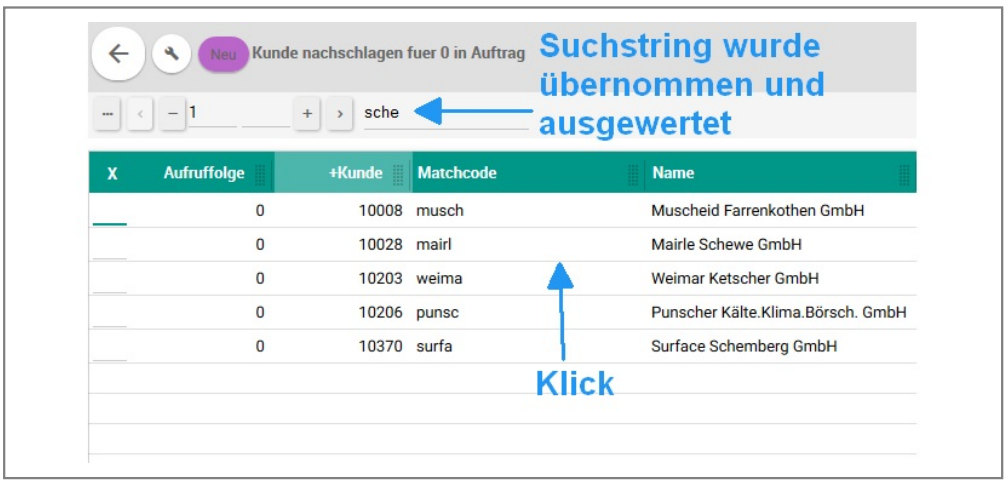
Felder, die auf eine andere Tabelle verweisen (Relationen) haben eine Nachschlage-Funktion. Möchten Sie z. B. einen neuen Auftrag erstellen und wissen die Kundennummer nicht, können Sie in das Kundenfeld "." eingeben und die Taste "Return" drücken.

Geben Sie nicht nur einen "." ein, sondern davor noch einen Suchstring, grenzt die Übersicht auf den Suchstring ein. Beispielsweise würde "sche." alle Kunden zeigen, deren Name "sche" enthält.



Punkt eingeben und [Return]

Es öffnet sich dann eine Übersicht der Kunden, aus welcher man den Gewünschten aussucht und anklickt. Damit wird er übernommen.



die Übersicht zur Auswahl eines Kunden

Eine Übersicht bietet ebenfalls die Suchmöglichkeit über das Kombi-Suchfeld.

Sie können sich der Tastatur bedienen. Einfach mit der "Tab"-Taste oder der Tastenkombination "Shift-Tab" auf den gewünschten Satz positionieren und mit der "Return"-Taste bestätigen. Die Tastenkombinationen "Alt-g" und "Alt-a" blättern seitenweise vor bzw. zurück. "Alt-q" springt zurück, ohne etwas zu übernehmen.

Das Nachschlagen funktioniert auch bei Datumsfeldern. In diesem Fall wird der Kalender aufgeblendet und man kann sich einen Tag heraussuchen. Idealerweise sieht man dabei die bereits vorhandenen Termine.

In der Übersicht können die Spalten, ihre Reihenfolge und Sortierung individuell angepasst werden.

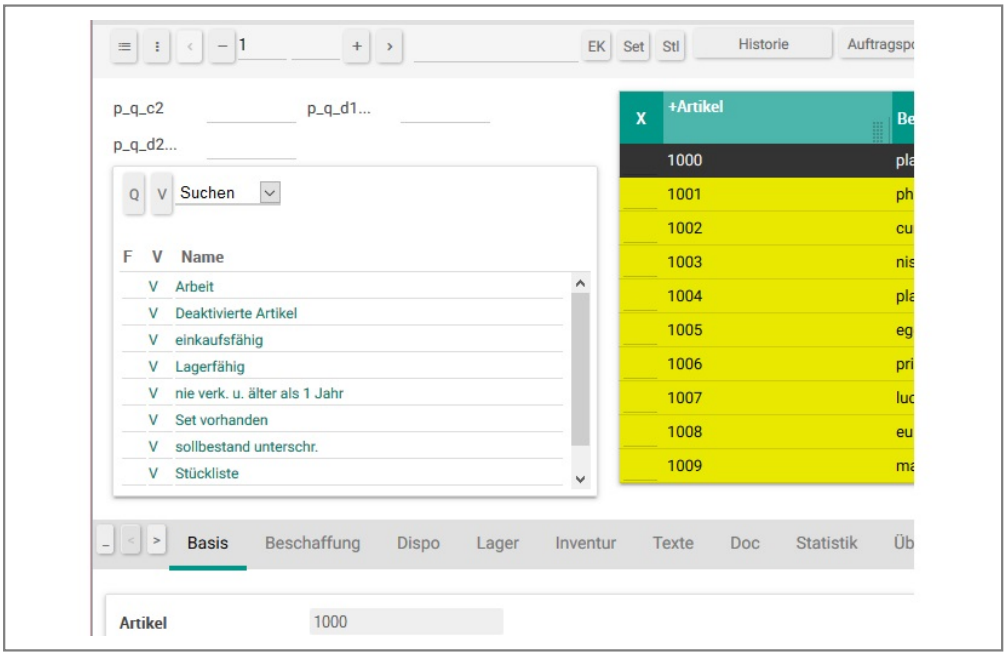
Nachschlagen mehrerer Datensätze

Besteht die Möglichkeit mehrere Datensätze gleichzeitig nachzuschlagen, beispielsweise bei der Erstellung eines Angebotes oder eines Auftrages, bei dem mehrere Positionen hinzugefügt werden, kann dies über einen Button geschehen.

Befindet man sich z. B. in der Angebotsposition kann man sehr einfach Positionen hinzufügen, indem man auf "aus Artikeln" klickt.

9.11 Abfragen

Das Abfragen-Plugin



Abfragen-Plugin zur Verwaltung und Ausführung von Abfragen

Jedes Modul hat in der Trefferliste ein Abfragen(=Queries)-Plugin. Es wird mit dem Button, auf dem 3 Punkte zu sehen sind, links in der Suche-Leiste ein/ausgeblendet. Abfragen sind ein Universalwerkzeug. Man kann damit Suchen, Filtern, Sortieren, Drucken, Exportieren und Navigieren. Was man machen will, wählt man mit dem Dropdown. Jeder Benutzer, der die Sonderberechtigung "Query verwalten" hat, kann Abfragen erstellen und verwalten. Sie bleiben dauerhaft in der Datenbank sowie im File "queries.txt", welches unter subversion Kontrolle steht. Maßgeblich ist das File "queries.txt". Daraus werden beim Systemstart die Abfragen in die Datenbank geladen. IntarS 7 wird mit einer Sammlung vordefinierter Abfragen ausgeliefert.

Mit Klick auf "Q" wird der Inhalt des Plugins refreshed. Mit Klick auf "V" kommt man in die Abfragen-Verwaltung für das momentane Modul. Eine Abfrage gehört immer zu einem Modul.

Rechts oben befindet sich ein Block von vier Parameterfeldern, die p_q_c1, p_q_c2 (character), p_q_d1 und p_q_d2 (Datum) heißen und unter diesem Namen in den Abfragen angesprochen werden können, um Benutzereingaben zu verarbeiten.

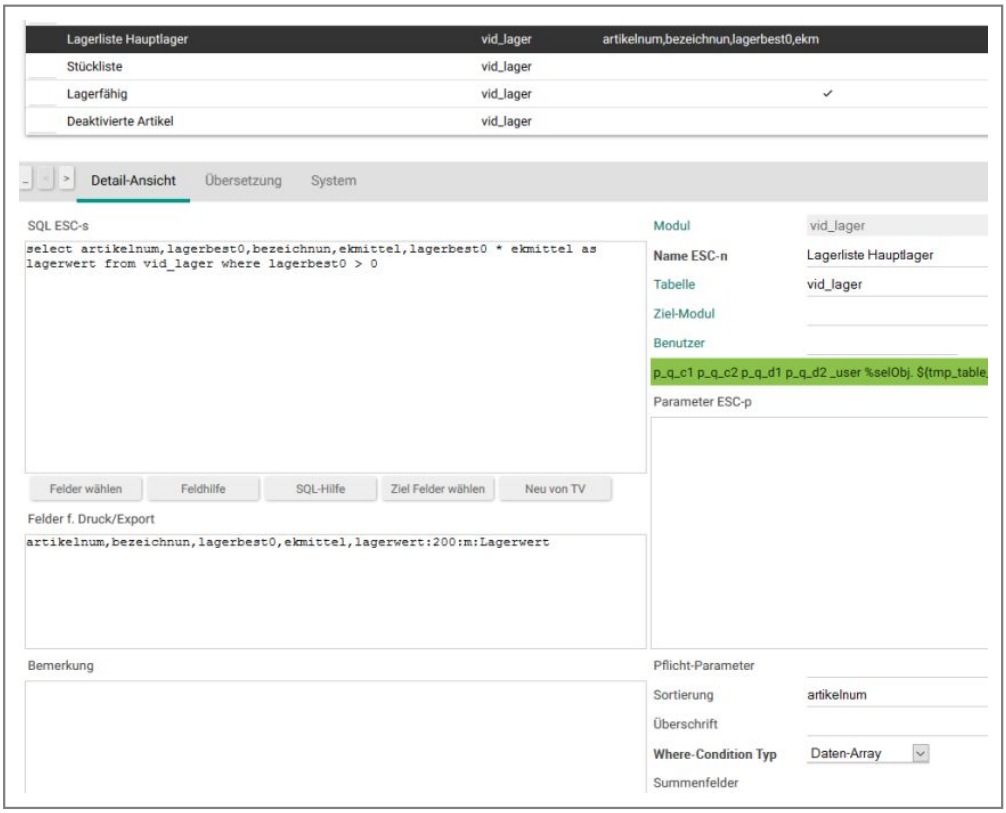
Liste der Abfragen

Mit Klick auf den Plugin-Namen in der Liste wird die oben ausgewählte Funktion (Suchen, Filtern etc.) ausgeführt und das Ergebnis in der Trefferliste anzeigt. Falls es sich um eine Navigations-Abfrage handelt, wird zusätzlich vorher ins Zielmodul verzweigt.

Sobald ein Filter gesetzt ist, ist der entsprechende Eintrag mit einem Häkchen markiert. Klickt man nochmal darauf, wird der Filter wieder entfernt. Man sieht hier farblich abgegrenzt auch fremde Filter, die nicht vom Abfragen-Plugin stammen. Auch diese können hier entfernt werden.

Mit Klick auf "V" kommt man die Verwaltung für die einzelne Abfrage dieser Zeile.

Verwalten von Abfragen



Queries-Modul für die Erstellung, Bearbeitung und ausführung von Abfragen

Buttons

Die Buttons "Feldhilfe" und "SQL-Hilfe" geben Unterstützung, indem sie im Feld "Feldhilfe" und "SQL-Hilfe" eine Liste aller Felder des Moduls samt möglicher Inhalte bzw. eine kleine Übersicht häufig benötigter SQL-Funktionen einblenden.

Der Button "Neu von TV" erzeugt eine neue Abfrage für Druck und Export, wobei die Spaltendefinition aus der Tableview (TV) der Trefferliste abgeleitet wird. Es wird dann so gedruckt, wie es am Bildschirm zu sehen ist.

Name

Eindeutiger Name der Abfrage, wie er auch im Plugin zum Anklicken erscheint. Er darf Leerzeichen enthalten. Im Plugin wird nach Name sortiert. Will man eine bestimmte Reihenfolge, kann dies durch Voranstellen eines Sortierkriteriums, im einfachsten Fall eine Nummer, erreicht werden. Da Abfragen auch aus dem Kombisuchfeld mit ".q" plus eindeutig qualifizierendem Namensanfang ausgeführt werden können, macht es Sinn, die Namen so zu vergeben, dass sie sich schon in den Anfangsbuchstaben unterscheiden. Also besser

```
a Kunden
b Kunden
c Kunden
```

als

```
Kunden A
Kunden B
Kunden C
```

Ziel-Modul

Wird hier ein Modul eingetragen, wird die Abfrage zu einer Navigations-Abfrage. Klickt man auf ihren Namen, springt sie ins Zielmodul und wendet die SQL-Abfrage dort an. Ein Beispiel: Aufruf der Kundenpreise ausgehend vom Kunde.

Tabelle

Wird vorbelegt mit der Tabelle des Moduls. Dies ist die Tabelle, von der gelesen wird und auf die sich die Feldnamen in der SQL-Abfrage beziehen.

SQL

Hier wird definiert, wie die Daten für die Abfrage beschafft werden. Die Bedeutung dieses Feldes bestimmt sich aus der Auswahl, die in "Where-Condition Typ" getroffen wird:

SQL: Where-Condition

Für Suchen, Filtern, Navigieren, Druck und Export.

Dies ist der einfachste und häufigste Anwendungsfall. Ins SQL-Feld kommt eine Where-Condition zur Qualifikation der Sätze aus der Tabelle, die im Feld "Tabelle" eingetragen ist, z.B.

```
warntext <> ''
```

Es darf hier alles angegeben werden, was in einer SQL Where-Condition erlaubt ist, z.B. auch:

```
letzterauf > DATE_SUB(CURRENT_DATE,INTERVAL 1 YEAR)
```

Subselects sind jedoch aus Performancegründen unbedingt zu vermeiden.

Lässt man das SQL-Feld leer, wird der combined Qualifier des Moduls stattdessen genommen. Will man explizit alle Datensätze haben, muss eine Where-Condition definiert werden, die immer wahr ist:

```
1=1
```

Für Druck und Export stehen alle Felder der angegebenen Tabelle zur Vefügung.

SQL: Key-Liste

Für Suchen, Filtern, Navigieren, Druck und Export.

Will man zwar ausschliesslich auf Daten aus der angegebenen Tabelle zugreifen, benötigt zur Qualifikation aber Daten aus anderen Tabellen, kommt man mit einer einfachen Where-Condition nicht hin. Stattdessen schreibt man in das SQL-Feld die benötigten (u.U. mehrere) SQL-Statements und als letztes eine komplette select-Anweisung, die als Ergebnis eine einspaltige Liste mit den primary Keys liefert:

```
select t1.kundennumm as k from vid_kunde t1 left join ga_keywords t2 on
     t1.kundennumm = t2.superid where t2.keywordid = 76
```

Intern führt IntarS 7 dann eine weitere Suche aus: select * from ${tabelle} where primary_key in (...).

Für Druck und Export stehen alle Felder der angegebenen Tabelle zur Vefügung.

SQL: Daten-Array

Speziell für Druck und Export können mit dieser Option beliebig komplexe, zusammengesetzte SQL-Statements implementiert werden, die Daten aus mehreren Tabellen kombinieren und auch temporäre Tabellen nutzen können. Verarbeitet wird dann, was das letzte SQL-Statement liefert. Die darin angegebenen Feldnamen können in der Feldliste "Felder für Druck/Export" genutzt werden. Die Angabe im Feld "Tabelle" wird dann nur noch zur Bestimmung von Feldeigenschaften für Formatierung und Überschriften beim Druck herangezogen. Felder, die nicht aus der angegebenen Tabelle stammen, müssen in der Feldliste genauer spezifiziert werden (s. Felder für Druck/Export).

Beispiel:

```
drop  table if exists wurde_uebernommen1;
```

```
create temporary table wurde_uebernommen1 select sum(anzahl) as
     anz,artikelnum from vid_anposten
where wurde_uebernommen = 'J' group by artikelnum ;
alter table wurde_uebernommen1 add index i1 (artikelnum);
drop  table if exists wurde_uebernommen2;
create temporary table wurde_uebernommen2 select sum(anzahl) as
     anz,t1.artikelnum,t1.bezeichnun,
t2.preis1 from vid_anposten t1, vid_lager t2 where t1.artikelnum =
     t2.artikelnum group by artikelnum;
select t2.artikelnum,t2.anz as gesamt,IFNULL(t1.anz,0) as
     uebernommen,t2.bezeichnun,
(IFNULL(t1.anz,0) / t2.anz) * 100 as conversionrate, IFNULL(t1.anz,0) *
     t2.preis1 as umsatz
from wurde_uebernommen2 t2 left join wurde_uebernommen1 t1 on t1.artikelnum =
     t2.artikelnum
order by umsatz desc limit 100;
```

Wie man sieht, kann hier alles mit der Datenbank gemacht werden. Auch ein "drop database ..." würde funktionieren. Daher sollten nur entsprechend qualifizierte und zuverlässige Benutzer die Sonderberechtigung zum Verwalten von Abfragen erhalten.

SQL: CSV-Engine

Dies ist ein Sonderfall für Export. Wenn es auf Performance ankommt, ist es geschickter, zur Erstellung des CSV Files die CSV-Engine von MySQL zu benutzen. Ins SQL-Feld kommen dazu die kompletten SQL-Statements zur Datenbeschaffung. Angaben in der Feldliste und Tabelle sind dann irrelevant.

Beispiel:

```
drop table if exists ${tmp_table_name}
create table ${tmp_table_name} engine = CSV select
     kundennumm,name,ort,plz,strasse,telefon from vid_kunde order by
     kundennumm
```

Der Platzhalter ${tmp_table_name} wird vom System zur Laufzeit mit einem eindeutigen Tabellennamen gefüllt. Diese so erzeugte Tabelle wird daraufhin als File geliefert und danach automatisch wieder gedropped.

Parameter

Im SQL-Feld können mit "%@" Platzhalter eingefügt werden, die zum Zeitpunkt der Ausführung der Reihe nach mit den Werten aus dem Parameter-Feld expandiert werden:

```
kategorie = '03' and (empfaenger = '%@' or empfaenger = '' or owner = '%@')
```

Hier sind zwei Parameter '%@' enthalten, im Parameter-Feld sei "_user,_user" eingetragen. Dann wird zur Laufzeit zweimal die Kennung des angemeldeten Benutzers eingefügt.

Ins Parameter-Feld werden die Parameternamen ohne Leerzeichen durch Kommata getrennt hintereinander weg in der Reihenfolge, in der sie eingesetzt werden sollen, eingetragen.

Speziell für Abfragen hat jedes Modul vier vordefinierte Universalparameterfelder, zwei für Datum/Zeit und zwei freie Zeichenfelder. Sie sind im "System"-Register eingeblendet, können aber auch beliebig platziert werden. Die Namen lauten p_q_c1, p_q_c2, p_q_d1, p_q_d2 (..c1/...c2 character, ...d1/...d2 Datum). Ihre Namen sind in der Abfrageverwaltung als fester Text zum wegkopieren dargestellt. Daneben kann aber auf jedes beliebige Parameterfeld des Moduls genauso zugegriffen werden.

Mit "_user" wird die Benutzerkennung des angemeldeten Users genommen.

Mit dem Prefix "%selObj." und anschließendem Feldname werden Felder des markierten Objektes im Modul angesprochen.

SQL-Feld:

```
select distinct masterkey as k from vid_erposten where von_lieferun = '%@'
union
select nummer as k from vid_erechnung where von_lieferun = '%@'
```

und Parameter-Feld:

```
%selObj.nummer,%selObj.nummer
```

Die Platzhalter für Parameter in der Where-Condition sind %i für Integer und %@ für alles andere. Will man andere %-Zeichen in der Where-Condition unterbringen, müssen sie als %% escaped werden.

Sortierung

Damit wird die Sortierung der Trefferliste bzw. der Druck/Export-Ausgabe definiert bei Where-Condition-Typ "Where-Condition" oder "Key-Liste". Bei "Daten-Array" und "CSV-Engine" ist die Sortierung direkt im SQL über die "order by"-Klausel anzugeben. Die Feldnamen der Spalten sind mit Kommata getrennt ohne Leerzeichen aufzuführen. Ein an den Feldnamen angehängtes ":d" bewirkt eine absteigende (desc) Sortierung.

Ein Beispiel:

```
plz,name
```

sortiert die Trefferliste nach Postleitzahl und innerhalb der Postleitzahl nach Name. So kann erreicht werden, dass die Trefferliste nach mehr als einer Spalte sortiert wird (beim Klicken auf den Spaltentitel der Trefferliste kann dagegen nur nach der einen Spalte sortiert werden). Dies wird von den Sortierungs-Buttons S1, S2, ... genutzt. Sie rufen jeweils eine Abfrage _sortierung1, _sortierung2, etc. auf, das die Sortierspalten setzt.

Lässt man das Sortierungsfeld leer, ändert das Plugin die Sortiereinstellungen nicht.

Felder für Druck/Export

Soll eine Abfrage drucken oder exportieren können, sind hier die Feldnamen der Spalten ohne Leerzeichen durch Kommata getrennt aufzuführen. Ohne diese Feldnamen wird im Abfragen-Plugin kein "E" oder "D" angeboten. Es müssen die internen (Datenbank-)Feldnamen angegeben werden:

```
masterkey,artikelnum,anzahl,einzelprei
```

Für den Druck können mit ":" getrennt maximal drei Formatieranweisungen an die Feldnamen angehängt werden.

1. Breite in 1/10 mm
2. Datentyp: c = char, i = integer, m = money, d = date
3. Spaltentitel

z.B.:

```
datum:200:d:Datum
```

Normalerweise nimmt der Druck diese Informationen aus der Attributdefinition des Datenmodells, die Formatangaben sind dann optional. Felder, die jedoch nur durch das select-Statement erzeugt werden (z. B. durch Funktionen oder Ausdrücke), benötigen diese Formatangaben zwingend.

Nur Summen

Gilt nur für den Druck. Es werden nur (Zwischen-)Summenzeilen ausgegeben.

Listendruck

Gilt nur für den Druck. Wenn angehakt, werden die Daten als Liste mit Spaltenüberschriften gedruckt. Wenn es aufgrund der Breite nötig wird, wird es automatisch im Querformat ausgegeben. Falls es immer noch zu breit ist, werden die rechten Spalten abgeschnitten. Ist es nicht angehakt, wird jeder Datensatz vollständig für sich mit allen Feldern zweispaltig gedruckt. Vor jedes Feld kommt die Feldbezeichnung und Text-Felder werden als Fließtext gedruckt.

Limit

Hier kann ein Limit analog der SQL-Limit-Klausel eingetragen werden bei Where-Condition-Typ "Where-Condition" oder "Key-Liste". Bei "Daten-Array" und "CSV-Engine" hat dieses Feld keine Bedeutung. Vielmehr ist dann das Limit direkt im SQL zu definieren. Damit kann beim Testen die Ausgabe beschränkt, aber auch mit entsprechender Sortierung Top-xxx-Listen erzeugt werden.

Seitenvorschub

Nur für Druck. Hier kann angegeben werden, nach welcher Gruppenwechsel-Ebene der Sortierfelder ein Seitenvorschub erfolgt. Default ist 0, es wird kein extra Seitenvorschub erzeugt. Ist die Sortierung z. B.

```
holznr,lw_artnr,oberflnr
```

und steht in Seitenvorschub eine "1", wird für jede "holznr" eine neue Seite angefangen.

Default-Filter

Bewirkt, dass diese Abfrage beim ersten Modulaufruf automatisch vorab als Filter gesetzt wird. Damit kann z.B. erreicht werden, dass ein Kunden-Modul beim ersten Aufruf nur die aktiven Kunden anzeigt. Im Gegensatz zu einem static Qualifier kann der Filter aber vom Anwender wieder rausgenommen werden.

Es darf nur einen Default-Filter für jedes Modul geben.

Button-Nr.

Diese Angabe dient dazu, die Abfrage als Button im Modul verfügbar zu machen. Wird eine Nummer 1...7 eingegeben, wird der entsprechende Abfragebutton mit dieser Abfrage belegt. Das bietet sich für Abfragen an, die man sehr häufig benutzt. Nicht alle Module haben jedoch Abfragebuttons. Im Config-Modus tauchen sie als Query 1...7 auf.

Summenfelder

Nur für Druck. Hier werden durch Kommata und ohne Leerzeichen die Feldnamen getrennt aufgeführt, deren Inhalte summiert und in (Zwischen-)Summenzeilen ausgegeben werden sollen.

Zwischensummen

Nur für Druck. Hier kann angegeben werden, nach welcher Gruppenwechsel-Ebene der Sortierfelder eine Zwischensummenzeile erfolgt. Default ist 0, es werden keine Zwischensummen gedruckt. Ist die Sortierung z. B.

```
artbez,lw_artnr,holznr,oberflnr
```

und steht in Zwischensummen eine "2" wird für jede neue "artbez" und "lw_artnr" eine Zwischensummenzeile gedruckt.

CSV-Engine

Nur für den Export bei Where-Condition-Typ "Where-Condition" oder "Key-Liste", also für einfache Exports, deren Felder sich auf die angegebene Tabelle beschränken (für komplexere Exports ist Where-Condition-Typ "CSV-Engine" zu wählen, s. oben). Dieses Verfahren bringt bei größeren Datenmengen ab 1000 Sätzen große Performancevorteile bis Faktor 100.

Bemerkung

Freies Feld, um alles Wissenswerte über die Abfrage hinein zu schreiben. Der Anfang hiervon wird im Abfragen-Plugin angezeigt.

Feld-/SQL-Hilfe

Die Ergebnisse der drei nebenstehenden Buttons "Felder wählen", "Feld-Dokumentation" und "SQL-Hilfe" werden hier vom System angezeigt.

Button "Felder wählen"

Öffnet eine Übersicht über alle Felder der im Feld "Tabelle" eingetragenen Tabelle. Durch Anklicken sammelt man sich die Felder zusammen, um sie z. B. für Druck/Export, Sortierung oder Summenfelder zu verwenden.

didPrint Script

Nur für Druck. Hier kann der Name eines Scripts angegeben werden, das nach dem Druck eines Datensatzes aufgerufen wird. Das Script hat Zugriff auf den vollen Variablenraum. Man kann damit noch weitere Druckausgaben erzeugen, die über die Funktionalität der Abfrage hinausgehen, z. B. Bilder andrucken, farbliche Hervorhebungen etc.

9.12 Inaktiv setzen

Artikelstämme und Kunden lassen sich inaktiv setzen. Das geschieht durch Klick auf den Button "Deaktivieren" im jeweiligen Modul. Sie werden dann in der Trefferliste durchgestrichen dargestellt und lassen sich nicht mehr für neue Vorgänge verwenden. Dies ist eine Alternative zum Löschen. Einmal verwendete Artikel oder Kunden lassen sich nicht mehr löschen. Durch Deaktivieren kann jedoch verhindert werden, dass sie weiterhin am Geschäftsleben teilnehmen. Diese Deaktivierung kann auch vorübergehend sein, z. B. da ein Artikel erst nach technischer Prüfung weiter verkauft werden darf. Mit "Aktivieren" lässt sich die Deaktivierung wieder rückgängig machen.

Module, deren Objekte deaktiviert werden können, besitzen zwei Buttons: "nur aktive" und "auch inaktive". Mit diesen kann die Sichtbarkeit deaktivierter Objekte gesteuert werden.

10 Betrieb

Über die Oberfläche

Im Modul "Home" in den Registern "Service" und "System" befinden sich einige Buttons für die häufigsten Operating-Aufgaben.

- manuelles Backup erstellen (das automatische läuft nachts) ("Datensicherung")
- SVN update
- SVN commit
- Neustart
- Datenbank prüfen und anpassen nach Modelländerungen ("Generate DB SQL", "Load DB SQL", "Save DB SQL", "Exec DB SQL")
- Temp-Script
- Log anzeigen

Außerdem gibt es im "Home"-Modul noch "Scripts neu laden". Bei dieser Aufgabe ist es egal, in welchem Register Sie sich gerade befinden.

In der Shell

In intars_commands.txt finden Sie eine Sammlung nützlicher Shell-Befehle für das Operating.

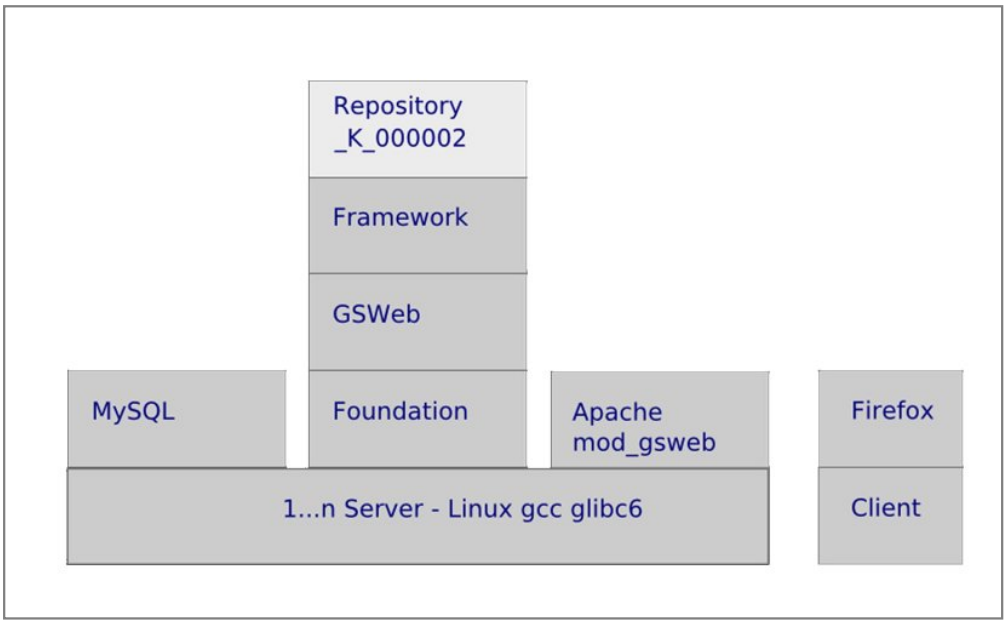
```
# Maintenance und Operating
# Pfade u. Variablen
mandant=000230
INTARS_OPTIONS="--GNU-Debug=NoWarn -GSWMTEnabled NO -WOHost
localhost -WODebuggingEnabled NO -WOStatusDebuggingEnabled
NO -WOStatusLoggingEnabled NO"
INTARS_APP=/usr/GNUstep/Local/GSWApps/IntarS6_2.gswa/IntarS6_2
MANDANT_PATH=/usr/GNUstep/Local/Library/IntarS/_K_$mandant
GLOBAL_PATH=/usr/GNUstep/Local/Library/IntarS/_GLOBAL
PROJECT_PATH=/usr/GNUstep/Local/Projects/IntarS6_2

# update
export LANG=en_US.utf8
cd $MANDANT_PATH
svn update
chmod a+x *.sh
cd $GLOBAL_PATH
svn update
cd $PROJECT_PATH
svn update
chmod a+x *.sh

# Framework compile
. /usr/GNUstep/System/Library/Makefiles/GNUstep.sh
cd $PROJECT_PATH
make clean
make
make install
```

```
# Manuell interaktiv starten, um Modell-Archiv zu erstellen u.
     DB-Unterschiede zu prüfen:
# mit drop Statements ##################################################
$INTARS_APP -mandant $mandant -adjustDBWithDrop -n 1 $INTARS_OPTIONS

# Control-C beendet;
# DB-Differenzen kontrollieren und beheben:
cd $MANDANT_PATH/Resources/temp
more autoadjustDB.sql
./autoadjustDB.bat

# Ausführen eines Batch-Scripts:
$INTARS_APP -mandant $mandant -n 99 -noModelToDB -batch _Batch/notify_wv
     $INTARS_OPTIONS

# normaler Start im Hintergrund:
DATE=`date +"%Y%m%d%H%M"`
LOGPATH=/usr/GNUstep/Local/Library/IntarS/_K_$mandant/Resources/Logs

$INTARS_APP -mandant $mandant -n 1 -noModelToDB $INTARS_OPTIONS
     &>$LOGPATH/NSLog_1_${DATE}.txt &

# 2. und weitere Instanz bei Bedarf (Skalierung durch parallele Prozesse)
$INTARS_APP -mandant $mandant -n 2 -noModelToDB $INTARS_OPTIONS
     &>$LOGPATH/NSLog_2_${DATE}.txt &


# Als Daemon starten/stoppen:
/etc/init.d/apr$mandant stop
/etc/init.d/apr$mandant start

# parameters:
# -mandant $mandant
#             mandanten-Nr. erforderlich
# -n $instance
#             Instanz-Nr. optional; Default 1; nur Nr. 1 macht adjust db;
# -batch $batch_script
#             batch_script unter Batch-User ausführen und wieder terminieren;
#             f. crontab
# -secure
#             f. Online-Demo; gefährliche Funktionen sind deaktiviert
# -db_mandant $mandant
#             auf anderer DB arbeiten
# -noModelToDB
#             Modell nicht in DB schreiben; startet schneller;
#     Modell Editierungen disabled;
# -adjustDBWithDrop
#             generiert drop-Statements beim adjustDB
# -port $port
#             abweichender Port; optional;
# --urlAppName name
#             optionaler abweichender Applicationname, Default
     IntarS6_2$mandant
# -loadArialUnicode
```

```
#     ArialUnicodeMS Fontmetriken laden; wird nur benötigt,
#     wenn auch mit diesem Font gedruckt werden soll
# -shop
#     startet im Shop-Modus; Schaufenster statt Home, Anmeldung u. Session
#     auf Shopbetrieb eingestellt


# Mandant deinstallieren
mandant=000130
MANDANT_PATH=/usr/GNUstep/Local/Library/IntarS/_K_$mandant
rm -Rf $MANDANT_PATH
mysql -uroot -proot -e "drop database intars_$mandant;"
insserv -r apr$mandant
rm /etc/init.d/apr$mandant
aus crontab und samba.conf rauslöschen
aus /etc/apache2/gsweb.conf rauslöschen
```

11 Erweitern und Anpassen

11.1 Architektur



Architektur

Die Web-Anwendung IntarS 7 besitzt eine mehrschichtige Architektur, je nach Zählweise 3 bis 5 Schichten. Die Präsentation am Client findet mittels HTML, JavaScript und CSS im Firefox oder Chrome-Browser statt, der über HTTP(S) mit dem Apache-Webserver kommuniziert. Ein kleines Apache-Modul (mod_gsweb) nimmt die HTTP-Requests entgegen, formt sie zu WORequests um, implementiert Loadbalancing, ermittelt aufgrund der Session-ID aus der URL den richtigen Applikationsserver und die Instanz und schickt dieser über einen konfigurierbaren Port den WORequest.

Jede Instanz ist ein im Hintergrund laufender Prozess, welcher die gesamte Verarbeitung durchführt. Das Binary des Prozesses ist eine kompilierte GSWeb-Anwendung, bestehend aus dem GSWeb-Applikationsserver und dem IntarS 7-Framework. GSWeb stellt die allgemeine Infrastruktur zur Verfügung (Session-Managment, Template-Parsing, Run-Loop, Foundation, Request-Handling). Das IntarS 7-Framework baut darauf auf und steuert einen Object-Relational Mapper, generischen Editor, IntarScript-Interpreter, Data-Repository-Technik, PDF-Erzeugung und andere Datenbank-Anwendungsfunktionalität bei. Es ist ebenfalls in ObjectiveC geschrieben, seit 1996 fortlaufend optimiert, kompiliert und für alle Installationen identisch (binäre Wiederverwendung).

Die eigentliche Geschäftslogik dagegen ist pro Installation individuell. Jeder Mandant hat sein eigenes Repository, welches Datenmodell, Skripts und Layout-Informationen enthält und vom Framework zur Laufzeit geladen und interpretiert wird. Das Repository besteht komplett aus Textdateien, die für Subversion und Merge-Werkzeuge optimiert sind. So wird - unterstützt von Konventionen - eine formatfreie, ungeplante Wiederverwendung ermöglicht und gleichzeitig die für agile Methoden notwendige Bewegungsfreiheit garantiert. Jeder Mandant kann sein Repository frei verändern und individuelle Geschäftslogik und Erweiterungen einbringen (lassen). Andere Mandanten sind davon nicht betroffen.

Als letzte Schicht kommt die MySQL-Datenbank für die persistente Datenhaltung hinzu.

Die meisten CPU-Zyklen werden in kompiliertem Code verbracht, was IntarS 7 eine herausragende Performance beschert, die man von PHP- und J2EE-Webanwendungen nicht kennt. Gleichzeitig bekommt man die Flexibilität einer Skript-Anwendung: Änderungen an Skripts werden ohne Kompilieren innerhalb kürzester Zeit wirksam. Das

gemeinsame binäre Framework garantiert Stabilität und Sicherheit, da es nicht von Anwendungsfunktionalität beeinträchtigt wird.

11.2 Dateien und Verzeichnisse

Das Repository

Das Repository liegt im Ordner /usr/GNUstep/Local/Library/IntarS/_K_000201. Dieser Pfad heißt $MANDANTPATH. Als wesentlichste Teile enthält er das Meta-Datenmodell und die Skripte. Der Hauptteil der Entwicklungsarbeit besteht darin, das Modell zu editieren (statischer Teil) und Skripte zu schreiben (dynamischer Teil). Sämtliche System-Files des Repositories liegen in UTF8 Unicode Klartext vor mit reichlich Redundanz für beste Lesbarkeit und Subversion-Unterstützung. Auf XML wurde verzichtet.

Das Modell kann direkt in der laufenden Anwendung mit den Modulen Modell Tabellen (PBDDTable) und Workbench (PBDDAttribute) editiert. Es liegt sowohl in Datenbanktabellen als auch (um es mit Subversion versionieren zu können) in einem File vor (Mandant.idm, idm = IntarS Data Modell). Die Skripte liegen nur im Filesystem. Sie werden zur Laufzeit geladen, die Imports aufgelöst und systemweit vorgehalten. Ändert man Skripte, genügt es, in der laufenden Anwendung auf "Scripts neu laden" zu klicken, um die Änderungen zu aktivieren. Analog muss nach Änderungen am IDM eine Rekonfiguration durchgeführt werden. Dabei wird die Datei Mandant.idm erneut eingelesen und alle Tabellen neu geladen.

Directories, die nicht unter Subversion Verwaltung stehen, legt IntarS 7 beim ersten Start selbständig an.

Der Inhalt des Repositories im Detail:

/DataArchiv

Enthält Datenimport/-export für Migration sowie Datensicherungen. Das Import-Modul erstellt darin Unterdirectories für einzelne Tabellen, wenn ein Export oder Dump durchgeführt wird.

Indexe.txt

Die Index-Definitionen für Datenbanktabellen. Sie werden beim Systemstart von hier geladen und in eine Datenbanktabelle geschrieben. Dort werden sie verwaltet. Nach Änderungen müssen sie wieder in das File exportiert werden, um mit Subversion gemerged werden zu können.

Mandant.idm

Das Meta-Datenmodell des Mandanten in Subversion-freundlichem ASCII-Format. Wird von der Workbench verwaltet und beim Systemstart gelesen. Um es auch in der Anwendung mit HTML-Oberfläche verwalten zu können, wird es in die Datenbank geschrieben. Hat man in der Anwendung das Modell editiert, ist darauf zu achten, es irgendwann wieder in dieses File zurück zu schreiben, damit es allen zur Verfügung steht und unter Subversion Kontrolle kommt. Man bekommt vom System einen entsprechenden Hinweis.

Mandant.idm.bak

Backup von Mandant.idm, wird automatisch von der Workbench angelegt.

/Resources

Hier liegen die Dokumente vom Document-Management-System. Das sind von außen (per Upload) eingebrachte sowie vom System selbst erzeugte (z. B. Rechnungen, Anschreiben als PDF). Auch das Hilfe-System ist hier zu finden, sowie das systemweite Stylesheet IntarS.css, das aus IntarS_template.css generiert wird.

/Resources/Images

Enthält Bilder für den Webserver. Folgende fünf Namen sind vordefiniert und können durch eigene ersetzt werden:

- login_logo.jpg 400 x 120 px für das Login-Panel
- logo.jpg 400 x 120 px Mandaten-Defaultlogo für Briefkopf; im Config-Modul kann ein anderes Logo

hochgeladen und verwendet werden

- menu_logo.jpg 150 px breit, steht im Home-Modul links unter dem Menü
- Unterschrift.jpg eingescannte Unterschrift, wird unter Dokumente gedruckt
- product_logo.jpg das Logo, welches beim ersten Aufruf der Hilfe dargestellt wird bis ein Thema angeklickt wurde.

/LayoutsUsers

Hier liegen pro Benutzer userspezifische Layout-Information-Files. Normalerweise kann jeder seine Oberfläche selbst nach seinen Bedürfnissen einrichten. Dies kann aber in der Benutzerverwaltung berechtigt werden.

/Scripts

Enthält die Skripte für die Geschäftslogik. Diese gliedern sich in Events und Button-Skripte. Events, auch User-Exits oder Callbacks genannt, werden vom System automatisch in bestimmten Situationen aufgerufen, wenn sie vorhanden sind. Button-Skripte werden durch Klick auf einen Button aufgerufen. Events liegen jeweils in einem Subdirectory mit dem selben Namen wie das Modul, welches wiederum meist wie die Tabelle heißt, die es verwaltet. Skript-Teile, die von anderen Skripten importiert werden, beginnen mit einem "_". Man kann frei entscheiden, weitere geeignete Directory-Strukturen zu bauen, um seine Skripte zu organisieren.

config.txt

Das gerade aktive Konfigurations-Profil.

module.txt

Moduldefinitionen im Text-Format. Liegt in der Datenbank und wird dort vom "Module"-Modul verwaltet. Dieses File entsteht durch Export aus der Datenbank und dient dem Mergen mit Subversion.

translations.txt

Enthält Übersetzungen fester Begriffe. Die Daten kommen aus dem Übersetzungsmodul. Zu übersetzende Begriffe erkennt das System in IntarScript am <trans> Formatierkommando und in den Templates an den <|...|> tags. translationsm.arc ist ein binäres Archiv zur Beschleunigung des Startvorgangs. Es wird vom System automatisch aktuell gehalten.

nppModel.script

Ein Dokumentationsfile, welches alle Tabellen mit allen Attributen des Meta-Datenmodells mit "label"-Statements auflistet. Dient der Unterstützung im Notepad++. Mit den Spracherweiterungen für IntarScript listet Notepad++ in der Function List die Labels auf und man kann direkt hinspringen und so sehr schnell im Modell navigieren.

stdScript0.js und stdScript1.js

Das Javascript, welches IntarS 7 clientseitig unterstützt. Will man davon eine abweichende eigene Logik einbringen, sind diese beiden Files ins _K_000201-Directory zu kopieren und anzupassen.

systemtab.txt

Layout-Info für den System-Reiter in den Modulen. Sollen diese anders aussehen, ist dieses File ins _K_000201-Directory zu kopieren und anzupassen.

TempScriptTemplate.txt

Ist die Basis für das Beispiel-Script, welches pro Modul vom gleichnamigen Button im System-Reiter erzeugt wird. Will man es anders haben, ist dieses File ins _K_000201-Directory zu kopieren und anzupassen.

11.3 neue Felder anlegen

1. Suchen Sie die Tabelle im Modell.
2. Erfassen Sie das Feld.
3. Klicken Sie auf "db to idm".
4. Klicken Sie auf "Neustart".
5. DB-Differenzen autoadjust.bat
6. Rufen Sie das Modul auf.
7. Klicken Sie auf "Config" und blenden das Feld an der gewünschten Stelle ein.

11.4 neue Buttons erstellen

1. Schreiben Sie ein Script.
2. Testen Sie es über "Temp-Script".
3. Die Tabelle erscheint im Modell.
4. Erfassen Sie die Buttons und geben diese im Script an.
5. Erfassen Sie, falls nötig, Parameterfelder.
6. Klicken Sie auf "db to idm".
7. Klicken Sie auf "Neustart".
8. Rufen Sie das Modul auf.
9. Klicken Sie auf "Config" und blenden Sie die Buttons sowie die evtl. vorhandenen Parameterfelder an der gewünschten Stelle ein.

11.5 neue Module erzeugen

1. Erstellen Sie, falls nötig, eine neue Tabelle.
2. Gehen Sie in die Modulverwaltung.
3. Klicken Sie auf "Neu".
4. Geben Sie "interner Name", "guiName" und "Unterbereich" an. Wählen Sie die Tabelle aus.
5. Klicken Sie auf "Speichern".
6. Klicken Sie auf "Neustart".

11.6 Drucken

IntarS 7 erstellt für Druckausgaben PDF-Dateien (von Direktansteuerung abgesehen). Die PDF-Generierung erfolgt über das Framework ohne Umweg über ein Zwischenformat. XSL-FO und FOP kommen ebenfalls nicht zum Einsatz. Vielmehr werden aus in IntarScript eingebetteten Druckbefehlen direkt die PDF-Operanden erzeugt.

Die PDF-Erzeugung erfolgt sehr schnell und das erzeugte PDF ist von hoher Qualität und geringer Größe.

Das erzeugte PDF wird entweder über ein Plugin im Firefox oder Chrome als Preview angezeigt und kann von dort gedruckt werden oder es wird als Direkt-Druck ohne Preview direkt an einen Drucker geschickt. Welcher Drucker das ist, wird in der Drucker-Konfiguration pro Arbeitsplatz und Druckerklasse festgelegt. Ein typischer Anwendungsfall sind Pick-Etiketten, die direkt im Lager auf einem Label-Drucker ausgedruckt werden.

Das einfachste Druckscript sieht so aus:

```
newPDF
newPage
v 100,200,l,'Hello World
finalizePDF
```

Es erzeugt ein neues PDF, darin eine neue Seite und druckt 10 mm von links und 20 mm von oben linksbündig den Text "Hello World". Anschließend wird das PDF abgeschlossen und Angezeigt.

Wichtige Druckausgaben, z. B. Rechnungen, werden automatisch ins Dokumentenmanagement eingebracht und archiviert.

Die Vorschau im Plugin funktioniert, indem das PDF in ein Temp-Directory kopiert wird, auf welches der Apache Webserver Zugriff hat. Dann wird über Javascript das PDF in einem neuen Tab geladen und angezeigt.

Ältere IntarScripts, welche PDF-Dokumente erzeugen, haben die Endung .cpdf (create pdf). Darauf sollte alledings bei neueren IntarScripts verzichtet werden, da diese Trennung nicht mehr notwendig ist.

11.7 Schnittstellen

Bsp. einer datanorm-Import Schnittstelle

```
// Lieferantendaten von CD (Preislisten) importieren in vid_datanorm
// Bestellnummern ändern sich praktisch nicht
// 2 Lieferanten können die gleiche Vergleichsnr. haben
// preis = fremdek / waehrung.kurs
// mit upload machen und dann dokument angeben
// in $p_lieferant wird Lieferantennr. erwartet

$p_lieferant,=,30
gosub commonF
end

sub splitLine
 invk $line,csvFieldsWithSeparator:andEscape:,\t,""
 $a1,=,$_rv
 if $a1.count,<,4
  $artnr,=,
  return
 endif
 $artnr,oai,$a1,0
 $bestelltext,oai,$a1,1
 $fremdek,oai,$a1,2
 $fremdek,=,$fremdek.dbMoneyFromGui
endsub

sub commonF
 // preConditions
 $lief_eo,getEOPkValue,vid_kunde,$p_lieferant
 if $lief_eo,!FILLED,
  logi "Lieferant existiert nicht
  end
 endif
 // waehrung des Lieferanten
 $w,=,$lief_eo.waehrung_pb.*eo
 if $w,!FILLED,
  logi "Waehrung des Lieferanten nicht vorhanden.
  end
 endif
 $kurs,=,$w.kurs

 if $p_document,!FILLED,
  logi "bitte Dokument angeben, von dem importiert werden soll
  logi "ggfs. Dokument zuerst uploaden
  markError p_document
  end
 endif
 $doc,getEOPkValue,document,$p_document
 if $doc,!FILLED,
  logi "Dokument fuer Import nicht gefunden.
  markError p_document
```

```
 end
endif
$path,spf,'%@/Ressources/documents/%@,$_MANDANTPATH,$doc.filename
$file,swcof,$path
if $file,!FILLED,
 logi $path," nicht gefunden.
 end
endif

$a,css,$file,\n
autorelease
$j,count,$a
log 'importiere ,<int>$j,' Artikel von, $p_lieferant
$newCount,=,0
$updCount,=,0
$ignoreCount,=,0
$i,=,0
foreach line,$a
 $i,+,1
 $m,=,$i
 $m,m,1000
 if $m,==,0
  autorelease
  $log,spf,'%i verarbeitet von %i,$i,$j
  log $log
 endif

 // jeweils spezielle Aufbereitung der Zeile
 gosub splitLine
 if $artnr,!FILLED,
  // war keine gueltige Zeile
  continuefor
 endif
 // logi 'ok
 $qf,spf,'artnr = '%@' and lieferante = '%@',$artnr,$p_lieferant
  // logi $qf
 $dn,getEOQf,vid_datanorm,$qf
 if $dn,FILLED,
  if $p_with_shift,eq,J
   %parmDict.p_eo,=,$dn
   execute _Application/vid_datanorm/shift_preise
  endif
  gosub dnVersorgen
  updat_eo $dn
  $updCount,+,1
 else
  // falls es die Nr. gibt, damit Artikelstamm feststellen und
  // neue Vgl.Nr. dafuer erzeugen
  $qf,spf,'artnr = '%@' and lieferante = '',$artnr
  $dn,getEOQf,vid_datanorm,$qf
  if $dn,FILLED,
   // eine vgl.nr. ohne Lieferant fuer diesen Lieferanten nehmen
   gosub dnVersorgen
   updat_eo $dn
```

```
      $updCount,+,1
     else
      // eine vgl.nr. eines anderen lieferanten, um artikelstamm zu bekommen
      $qf,spf,'artnr = '%@',$artnr
      $dn_other,getEOQf,vid_datanorm,$qf
      if $dn_other,FILLED,
       $dn,newEO,vid_datanorm
       gosub dnVersorgen
       $dn.masterkey,=,$dn_other.masterkey
       $newCount,+,1
       insrt_eo $dn
      else
       $ignoreCount,+,1
      endif
     endif
    endif
   endfor

   logi <int>$updCount,' Bestellnummern aktualisiert
   logi <int>$ignoreCount,' neue Bestellnummern ignoriert
   logi <int>$newCount,' neue Bestellnummern erzeugt
  endsub

  sub dnVersorgen
   $dn.fremdek,=,$fremdek
   $dn.waehrung_pb,=,$w.primaryKey
   $dn.lieferante,=,$p_lieferant
   $dn.bestelltext,=,$bestelltext
   $dn.preis,=,$dn.fremdek / $kurs
   $dn.artnr,=,$artnr
  endsub
```

batch-Schnittstellen

Zeitlich gesteuert. Es werden Daten aus anderen Systemen importiert oder für andere Systeme bereitgestellt.

SOA, XML RPC

Eingebauter leichtgewichtiger XML Parser, TCP/IP Socket Kommunikation;

Beispiele:

```
// Umsatzsteuer-ID pruefen
$ma,newArray,10
$ma,a,'DE259892620,'AB123456789012,name,ort,plz,strasse,N
$server,=,"evatr.bff-online.de
$uri,=,"/
$methodName,=,evatrRPC

invk
     %application,httpPostXMLRPC:host:methodName:params:single:,$uri,$server,$
     methodName,$ma,J
$NSString,class,NSString
invk $NSString,stringWithData:,$_rv
$s,=,$_rv.stringBySubstitutingXMLEscapes
```

```
log $s

// -----------------------------------
// Zugriff auf Jargon File
$server,=,'scripts.incutio.com
$uri,=,'/xmlrpc/services/jargonfile.php
$methodName,=,'jargon.about
invk
     %application,httpPostXMLRPC:host:methodName:params:single:,$uri,$server,$
     methodName,nil,N
$NSString,class,NSString
invk $NSString,stringWithData:,$_rv
log $_rv
```

11.8 Dokumentation

Die Syntax der Dokumentationssprache lehnt sich an LaTeX 2e an:

```
\h Überschrift
\tt teletype-Text eine Zeile
\tt* Beginn/Ende Teletype Text
\ol beginnt eine ordered List; Zeilen werden als Listeneinträge dargestellt;
    Ende durch Leerzeile;
\ul beginnt eine unordered List; Zeilen werden als Listeneinträge
    dargestellt; Ende durch Leerzeile;
\img {Name}{title} Bild; Name ist der Filename innerhalb
    $GLOBALCONFIGPATH/Helpdesk/Images; Bilder sollten zwischen 700 und 850
    px breit sein.
\% Kommentar
\import ${projectPath}/license.txt   Inhalt eines Files importieren
\import ${resourcePath}/Hilfe/Mandanten_spezifisches.txt
```

Diese Formatierungsbefehle müssen am Zeilenanfang stehen.

Folgende Variablen werden ersetzt:

- $ {product} wird ersetzt durch den im Config-Modul eingegebenen Produktnamen
- $ {mandant} wird ersetzt durch die Mandantennummer

Das funktioniert auch in Filenamen und Directorynamen.

Die Dokumentationssprache kommt für die Online-Hilfe und das Handbuch (PDF) gleichermaßen zum Einsatz.

Mit "Import" im Hilfe-Modul werden die Files neu eingelesen, indiziert und in die Datenbank gestellt. Dort können sie unter dem Hilfesystem gelesen werden. Sie sind durchsuchbar.

11.9 Gewährleistung

Nimmt ein Kunde selbst Veränderungen an den Scripten oder am Meta-Datenmodell vor, erlischt zu diesem Zeitpunkt die Gewährleistung.

11.10 neuer Mandant

Neuen Mandanten anlegen:

- Löschen Sie .../Local/Library/IntarS/_K_neu.
- Tragen Sie im "Home"-Modul im Register "Service" "Mandantennr. neu" ein, z. B. 000239.
- Klicken Sie anschließend auf "Neuer Mandant".
- Es wird das _K_neu-Verzeichnis gefüllt.
- Nennen Sie es um, z. B. in _K_000239.
- Legen Sie eine Datenbank , "intars_000239", an und füllen Sie sie mit einem Dump.
- Rufen Sie die Workbench auf und testen Sie, ob die Datenbank übereinstimmt, passen Sie diese ggf. an.
- Starten Sie den neuen Mandanten und starten Sie einmalig den Apache-Dienst neu.
- Rufen Sie die URL auf und stellen Sie das "Config"-Modul ein.

Tauschen Sie wichtige Grafik-Dateien in Resources/Images aus:

- login_logo.jpg: 400 x 120 px für das Login-Panel.
- logo.jpg: 400 x 120 px Mandanten-Defaultlogo für Briefkopf; im "Config"-Modul kann ein anderes Logo hochgeladen und verwendet werden.
- menu_logo.jpg: 150 px breit, erscheint im "Home"-Modul links unter dem Menü.
- Unterschrift.jpg: eingescannte Unterschrift, wird unter Dokumente gedruckt.
- product_logo.jpg: das Logo, welches beim ersten Aufruf der Hilfe dargestellt wird bis ein Thema ausgewählt wurde.

12 IntarScript

12.1 Konzepte

Warum eine eigene Sprache?

Es gibt sehr viele Programmiersprachen. Hätte man da nicht eine passende finden können?

1996 hatten wir eine gefunden: ObjectiveC. Die früheren Versionen von IntarS 7 waren vollständig in ObjectiveC programmiert. Die Architektur sah so aus, dass es viel Geschäftslogik gab, die auf ein ständig wachsendes Set von nützlichen technischen Grundfunktionen zugriff. Während sich die Grundfunktionen zu einem Framework verdichteten, wurde die Geschäftslogik ständig umgeschrieben. Sie bestand schließlich nur noch aus Listen von Funktionsaufrufen. Da lag es nahe, diese in Textdateien auszulagern und einen Interpreter zu konstruieren, der die Textdateien liest und die Funktionen aufruft. Somit war IntarScript erschaffen. Änderungen und Erweiterungen waren nun durch einfaches editieren der Textdateien möglich. Es musste nichts mehr kompiliert werden. Alles komplizierte, technische war im Framework. Während in den Textdateien - die seitdem Skripte heißen - das fachliche, betriebswirtschaftliche steht.

Es gibt inzwischen einen Namen dafür: DSL - Domain Specific Language (http://de.wikipedia.org/wiki/Domänenspezifische_Sprache).

IntarScript bietet DSL-Befehle für:

- PDF-Erzeugung
- Feld- und Dialogsteuerung
- Datenbank-Abstraktion
- Eingabeverarbeitung
- Auswertungen, Filtern, Suchen, Import-/Export-Schnittstellen
- Batch-Verarbeitung
- E-Mail, Telefax, Dokumentenmanagement

Technisches

IntarScript ist beeinflusst von C, ObjectiveC, bash, Ruby, Java, Perl, Basic und dBase.

Man hat neben den eingebauten Befehlen direkten Zugriff auf die Framework-Methoden:

- Application
- Session
- Module
- Datenmodell
- Benutzer
- Funktionalitäten für Barcode
- XML, XML-RPC
- Files
- Betriebssystem, Netzwerk

Die Kombination aus kompiliertem Framework und Interpreter ergibt eine Gesamtperformance, die nur wenig hinter der eines komplett kompilierten Systems zurückbleibt. Das liegt daran, dass zum einen beim Systemstart zunächst alle Skripte eingelesen, geparst und in eine Art Bytecode überführt werden. Zum anderen laufen beim Interpretieren die meisten Taktzyklen wieder in kompiliertem Code, da es sich im Wesentlichen um eine Liste von Funktionsaufrufen handelt.

Wäre das Framework dagegen ebenfalls in einer Skriptsprache implementiert (derartige ERP-Software-Systeme gibt es auch), wäre die Performance schlechter.

Aus einem Guss

Die gesamte Anwendungslogik ist in über 1000 Skripte geschrieben. Erweiterungen und Anpassungen erfolgen mit derselben Technik, mit der auch der Standard geschaffen wurde.

Formales

Skripte sind klein und einfach. Da sie standalone leben, lässt sich ohne Probleme programmieren.

Variablen sind standardmäßig im Scope eines Scriptes global, typenlos und haben kurze Namen. Natürlich gibt es auch lokale Variablen und Typprüfungen. Jedoch nur, wenn man extra etwas dafür tut.

Zum Programmieren genügt ein Text-Editor, z. B. Notepad++. Mit diesem arbeiten wir und für diesen bieten wir eine IntarS 7-Sprachdefinition an.

Als DSL (s. oben) ist IntarScript sehr effizient und erfordert wenig Schreibarbeit.

IntarScript lässt dem Programmierer keinen künstlerischen Freiraum. Es gibt genau ein Statement pro Zeile. Befehle und Operanden werden ordentlich mit Kommata getrennt. Der Schachtelung von Ausdrücken sind Grenzen gesetzt. Demzufolge sehen alle Skripte gleich aus. Sie sind übersichtlich und man findet sich auch in fremden Skripte sofort zurecht.

Namenskonventionen tragen zusätzlich zur Klarheit bei.

Die restriktive Syntax ist zudem ideal für die Versionsverwaltung subversion und WinMerge.

Convention over configuration

Oder auch "coding by convention" ist ein Grundprinzip von IntarScript. Es bedeutet, dass man nichts tun muss, wenn man ein normales Modul erstellt. Es werden so wenig Namen wie möglich verbraucht. Standardmäßig heißt ein Modul wie die Tabelle, die es verwaltet. Skripte heißen so wie der Button, der es aufruft.

Nur vom Standard abweichendes Verhalten wird implementiert, indem dafür vorgesehene Events aktiviert und mit Leben gefüllt werden. Auch die Directory-Struktur ist fest vorgegeben.

Programming by Example

Neugeborene lernen sprechen, indem sie Beispiele von gesprochenem aus ihrer Umwelt aufnehmen und mit der Zeit daraus Grammatikregeln ableiten. Genauso lernt man auch eine Programmiersprache. Ein Beispiel ist eindeutiger als hundert Seiten abstrakter Syntaxabhandlung. Mit IntarScript verfolgen wir dieses Prinzip. Die über 1000 Skripte dienen als Beispiele. Sie lassen sich optimal mit dem Notepad++ oder mit der in IntarS 7 integrierten Scriptverwaltung durchsuchen. Natürlich gibt es auch eine Sprachreferenz, um die Feinheiten nachzulesen. Ebenso nutzen wir die Suchfunktion und copy/paste. Das ist effektiver.

Wenn man, vor allem am Anfang, noch nicht weiß, wonach genau man suchen soll, hilft einem der Button "Modulscripte" im "System"-Register eines jeden Moduls weiter. Will man wissen, wie etwas bestimmtes in einem Modul programmiert wurde, muss man auf "Modulskripte" klicken und bekommt die Skripte aufgelistet, derer sich das Modul bedient.

12.2 Sprachreferenz

Befehlssyntax

Es gibt drei Arten von Befehlen in IntarScript: Zuweisungsbefehle, Verarbeitungsbefehle und Codemanagementbefehle. Jeder Befehl kommt in eine Zeile. Die Elemente - Name, Operator und Operanden - werden durch Kommata getrennt. Ein Befehl ist immer am Zeilenende zu Ende. Leerzeichen und Tabulatoren vor dem Befehl werden ignoriert. Man kann also einrücken, um Blöcke und Ablaufsteuerung zu verdeutlichen.

Zuweisungsbefehle

Diese haben die Form: Ziel,Name,Operand(en). Das Ziel ist eine Variable. Der Name ist der Name des Befehls, der Operand oder die Operanden sind wiederum Variablen oder Konstanten oder Ausdrücke. Der einfachste Zuweisungsbefehl heißt "=", z. B.:

```
$a,=,$b
```

weist der Variable $a den Wert aus $b zu. Es gibt jedoch auch komplexere Zuweisungsbefehle, z. B.:

```
$a,getEOsQSoa,vid_lager,$q,$soa
```

beschafft ein Array von EnterpriseObjects aus der Datenbanktabelle "vid_lager" unter Anwendung des Qualifiers $q und des Sortorderarrays $soa (serverside sort).

```
Ein Bespiel:
$a1,=,'12
$a2,=,'23
$a3,=,'10
$ziel,spf,'%@ kg %@ Liter %@ Stück,$a1,$a2,$a3

Der Inhalt von $ziel würde dann wie folgt aussehen:
12 kg 23 Liter 10 Stück
```

Verarbeitungsbefehle

Diese haben die Form: Name[{' '|','|'\t'}Operand[,Operand...]]. Der Name ist der Name des Befehls, der Operand oder die Operanden sind Variablen oder Konstanten oder Ausdrücke, z. B.:

```
logi "Hello World
```

gibt den String "Hello World" interaktiv (auf der Oberfläche) aus. Zwischen Befehl und erstem Operand steht genau ein Leerzeichen oder ein Komma oder ein Tab. Man beachte dass am Ende kein Anführungszeichen steht, da der Befehl und damit der String am Zeilenende aufhört.

Codemanagementbefehle

import Skriptname

Damit werden die Befehle eines anderen Scripts importiert. Der Skriptname ist der Pfadname des Skripts relativ zum /Script- Verzeichnis. Die Extension ist wegzulassen. Zur Unterscheidung von normalen Scripts beginnt der Name von importierten Scripts mit einem "_". Enthält der Skriptname die Zeichenkette "$mandant", wird diese vor dem Laden zur Mandantennummer expandiert. Der Import geschieht einmalig beim Start der Anwendung bzw. jeweils beim Neuladen der Skripte. Diese Technik ist zur Wiederverwendung kleinerer, unkritischer Codesequenzen gedacht. Zu beachten ist die enge Verflechtung mit dem importierenden Skript. Eine stärkere Kapselung erhält man mit dem "execute" Befehl. Import ist vergleichbar mit der #include Compiler-Directive in C. Z. B.:

```
import _hilfslinien
```

subscript:Skriptname

Hiermit kann man mehrere Skripte in einer Skript-Datei unterbringen. Normalerweise hat man ein Skript pro Datei. Bei vielen kleinen Skripten ist dies unübersichtlich und mühsam. Dann schreibt man mit subScript:Skriptname ... end die Skripte zusammen in eine Datei. Sie werden vom System unter diesem hier angegebenen Skriptnamen gefunden und nicht mehr wie sonst über den Dateinamen. Events-Skripte werden normalerweise so implementiert. Z. B.:

```
subScript:didUpdate
    import _peo_durchrechnen
end
```

#eo Variable Tabellenname

Deklariert, dass die Variable ein EO der angegebenen Tabelle enthalten muss. Diese Information nutzt der Interpreter, um feststellen zu können, ob mit gültigen Attributnamen darauf zugegriffen wird. Bei ungültigen wird zur Laufzeit ein Namechecking-Fehler geloggt.

#warning text

Damit schreibt man eine Warnung in das Script, die beim Laden des Scripts ausgegeben wird. Auf die Ausführung hat das keinen Einfluss. Warnungen dienen der Kommunikation zwischen Entwicklern und als Notizblock.

#didClickRowTitle "d=duplicate i=invisible p=protected m=mandatory l=löschen

Versorgt den tooltip auf dem ersten Trefflisten-Auswahlfeld.

Kommentare

Kommentare werden mit // für einzelne Zeilen und /*....*/ mehrere Zeilen analog C angegeben. Bsp.:

```
// das ist ein Kommentar
    // hier ein eingerückter
/*
    alle
    Befehle
    hier drin sind auskommentiert
*/
Wichtig ist dabei, dass das End-Tag */ in einer neuen Zeile steht.
```

Notation von Operanden

{Formatierung}Ausdruck{Data-Option}

Einem Operanden kann man eine oder mehrere optionale Formatierung(en) voranstellen. Formatierungen können kombiniert werden. Sie werden nach Auswertung des eigentlichen Ausdrucks angewendet. So kann z. B. mit <int> das Ergebnis ganzzahlig gemacht werden.

Der Ausdruck des Operanden kann eine Konstante, eine einfache Variable oder ein zusammengesetzter Ausdruck sein.

Konstanten werden einfach hingeschrieben. Ohne ein vorangestelltes Anführungszeichen wird die Zeichenkette bis zum nächsten Leerzeichen oder Komma als Konstante interpretiert.

Ein vorangestelltes einzelnes Anführungszeichen interpretiert die folgende Zeichenkette bis zum ersten Komma als Konstante.

Will man Kommata in einer solchen Konstante angeben, geschieht dies mit <comma>.

Ein vorangestelltes doppeltes Anführungszeichen interpretiert die folgende Zeichenkette bis zum Zeilenende als Konstante. Mehrzeilige Konstanten werden mit dem "ml"-Zuweisungsbefehl definiert.

```
logi "Hello, World
```

Variablen bestehen aus einem Namen und vorangestelltem Kennzeichen "%" oder "$". Das Kennzeichen besagt, wo die Variable ist. "$"-Variablen leben im Variablenraum des Skripts. Sie werden dort erzeugt und vergehen auch wieder mit dem Ende des Skripts. Variablen, die mit "$l_" beginnen, sind zusätzlich noch lokal in ihrer Subroutine, in der sie zum ersten Mal genannt werden. Dagegen leben Variablen mit Prefix "%" außerhalb des Skripts. Das Skript fragt dazu seine Datasource. Variablen können gelesen und gesetzt werden, z. B.:

```
$a,=,1
```

der Script-Variablen $a wird der konstante Werte 1 zugewiesen.

```
%parmDict.p_abc,=,xyz
```

in das parmDict der Datasource wird unter dem Schlüssel "p_abc" der konstante String "xyz" geschrieben.

Variablen können über den sog. Keypath benannt werden. Ein Keypath ist die Verkettung von Namen mit Punkten. Diese Syntax wurde in C zur Navigation von Datenstrukturen ersonnen. In SQL ist sie ebenfalls bekannt, um Felder in Tabellen anzusprechen. Heute ist die Keypath-Notation vor allem durch Java bekannt geworden. Sie ist mittlerweile Allgemeingut in allen modernen Sprachen. Ein Keypath wird Element für Element abgearbeitet, indem das erste Element nach dem Namen des zweiten gefragt wird. Das Ergebnis dieser Frage wird wiederum nach dem Namen des dritten gefragt usw. Als Besonderheit in IntarScript können Keypath-Elemente und auch Teile davon wiederum Variablen sein. Daraus ergeben sich sehr viele Möglichkeiten, z. B.:

```
$qrs.$tuv.wxy
```

$qrs wird nach dem Wert gefragt, der in $tuw steht; das dabei erhaltene Objekt wird nach "wxy" gefragt.

```
$abc.def$ghi
```

$abc wird nach dem Wert gefragt, dessen Name mit "def" beginnt und mit dem Wert endet, der in $ghi enthalten ist.

Für die Navigation in einem XMLObject gibt es spezielle Prefixe:

```
$entry,=,$contact.>t:EmailAddresses.>Key=EmailAddress1>t:Entry
```

">name" bedeutet "child with name "

"a>name" "Array of children with name

">Key=EmailAddress1>t:Entry" "Array of children with Name t:Entry und tagAttribute Key = EmailAddress1

Ausdrücke

Zusammengesetzte Ausdrücke werden aus Ausdrücken gebildet, die mit den Grundrechenarten (+,-,*,/), der Modulo-Division (m) und Klammern verknüpft werden, z. B.:

```
($a + $b) / $c)
```

Zu beachten sind die Leerzeichen zwischen Operatoren und Ausdrücken. Sie sind notwendig.

Die Data-Option ist eine Spezialität von EOs (EnterpriseObjects). Man kann mit ".*eo" das EO anstatt seines foreignkeys beschaffen. Mit ".*bez" wird die Oberflächenbezeichnung des internen Werts einer Werteliste besorgt. Z. B.:

```
$ap.artikelnum.*eo
```

besorgt den Artikelstamm des $ap, wenn $ap ein EO und artikelnum eine Relation zum Artikelstamm ist. Ohne das ".*eo" würde nur die Artikelnummer geliefert. Data-Optionen werden immer am Ende angegeben. Relationen innerhalb des Keypaths werden von IntarScript von selbst aufgelöst. Mit .*eo wird nur gesteuert, ob am Ende des Keypaths auch noch das EO geholt werden soll.

Alle Formatieroptionen

```
<ev>        expand Variables
<date>      guiDate machen
<week>      Kalenderwoche
<month>     Monat 2-stellig
<monthName> Monat Abkürzung
<yw>        Jahr/Kalenderwoche
<year>      Jahr 4-stellig
<yy>        Jahr 2-stellig
<wdayn>     Wochenname kurz
<int>       integer
<floor>     floor()
<ceil>      ceil()
<intnd>     integer = <int>
<f0>        GUI 0 NK
<f1>        GUI 1 NK
<money>     GUI 2 NK
<f2>        GUI 2 NK
<f3>        GUI 3 NK
<f1nd>      GUI 1 NK no dot
<f2nd>      GUI 2 NK no dot
<f3nd>      GUI 3 NK no dot
<uc>        uppercase
<lc>        lowercase
<stri..>    rechtsbündig, GUI Int, auf Laenge .. mit Spaces gefüllt; falls zu
     lang ####
<strm..>    rechtsbündig, GUI Moneyformat, auf Laenge .. mit Spaces gefüllt;
     falls zu lang ####
<strs..>    rechtsbündig, string, auf Laenge .. mit Spaces gefüllt bzw.
     rechts abgeschnitten
<stls..>    linksbündig, string, auf Laenge .. mit Spaces gefüllt bzw. rechts
     abgeschnitten

<trans>     Übersetzen in Sprache, die mit dem "lang"-Befehl gesetzt wurde
<tri>       translation inline; String enthält Übersetzungen, mit "|"
     getrennt; fallback auf <trans>

<trim..>    abschneiden auf .. Zeichen
<round..>   auf .. NK runden
'           wörtlich bis zum nächsten Komma
"           wörtlich bis zum Zeilenende
```

Environment

```
folgende Variablen werden schon vorab vom System gesetzt:
$_MANDANTPATH   MANDANTPATH

$_user          angemeldeter Benuzter
$_actionName    zuletzt aufgerufene Aktion bei Attribute-Buttons
$_today         Datum im dbFormat
$_rv            Returnvalue des letzten invk-Befehls
                und des expression-scripts von attributen
                und des ganzen Variablenraumes eines aufgerufenen Scripts
```

```
$scriptRC       Rückgabewert, von "will..." Event-Scripts;
                wenn "N" geliefert wird, wird dies als Ablehnung
                durch das Event-Script gewertet;
$_myFM          Filemanager

$_lastFont      der zuletzt gesetzte Font
$_lastSize      Größe des zuletzt verwendeten Font
$_path          kompletter Pfad, in welches das PDF
                geschrieben wird
$_path_back_slash     dito mit backslashes
$_name          Name des PDF
$_fn_ext        Filename u. extension des PDF-Files
$_url           URL des erzeugten PDF
$_ts            current tempName (Filename des PDF in temp-Directory)

%parmDict        MutableDictionary in der datasource zur freien Verwendung;
                z. B. für Parameterübergabe an andere Scripts;
                was sich im parmDict befindet, wird vor dem Start eines
     Scripts
                in dessen Variablenraum gestellt;
%datasource     Datasource, i. a. das Modul oder Plugin; Objekt, unter dessen
     Kontrolle
                das Script abläuft
%session        die Session
%session.ueo    User-EO; ... .vorname .nachname
%session.parmDict
                MutableDictionary zur freien Verwendung;
                bleibt für die Dauer der Session erhalten;
%session.pbpdf  das momentane PDF
%session.pbpdf.currentFontName
%session.pbpdf.currentSize
%session.pbpdf.restString
%session.pbpdf.pdfPath
%application    die Application / Instanz
%application.currentPDFVorgang
                Form-Name des PDFs und Nummer des Belegs; wird für
                PDF-Archiv verwendet;
%application.mandant
                die Mandanten-nr. ohne _K_
%application.myDD
                das Data-Dictionary / Modell
%application.parmDict
                MutableDictionary zur freien Verwendung;
                wird in jedem Request-Response-Zyklus wieder geleert;
%application.staticDict
                MutableDictionary zur freien Verwendung;
                wird nie geleert;
%application.now
                Systemzeit als normalized Date
```

Namenskonventionen

```
Environment, Systemvariablen: _...
lokale Variablen in subroutines: l_...
a,a1 Array allgemein
ma Mutable Array
md Mutable Dictionary
s String allgemein
i,j Integer allgemein
d Dictionary
eo  EO
k Key
p_ Parameter allgemein
q Qualifier
qf Qualifierformat
..f Formatstring allgemein
sql Sql Befehl
v Value
bu_... Button
po_... Plugin Object

\h Konstanten
\tt
 \n      newLine
 "\n     newLine
 \t      tab
 "\t     tab
 \r\n    windows cr lf
 "\r\n   windows cr lf
```

12.3 Ausgabe

clearErrors

Syntax: clearErrors

```
löscht die blinkenden Meldungen, die u.a. mit logi erzeugt wurden

clearErrors
```

diaAlert

Syntax: diaAlert [msg,confirm]

```
blendet am Ende der Transaktion eine modale Box mit der Meldung "msg" auf und
wartet auf Bestätigung durch Click auf den mit "confirm" beschrifteten
     Button.
Die Defaultwerte für "msg" und "confirm" sind "Achtung" und "OK".
Verhindert keine Verarbeitung, sondern informiert nur.

diaAlert "Kunde hat Kreditlimit überschritten.
```

displayHTML

Syntax: displayHTML string

```
gibt "string" in einem neuen Fenster als HTML aus.

$html,=,"<html><head></head><body>Hello IntarS.</body></html>
displayHTML $html
```

displayText

Syntax: displayText string

```
gibt "string" in einem neuen Fenster als Text aus.

displayText 'Hello IntarS.
```

log,logs

Syntax: log string[,string1...n]

```
gibt "string" in der Log-Datei aus. Lange Strings werden abgeschnitten.

log "hello IntarS.
```

logi

Syntax: logi string[,string1...n]

```
gibt "string" interaktiv und in der Log-Datei aus. Lange Strings werden
abgeschnitten.

logi "hello IntarS.
```

12.4 Datum

d+

Syntax: target,d+,anzahl,interval

```
auf das "target" Datum im normalized Date Format die "anzahl" "interval"
aufaddieren;
interval kann sein: y,m,d,w,wd,H,M,S


$d1,d+,3,m
```

d-

Syntax: target,d-,anzahl,interval

```
von dem "target" Datum im normalized Date Format die "anzahl" "interval"
abziehen;
interval kann sein: y,m,d,w,wd,H,M,S


$d1,d-,3,m
```

dd

Syntax: target,dd,d1,d2

```
ermittelt Differenz der beiden normalized Dates "d1" - "d2" als Sekunden

$i,dd,$d1,%application.now
```

nowFormat

Syntax: target,nowFormat,dateFormat

```
liefert den momentanen Zeitpunkt als String aufbreitet gemäß dem "dateFormat"

$s,nowFormat,"%Y%m%d%H%M%S
```

12.5 Datenbank

allq VERALTET !!!

Syntax: target,allq,

```
liefert einen Qualifier (1=1), der immer zu "wahr" evaluiert;
entspricht:
$q,=,'(1=1)

$q,allq,
```

arraySQL

Syntax: target,arraySQL,sql

```
evaliert die "sql"-Anweisung und liefert ein Array of Dictionaries mit den
Spaltennamen und zugehörigen Werten der Records im Resultset

$a,arraySQL,"select artikelnum,bezeichnun from vid_lager
```

arraySQLs

Syntax: target,arraySQLs,sql

```
evaliert die "sql"-Anweisung und liefert ein Array of Strings mit den Werten
der einzigen Spalte der Records im Resultset

$a,arraySQLs,"select artikelnum from vid_lager
```

dbnr

Syntax: dbnr n

```
wählt die Datenbank Nr. "n" aus; alle folgenden Datenbankoperationen beziehen
sich dann darauf bis zum nächsten dbnr Befehl oder spätestens bis zum Ende
     der
Request-Verarbeitung.

dbnr 1
```

delet_eo,delete

Syntax: delete eo

```
löscht den zu dem EO gehörenden Datensatz in der Datenbank;
Erfolg wird als J/N in $_rv gemeldet.
Events werden keine ausgelöst.
Das EO muss einen primaryKey haben.
Das EO bleibt als Objekt im Speicher bestehen.

delete $p_eo
```

getEOPkValue

Syntax: target,getEOPkValue,tableName,pkValue

```
holt ein EO aus der Tabelle "tableName" mit dem Primary Key "pkValue"

$eo,getEOPkValue,vid_lager,1234
```

getEOQf,getEOQ

Syntax: target,getEOQf,tableName,qualifier

```
holt das erste EO aus der Tabelle "tableName", das mit der where-Condition
"qualifier" matcht


$eo,getEOQf,vid_lager,'bezeichnun like '%bla%'
```

getEOs12jwsoc

Syntax: target,getEOs12jwsoc,tn1,tn2,jc,wc,oc,offset,count

```
get joined EOs: aus Tabelle "tn1", optional joined with Tabelle "tn2" über
join-Condition "jc", matching Where-Condition "wc", sortiert gemäß
Order-Condition "oc", ab "offset" und limitiert auf "count";
der allgemeinste Befehl, um EOs zu holen; die dazugejointe Tabelle "tn2" kann
in der "wc" und "oc" angesprochen werden über t2. Die EOs enthalten nur
     Felder
von Tabelle "tn1". Der join ist hilfreich, wenn man z.B. Positionssätze lesen
will, die eine bestimmte Eigenschaft des Kopfes haben.

$a,getEOs12jwsoc,ri_plposten,,,$qf,$soa,$p_offset,$p_pagelength
```

getEOsQf,getEOsQ VERALTET !!!

Syntax: target,getEOsQf,tableName,qualifier

```
holt alle EOs aus der Tabelle "tableName", die mit der where-Condition
"qualifier" matchen
s. auch: getEOstws

$a,getEOsQf,vid_lager,'bezeichnun like '%bla%'
```

getEOsQSoa VERALTET !!!

Syntax: target,getEOsQSoa,tableName,qualifier,sortOrderArray[,limit]

```
holt (optional max. "limit" Anzahl) EOs aus der Tabelle "tableName", die mit
dem "qualifier" matchen, sortiert nach "sortOrderArray";
"qualifier" ist die where-condition
s. auch: getEOstws

$soa,soaFrom,artikelnum
$a,getEOsQSoa,vid_lager,'bezeichnung like 'bla%',$soa
```

getEOstws

Syntax: target,getEOstws,tableName,whereCondition,sortCondition

```
holt EOs aus der Tabelle "tableName", die mit "whereCondition" matchen,
sortiert nach "sortCondition"; Kurzform von getEOs12jwsoc, wenn man nicht
joinen muss und kein offset/limit braucht;

$a,getEOstws,vid_lager,'bezeichnun like 'bla%','ldate desc
```

insrt_eo,insert

Syntax: insert eo

```
inserted den das EO repräsentierenden Datensatz in der Datenbank;
Erfolg wird als J/N in $_rv gemeldet.
Events werden keine ausgelöst.
Das EO muss einen primaryKey haben.
EO bedeutet "Enterprise Object". Ein EO ist im IntarS Database-Abstraction
Layer die Repräsentation eines Datensatzes aus einer Datenbanktabelle.


$eo,newEO,vid_lager
$eo.nummer,=,1234
$eo.bezeichnun,=,"bla fasel
insert $eo
```

newEO

Syntax: target,newEO,tableName

```
liefert ein neues, leeres EO für die Tabelle "tableName"

$eo,newEO,vid_lager
```

nothingq

Syntax: target,nothingq,

```
liefert einen Qualifier (1=2), der immer zu "falsch" evaluiert;
entspricht:
$q,=,'(1=2)

$q,nothingq,
```

q,sqlq VERALTET !!!

Syntax: target,q,whereCondition

```
macht nichts mehr ausser einer direkten Zuweisung mittels OPC_Z_set; man
     könnte
stattdessen auch schreiben:
$q,=,$qf


$q,q,$qf
```

qand VERALTET !!!

Syntax: target,qand,array

```
verknüpft die Qualifier (=where-Bedingungen) in "array" mit "and" zu einer
einzigen where-Bedingung;
geht mittlerweile geschickter mittels konventioneller Stringbearbeitung, z.B:
$qf,cjs,$a,' and

$q,qand,$a
```

qor VERALTET !!!

Syntax: target,qor,array

```
verknüpft die Qualifier (=where-Bedingungen) in "array" mit "or" zu einer
einzigen where-Bedingung;
geht mittlerweile geschickter mittels konventioneller Stringbearbeitung, z.B:
$qf,cjs,$a,' or

$q,qor,$a
```

setFetchCond

Syntax: target,setFetchCond,qualifier,tableName,sortOrderArray[,limit]

```
liefert einen FetchRequest für sequentielle Verarbeitung von EOs; wird für
große Resultsets verwendet, die nicht in den Speicher passen; intern werden
     die
primaryKeys der matching records in eine temp table weggesichert und in einer
Schleife abgearbeitet wobei dann das jeweils dazugehörige echte EO gelesen
     wird.

$pbfr,setFetchCond,$qf,vid_lager,$soa,1000
if $pbfr,FILLED,
 $eo,=,$pbfr.nextEO
 while $eo,FILLED,
  ... Verarbeitung
  $eo,=,$pbfr.nextEO
 endwhile
endif
```

singleRecordSQL

Syntax: target,singleRecordSQL,sql

```
evaluiert die "sql"-Anweisung und liefert ein Dictionary mit den Feldnamen
     und
zugehörigen Werten des aus einem einzigen Datensatz bestehenden Resultset.

$d,singleRecordSQL,"select artikelnum,bezeichnun from vid_lager where
     artikelnum = 1234
```

singleValueSQL

Syntax: target,singleValueSQL,sql

```
evaluiert das SQL und liefert einen Ergebniswert als String

$s,singleValueSQL,"select max(anzahl) from vid_anposten
```

soaFrom VERALTET !!!

Syntax: target,soaFrom,fieldList

```
erzeugt ein SortOrderArray aus der "fieldList"; diese enthält Komma-getrennt
die Feldnamen, optional mit angehängtem ":d" (descending) ":a" (ascending)
     und
"n" (numeric); ":dn" ist descending numeric;
geht mittlerweile geschickter mit -[NSString soa]; ausserdem kann getEOstws
direkt mit der whereCondition arbeiten und benötigt kein SortOrderArray mehr.
Nur zum Sortieren von Arrays benötigt man noch ein SortOrderArray.

$soa,soaFrom,"anzahl:d,artikelnum
```

sql

Syntax: sql sqlStatements

```
führt die "sqlStatements" in der Datenbank aus. In $_rv werden affectedRows
bzw. bei einem Fehler -1 geliefert. Die "sqlStatements" können mehrere durch
newline getrennte Befehle enthalten.

sql "delete from vid_angebot
```

updat_eo,update

Syntax: update eo

```
schreibt die geänderten Werte eines EO zurück in Datenbank;
Erfolg wird als J/N in $_rv gemeldet.
Events werden keine ausgelöst.
Das EO muss einen primaryKey haben.
Danach sind die oldValues des EO geleert und die changedValues gefüllt.

update $p_eo
```

12.6 dBase

@

Syntax: @ zeile,spalte,value[,...]

```
schreibt ins scratchpad in "zeile" an "spalte" den "value";
wg. dBase Kompatibilität erst zeile, dann spalte; alle anderen IntarScript
Ausgabebefehle adressieren eine Position mit x,y.

das scratchpad emuliert einen zeichenorientierten Bildschirm wie DOS, VT52
o.ä.; es ist als Mutable Array implementiert und wird im %parmDict unter dem
Namen "scratchpad" gehalten und on demand angelegt;

@ 1,20,'bla fasel
```

@clear

Syntax: @clear

```
löscht den scratchpad-Inhalt;
das scratchpad emuliert einen zeichenorientierten Bildschirm wie DOS, VT52
o.ä.; es ist als Mutable Array implementiert und wird im %parmDict unter dem
Namen "scratchpad" gehalten und on demand angelegt;

@clear
```

dictAdd VERALTET !!!

Syntax: dictAdd dict,wert,key

```
fügt dem "dict", das ein Mutable Dictionary sein muss, den "wert" unter "key"
hinzu.
Geht mittlerweile geschickter:
$md.fasel,=,bla

dictAdd $md,bla,fasel
```

say

Syntax: say x,y,o,value

```
schreibt an die Koordinaten "x", "y" je nach orientation "o" (l,r) links-
     oder
rechtsbündig den "value";
das scratchpad emuliert einen zeichenorientierten Bildschirm wie DOS, VT52
o.ä.; es ist als Mutable Array implementiert und wird im %parmDict unter dem
Namen "scratchpad" gehalten und on demand angelegt;

say 10,1,r,<money>$summe
```

sayw

Syntax: sayw x,y,o,w,value

```
schreibt an die Koordinaten "x", "y" je nach orientation "o" (l,r) links-
     oder
rechtsbündig den "value" in Breite "w";
das scratchpad emuliert einen zeichenorientierten Bildschirm wie DOS, VT52
o.ä.; es ist als Mutable Array implementiert und wird im %parmDict unter dem
Namen "scratchpad" gehalten und on demand angelegt;

say 10,1,r,9,<money>$summe
```

12.7 Entwicklung

debug

Syntax: debug

```
ab hier schreibt IntarS jeden Befehl ins log, so dass man nachvollziehen
     kann,
was das Script macht; also eigentlich eher ein trace.

debug
```

gdb

Syntax: gdb

```
darauf kann im gdb mit:
b [Application gdb]
ein Breakpoint gesetzt werden

gdb
```

noDebug,endDebug

Syntax: endDebug

```
ab hier wird nicht mehr getraced.

endDebug
```

12.8 Druck

?

Syntax: ? string[,string1..n]

```
Drucken im C=64 Stil
gesteuert über diese Variablen:
$?line
$?size
$?leftMargin
$?topMargin
$?cr
$?font

newPDF
newPage
$?leftMargin,=,150
$?topMargin,=,150
$?cr,=,50
$?font,=,Courier
$?size,=,10
$?line,=,1
? "Hello IntarS
? "   Hello World
? "I like my Typewriter.
? a,b,c
?b "************
?b " be bold !
?b "************
? '
? -_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_
? '
$?size,=,20
? 'think big.
$?size,=,10
$i,=,0
while $i,<=,200
   if $?line,>,50
       newPage
    $?line,=,1
 endif
   ? <int>$i
   $i,+,1
endwhile
finalizePDF
```

?b

Syntax: ?b string[,string1..n]

```
Drucken bold im C=64 Stil

newPDF
```

```
newPage
$?leftMargin,=,150
$?topMargin,=,150
$?cr,=,50
$?font,=,Courier
$?size,=,10
$?line,=,1
? "Hello IntarS
? "   Hello World
? "I like my Typewriter.
? a,b,c
?b "***********
?b " be bold !
?b "***********
? '
? -_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_-_
? '
$?size,=,20
? 'think big.
$?size,=,10
$i,=,0
while $i,<=,200
   if $?line,>,50
       newPage
    $?line,=,1
 endif
   ? <int>$i
   $i,+,1
endwhile
finalizePDF
```

b VERALTET !!!

Syntax: b x,y,width,1,orientation,string[,string1...n]

```
begin block;
Fließtext drucken ab "x" "y" in Breite "width"
mit "orientation" = {l,r,c}
interpretiert .inline f,s Anweisungen
weiterdrucken mit bf unter Beachtung von %application.restString
besser: pdfLines

b 100,100,400,1,l,"Hello IntarS.
```

bf VERALTET !!!

Syntax: bf x,y,width,1,orientation

```
block Folge;
Fließtext drucken ab "x" "y" in Breite "width"
mit "orientation" = {l,r,c}
interpretiert .inline f,s Anweisungen
weiterdrucken unter Beachtung von %application.restString
besser: pdfLines

b $x1,$ty,$x12 - $x1,1,l,$fusstext
```

```
while %application.hasRestString,eq,J
 gosub nextLine
 bf $x1,$ty,$x12 - $x1,1,l
endwhile
```

embedFonts

Syntax: embedFonts

```
sorgt dafür, dass die Fonts auf jednfall embedded werden, auch wenn das nicht
notwendig wäre, weil nur die base14 fonts verwendet werden.

embedFonts
```

f

Syntax: f fontName[,size]

```
wählt den Font u. optional Größe aus, mit dem folgende Schriftausgaben
erfolgen. Verfügbar sind die base14 Fonts
0  Helvetica
1  Helvetica-Bold
2  Helvetica-Oblique
3  Helvetica-BoldOblique
4  Times-Roman
5  Times-Bold
6  Times-Italic
7  Times-BoldItalic
8  Courier
9  Courier-Bold
10 Courier-Oblique
11 Courier-BoldOblique
12 Symbol
13 ZapfDingbats
sowie FrutigerCE-Roman, Arial#20Unicode#20MS, WASP#2039, Free#203#20of#209,
Code128, Code#20EAN13, any TTF

newPDF
$maxty,=,1900
$ty,=,$maxty
$tx,=,100
$tx2,=,600
$abc,=,abcdefghijklmnopqrstuvxxyz1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ

$fonts,ml,
Helvetica
Helvetica-Bold
Helvetica-Oblique
Helvetica-BoldOblique
Times-Roman
Times-Italic
Times-BoldItalic
Courier
Courier-Bold
Courier-Oblique
```

```
Courier-BoldOblique
Symbol
ZapfDingbats
Arial#20Unicode#20MS
\end
foreach font,$fonts
 if $ty,>=,$maxty
  newPageQuer
  $ty,=,100
 endif
 f Helvetica,10
 v $tx,$ty,l,$font
 f $font
 v $tx2,$ty,l,$abc
 $ty,+,50
endfor
finalizePDF
end
```

fc

Syntax: fc shade

```
setzt die Schwarz-Weiss Font/Fill-Farbe von 0 = Schwarz ... 100 = Weiss

newPDF
newPage
fc 80
f Helvetica-Bold,70
v 200,500,l,"Hello IntarS
finalizePDF
```

fc#

Syntax: fc# hexRGBColor

```
setzt die RGB Font/Fill-Farbe als Hexwert mit jew. 00...FF für R, G, B

newPDF
newPage
fc# FF00AA
f Helvetica-Bold,70
v 200,500,l,"Hello IntarS
finalizePDF
```

fc255

Syntax: fc255 r,g,b

```
setzt die RGB Font/Fill-Farbe als 3 Dezimalzahlen mit jew. 0...255 für R, G, B

newPDF
newPage
$b,=,0
$ty,=,200
```

```
$endy,=,2800
$steps,=,30
$incb,=,255 / $steps
$incy,=,($endy - $ty) / $steps
$step,=,0
f Helvetica-Bold,$incy * 0.9 / 2.83
while $step,<,$steps
 fc255 255,0,$b
 v 200,$ty,l,"Hello IntarS
 $b,+,$incb
 $ty,+,$incy
 $step,+,1
endwhile
finalizePDF
```

fcrgb

Syntax: fcrgb r,g,b

```
setzt die RGB Font/Fill-Farbe als 3 Dezimalzahlen mit jew. 0...100 für R, G, B

newPDF
newPage
$b,=,0
$ty,=,200
$endy,=,2800
$steps,=,30
$incb,=,100 / $steps
$incy,=,($endy - $ty) / $steps
$step,=,0
f Helvetica-Bold,$incy * 0.9 / 2.83
while $step,<,$steps
 fcrgb 80,0,$b
 v 200,$ty,l,'Hello IntarS,80,0,<int>$b
 $b,+,$incb
 $ty,+,$incy
 $step,+,1
endwhile
finalizePDF
```

finalizePDF

Syntax: finalizePDF

```
macht render und previewPDF. In .cpdf scripts nicht erforderlich. Diese sind
jedoch veraltet

finalizePDF
```

gh

Syntax: gh x,y,h,path

```
druckt jpg Grafik "path" an "x", "y" (linke untere Ecke) auf Höhe "h"
     skaliert;
beginnt "path" mit /, ist es ein absoluter Pfad; sonst relativ zu
```

```
%application.resourcePath,/Images

newPDF
newPage
$x,=,200
$y,=,500
gh $x,$y,400,%application.logoName
l $x,$y,x,1000
l $x,$y,y,400
finalizePDF
```

gw,g

Syntax: gw x,y,w,path

```
druckt jpg Grafik "path" an "x", "y" (linke untere Ecke) auf Breite "w"
skaliert;
beginnt "path" mit /, ist es ein absoluter Pfad; sonst relativ zu
%application.resourcePath,/Images

newPDF
newPage
$x,=,200
$y,=,500
gw $x,$y,400,%application.logoName
l $x,$y,x,1000
l $x,$y,y,400
finalizePDF
```

kf

Syntax: kf x,y,r

```
zeichnet einen gefüllten Kreis an x,y mit Radius r mit geltender Fill-Farbe.

newPDF
newPage
$b,=,0
$x,=,100
$r,=,50
$rend,=,500
$xend,=,1900 - $wend
$y,=,200
$endy,=,2800
$steps,=,30
$incb,=,255 / $steps
$incy,=,($endy - $y) / $steps
$incr,=,$rend - $r / $steps
$incx,=,$xend - $x / $steps
$step,=,0
while $step,<,$steps
 fc 100
 kf $x,$y,$r + 10
 fc255 255,0,$b
 kf $x,$y,$r
 $r,+,$incr
```

```
 $b,+,$incb
 $x,+,$incx
 $y,+,$incy
 $step,+,1
endwhile
finalizePDF
```

kl

Syntax: kl x,y,r

```
zeichnet einen leeren Kreis an x,y mit Radius r mit geltender
Linien-Einstellung.

newPDF
newPage
$b,=,0
$x,=,100
$r,=,50
$rend,=,500
$xend,=,1900 - $wend
$y,=,200
$endy,=,2800
$steps,=,30
$incb,=,255 / $steps
$incy,=,($endy - $y) / $steps
$incr,=,$rend - $r / $steps
$incx,=,$xend - $x / $steps
$step,=,0
while $step,<,$steps
 lc255 255,0,$b
 lw $r / 50
 kl $x,$y,$r
 $r,+,$incr
 $b,+,$incb
 $x,+,$incx
 $y,+,$incy
 $step,+,1
endwhile
finalizePDF
```

l

Syntax: l x1,y1,x|y|grad,length

```
zeichnet senkrechte (y) oder waagrechte (x) oder "grad" gedrehte Linie von
x1/y1 in Länge "length"
wenn x = 'a' -> append mode

newPDF
newPage
$y,=,100
$yend,=,2900
$yinc,=,2
$x,=,100
$xend,=,1900
```

```
$xinc,=,2

lw 1
while $y,<,$yend
 l 100,$y,x,1800
 $y,+,$yinc
 $yinc,*,1.1
endwhile
while $x,<,$xend
 l $x,100,y,2800
 $x,+,$xinc
 $xinc,*,1.1
endwhile
$grad,=,0
while $grad,<,360
 l 1000,1400,$grad,600
 $grad,+,5
endwhile
finalizePDF
```

lc

Syntax: lc shade

```
setzt die Schwarz-Weiss Linienfarbe von 0 = Schwarz ... 100 = Weiss

newPDF
newPage
lc 80
lw 100
l 100,110,x,1900
finalizePDF
```

lc#

Syntax: lc# hexRGBColor

```
setzt die RGB Linienfarbe als Hexwert mit jew. 00...FF für R, G, B

newPDF
newPage
lc# FF00AA
lw 100
l 100,110,x,1900
finalizePDF
```

lc255

Syntax: lc255 r,g,b

```
setzt die RGB Linienfarbe als 3 Dezimalzahlen mit jew. 0...255 für R, G, B

newPDF
newPage
$b,=,0
$ly,=,200
```

```
$endy,=,2800
$steps,=,30
$incb,=,255 / $steps
$incy,=,($endy - $ly) / $steps
$step,=,0
lw $incy * 0.9
while $step,<,$steps
 lc255 255,0,$b
 l 100,$ly,x,1900
 $b,+,$incb
 $ly,+,$incy
 $step,+,1
endwhile
finalizePDF
```

lcrgb

Syntax: lcrgb r,g,b

```
setzt die RGB Linienfarbe als 3 Dezimalzahlen mit jew. 0...100 für R, G, B

newPDF
newPage
$b,=,0
$ly,=,200
$endy,=,2800
$steps,=,30
$incb,=,100 / $steps
$incy,=,($endy - $ly) / $steps
$step,=,0
lw $incy * 0.9
while $step,<,$steps
 lcrgb 100,0,$b
 l 100,$ly,x,1900
 $b,+,$incb
 $ly,+,$incy
 $step,+,1
endwhile
finalizePDF
```

ls

Syntax: ls mark,space

```
setzt den Linestyle für unterbrochene/gepunktete/gestrichelte Linie; mark =
Linie, space = Lücke;
mit lsn wieder zurück auf durchgezogene.

newPDF
newPage
$w,=,1
$mark,=,1
$space,=,2
$ly,=,200
$endy,=,2800
$steps,=,30
```

```
$incy,=,($endy - $ly) / $steps
$endw,=,$incy * 0.9
$incw,=,$endw - $w / $steps
$endmark,=,$endw * 2
$endspace,=,$endw * 3
$incmark,=,$endmark - $mark / $steps
$incspace,=,$endspace - $space / $steps
$step,=,0
while $step,<,$steps
 lw $w
 ls $mark,$space
 l 100,$ly,x,1900
 $w,+,$incw
 $mark,+,$incmark
 $space,+,$incspace
 $ly,+,$incy
 $step,+,1
endwhile
finalizePDF
```

lsn

Syntax: lsn

```
Linestyle normal (durchgezogen)

lsn
```

lvb

Syntax: lvb x1,y1,x2,y2

```
zeichnet Linie von x1/y1 bis x2/y2

newPDF
newPage
$x1,=,100
$x1b,=,300
$x2,=,500
$x2b,=,1900
$y1,=,100
$y1b,=,200
$y2,=,2900
$y2b,=,400

$steps,=,30
$incx1,=,$x1b - $x1 / $steps
$incx2,=,$x2b - $x2 / $steps
$incy1,=,$y1b - $y1 / $steps
$incy2,=,$y2b - $y2 / $steps
$step,=,0
while $step,<,$steps
 lvb $x1,$y1,$x2,$y2
 $x1,+,$incx1
 $x2,+,$incx2
```

```
 $y1,+,$incy1
 $y2,+,$incy2
 $step,+,1
endwhile
finalizePDF
```

lw

Syntax: lw width

```
setzt die Linienbreite

newPDF
newPage
$w,=,1
$ly,=,200
$endy,=,2800
$steps,=,30
$incy,=,($endy - $ly) / $steps
$endw,=,$incy * 0.9
$incw,=,$endw - $w / $steps
$step,=,0
lc 80
while $step,<,$steps
 lw $w
 l 100,$ly,x,1900
 $w,+,$incw
 $ly,+,$incy
 $step,+,1
endwhile
finalizePDF
```

newPage

Syntax: newPage

```
neue Seite DIN A4 hoch

newPDF
newPage
v 100,100,l,"Hello IntarS.
finalizePDF
```

newPageA5Quer

Syntax: newPageA5Quer

```
neue Seite mit DIN A5 quer

newPageA5Quer
```

newPageqm

Syntax: newPageqm

```
legt eine neue Seite mit 1 x 1 qm Größe an. Das ist ziemlich groß.

newPageqm
```

newPageQuer

Syntax: newPageQuer

```
neue Seite mit DIN A4 quer

newPageQuer
```

newPageWH

Syntax: newPageWH width,height[rotate]

```
neue Seite mit Breite "width", Höhe "height", optional gedreht um "rotate"
     Grad
(nur Vielfache von 90).
Achtung: rotate ist derzeit nicht wirksam. Wurde lediglich wg. eines Bugs im
Acrobat Reader implementiert, jetzt bedeutungslos.

newPageWH 800,400   // 8 x 4 cm
```

newPDF

Syntax: newPDF

```
erstellt ein neues PBPDF Objekt, das in der %session zuhause ist. Es kann
     über
%session.pbpdf erreicht werden. Alle folgenden PDF Befehle beziehen sich auf
dieses PDF Objekt. Es kann zu einem Zeitpunkt nur 1 PDF Objekt aktiv sein.
     Man
kann aber mehrere PDFs in einem Script hintereinander erzeugen.
Hat man ein Script mit dem Filesuffix .cpdf, wird bereits ein PDF vom
     Framework
 vorbereitet, in dem man gleich loslegen kann, ohne vorher newPDF zu machen.
Das ist aber veraltet, weil man zuwenig Kontrolle hat. Meist muss erst
     geprüft
werden, ob ein PDF erstellt werden soll.

newPDF
newPage
v 100,100,l,'Hello Intars.
finalizePDF
```

o

Syntax: o x,y

```
offset; verschiebt alle Zeichenoperationen um "x" und "y" Maßeinheiten
     relativ
zum Nullpunkt.
```

```
o 10,0
```

pages

Syntax: pages n

```
legt max. Anzahl von Seiten im PDF fest. Dient als Sicherung gegen
Endlosschleifen.

pages 1000
```

pdfLines

Syntax: target,pdfLines,string,width

```
liefert ein Array von Strings, die an Wortgrenzen aus "string" getrennt sind,
derart, dass die Strings im aktuellen Font des PDFs gerendert nicht breiter
     als
"width" sind.

$string,=,"liefert ein Array von Strings, die an Wortgrenzen aus "string"
     getrennt sind, derart, dass die Strings im aktuellen Font des PDFs
     gerendert nicht breiter als "width" sind.
$width,=,500
$ty,=,200
$tx,=,200
newPDF
newPage
$a,pdfLines,$string,$width
foreach line,$a
   v $tx,$ty,l,$line
   $ty,+,50
endfor
finalizePDF
```

popf

Syntax: popf

```
Holt den Font samt Größe vom Stack und setzt ihn als momentanen Font. Dient
dazu etwas in einem anderen Font zu drucken und danach mit popf auf den
ursprünglichen Font zurückzugehen.

newPDF
newPage
f Helvetica,12
v 100,100,l,'Hello World.
pushf
f Helvetica-Bold,6
v 100,150,l,'Hello World.
popf
v 100,200,l,'Hello World.
finalizePDF
```

previewPDF

Syntax: previewPDF

```
zeigt das PDF in einem neuen Fenster

previewPDF
```

pushf

Syntax: pushf

```
Speichert den momentanen Font samt Größe auf einem Stack. Dient dazu etwas in
einem anderen Font zu drucken und danach mit popf auf den ursprünglichen Font
zurückzugehen.

newPDF
newPage
f Helvetica,12
v 100,100,l,'Hello World.
pushf
f Helvetica-Bold,6
v 100,150,l,'Hello World.
popf
v 100,200,l,'Hello World.
finalizePDF
```

render

Syntax: render

```
rendert das in %session.pbpdf aktuelle PDF Objekt vorab in ein File. Der
Filename ist ein global unique Zufallsname und steht im %session.pbpdf drin.
Danach kann das File z.B. archiviert oder wo anders hinkopiert werden. Will
     man
aber nicht selber etwas mit dem File machen, muss kein "render" ausgeführt
werden.

render
```

rf

Syntax: rf x,y,w,h

```
zeichnet ein gefülltes Rechteck an x,y mit Breite w, Höhe h mit geltender
Fill-Farbe.

newPDF
newPage
$b,=,0
$x,=,100
$w,=,50
$h,=,30
$wend,=,500
$hend,=,1200
$xend,=,1900 - $wend
$y,=,200
```

```
$endy,=,2800
$steps,=,30
$incb,=,255 / $steps
$incy,=,($endy - $y) / $steps
$incw,=,$wend - $w / $steps
$inch,=,$hend - $h / $steps
$incx,=,$xend - $x / $steps
$step,=,0
while $step,<,$steps
 fc 100
 rf $x - 5,$y - 5,$w + 10,$h + 10
 fc255 255,0,$b
 rf $x,$y,$w,$h
 $w,+,$incw
 $h,+,$inch
 $b,+,$incb
 $x,+,$incx
 $y,+,$incy
 $step,+,1
endwhile
finalizePDF
```

rl

Syntax: rl x,y,w,h

```
zeichnet ein leeres Rechteck an x,y mit Breite w, Höhe h mit geltenden
Linien-Einstellungen.

newPDF
newPage
$b,=,0
$x,=,100
$w,=,50
$h,=,30
$wend,=,500
$hend,=,200
$xend,=,1900 - $wend
$y,=,200
$endy,=,2800
$steps,=,30
$incb,=,255 / $steps
$incy,=,($endy - $y) / $steps
$incw,=,$wend - $w / $steps
$inch,=,$hend - $h / $steps
$incx,=,$xend - $x / $steps
$step,=,0
while $step,<,$steps
 lc255 255,0,$b
 lw $h / 20
 rl $x,$y,$w,$h
 $w,+,$incw
 $h,+,$inch
 $b,+,$incb
 $x,+,$incx
```

```
 $y,+,$incy
 $step,+,1
endwhile
finalizePDF
```

s

Syntax: s size

```
setzt Fontsize für folgende Zeichenbefehle

s 10
```

scale

Syntax: scale faktor

```
Legt Skalierungsfaktor fest. Default ist 10. Dann ist eine Einheit 1/10 mm.

scale 10
```

spl

Syntax: spl

```
select Page Last;
wählt die bislang letzte Seite im Dokument aus als Ziel folgender
Zeichenkommandos.

newPDF
$seite,=,0
$ende,=,N
while $ende,eq,N
 newPage
 $seite,+,1
 v 200,100,r,Seite,<int>$seite
 $j,singleValueSQL,"select rand()
 if $j,<,0.1
  $ende,=,J
 endif
endwhile

$i,=,0
while $i,<,$seite
 spx $i
 v 210,100,l,von,<int>$seite
 $i,+,1
endwhile
spl
v 210,400,l,'this is the End.
finalizePDF
```

spx

Syntax: spx n

```
select Page x; Seite Nr. "n" im Dokument als Ziel folgender Zeichenkommandos
auswählen. Z.B. um im nachhinein "Seite Nr. xxx von yyy" draufzuzeichnen,
     wenn
man weiss, wieviele yyy Seiten es geworden sind.

newPDF
$seite,=,0
$ende,=,N
while $ende,eq,N
 newPage
 $seite,+,1
 v 200,100,r,Seite,<int>$seite
 $j,singleValueSQL,"select rand()
 if $j,<,0.1
  $ende,=,J
 endif
endwhile

$i,=,0
while $i,<,$seite
 spx $i
 v 210,100,l,von,<int>$seite
 $i,+,1
endwhile
finalizePDF
```

ub

Syntax: ub

```
underline begin

newPDF
newPage
ub
v 100,100,l,'Hello Intars.
v a,100,l,' Hello World.
ue
finalizePDF
```

ue

Syntax: ue

```
underline end

newPDF
newPage
ub
v 100,100,l,'Hello Intars.
v a,100,l,' Hello World.
```

```
ue
finalizePDF
```

v

Syntax: v x,y,o,string[,string1...n]

```
value;
Werte "string" (1...n mit einem Space angehängt) an Position "x"/"y" mit
orientation "o" drucken.
"o" = {l[xxx],r[xxx],c[xxx]} -> Links-/Rechtsbündig/Zentriert;
xxx = um xxx Grad gedreht
wenn "x" = 'a' -> append mode

newPDF
newPage
$s,=,"Hello IntarS.
$x,=,1000
$y,=,200
v $x,$y,l,$s
$y,+,50
v $x,$y,r,$s
$y,+,50
v $x,$y,c,$s
$y,+,150
v $x,$y,r45,$s
v $x,$y,l45,$s
v $x,$y,l315,'   ,$s
finalizePDF
```

vc

Syntax: vc x,y,o,string[,string1...n]

```
value compact;
Werte "string" (1...n ohne Space dazwischen) an Position "x"/"y" mit
orientation "o" drucken.
"o" = {l[xxx],r[xxx],c[xxx]} -> Links-/Rechtsbündig/Zentriert;
xxx = um xxx Grad gedreht
wenn "x" = 'a' -> append mode

newPDF
newPage
$s,=,"Hello IntarS.
$x,=,1000
$y,=,200
vc $x,$y,l,$s,$s
$y,+,50
vc $x,$y,r,$s,$s
$y,+,50
vc $x,$y,c,$s,$s
finalizePDF
```

vf

Syntax: vf x,y,o,w,string[,string1...n]

```
value fix width;
Werte "string" (1...n mit einem Space angehängt) an Position "x"/"y" mit
orientation "o" innerhalb der festen Breite "w" drucken.
"o" = {l[xxx],r[xxx],c[xxx]} -> Links-/Rechtsbündig/Zentriert;
xxx = um xxx Grad gedreht
wenn "x" = 'a' -> append mode.
Die x-Position, an der mittels append mode nach einem vf-Befehl gedruckt
     wird,
ist um die fixe Breite weitergeführt, unabhängig von der Stringlänge.
Das ist sehr geschickt für Tabellenspalten.

newPDF
newPage
$s,=,"Hello IntarS.
$x,=,1000
$y,=,200
$w,=,500
vf $x,$y,l,$w,$s
$y,+,50
vf $x,$y,r,$w,$s
$y,+,50
vf $x,$y,c,$w,$s
$y,+,150
$w,=,300
vf $x,$y,l,$w,$s
$y,+,50
vf $x,$y,r,$w,$s
$y,+,50
vf $x,$y,c,$w,$s

finalizePDF
```

vw VERALTET !!!

Syntax: vw x,y,o,w,string[,string1...n]

```
value width-max;
Werte "string" (1...n mit einem Space angehängt) an Position "x"/"y" mit
orientation "o" mit maximal Breite "w" drucken.
"o" = {l[xxx],r[xxx],c[xxx]} -> Links-/Rechtsbündig/Zentriert;
xxx = um xxx Grad gedreht
wenn "x" = 'a' -> append mode.
Meistens will man vf.

newPDF
newPage
$s,=,"Hello IntarS.
$x,=,1000
$y,=,200
$w,=,500
```

```
vw $x,$y,l,$w,$s
$y,+,50
vw $x,$y,r,$w,$s
$y,+,50
vw $x,$y,c,$w,$s
$y,+,150
$w,=,300
vw $x,$y,l,$w,$s
$y,+,50
vw $x,$y,r,$w,$s
$y,+,50
vw $x,$y,c,$w,$s

finalizePDF
```

12.9 File

attributes VERALTET !!!

Syntax: target,attributes,path

```
liefert ein Dictionary mit den File-Attributes des Files an "path"; geht
mittlerweile geschickter mit:
$d,=,$path.attributesAtPath

das Dictionary enthält:
NSFileModificationDate
NSFileSize
NSFileType -> NSFileTypeDirectory,NSFileTypeRegular
- (NSDate *)fileModificationDate;
- (unsigned long long)fileSize;

$d,attributes,"/tmp/bla.txt
```

copyPath

Syntax: copyPath fromPath,toPath

```
kopiert rekursiv alles, was sich an "fromPath" befindet nach "toPath"

copyPath '/tmp/my/long/path,'/tmp/my_other_path
```

createPath

Syntax: createPath path

```
legt alle Directories entlang des "path" an, die noch nicht existieren.

createPath '/tmp/my/long/path/that/leads/to/secret/files
```

deletePath

Syntax: deletePath path

```
löscht rekursiv alles, was sich an "path" befindet

deletePath '/tmp/my/long/path
```

movePath

Syntax: movePath fromPath,toPath

```
verschiebt  "fromPath" nach "toPath"

movePath '/tmp/my/long/path,'/tmp/my_other_path
```

newFile

Syntax: target,newFile,path

```
erzeugt ein neues File und liefert die offene Filehandle
fileHandleForUpdatingAtPath dazu; die directories in "path" müssen bereits
vorhanden sein

$fh,newFile,'/tmp/blafasel.txt
```

swcof

Syntax: target,swcof,path

```
string with content of File; liefert einen NSString mit dem Inhalt des Files
     an
"path". Encoding wird automatisch aufgrund der vorgefundenen Daten erraten.

$file,swcof,'/tmp/bla.txt
```

wtf

Syntax: wtf string,path[,encoding]

```
schreibt den "string" als utf-8 (wenn nicht explizit ein anderes "encoding"
angegegeben ist) in das File an "path". Ein bestehendes wird überschrieben.
"encoding" kann sein: CP1252, ISOLatin1, ISOLatin2, Unicode, ASCII

wtf $s,'/tmp/bla.txt
```

12.10 Flowcontrol

assert_continuefor

Syntax: assert_continuefor exp1,vgl1,exp2[,exp3,vgl2,exp4,...]

```
stellt sicher, dass alle aufgeführten Bedingungen erfüllt sind und macht
andernfalls continuefor. Die 1 bis 3 Vergleiche werden AND-verknüpft.

foreach s,$a
    assert_continuefor $s,FILLED,
    logi $s
endfor
```

assert_end

Syntax: assert_end exp1,vgl1,exp2[,exp3,vgl2,exp4,...]

```
stellt sicher, dass alle aufgeführten Bedingungen erfüllt sind und beendet
andernfalls die Ausführung. Die 1 bis 3 Vergleiche werden AND-verknüpft.

assert_end $p_eo,FILLED,
```

assert_mf_end

Syntax: assert_mf_end msg,fieldName,exp1,vgl1,exp2[,exp3,vgl2,exp4,...]

```
stellt sicher, dass alle aufgeführten Bedingungen erfüllt sind und beendet
andernfalls die Ausführung mit Meldung "msg" und markiert das Feld
     "fieldName"
als Fehler. Die 1 bis 3 Vergleiche werden AND-verknüpft.

assert_mf_end 'bitte Artikelnummer
     eingeben,p_artikelnum,$p_artikelnum,FILLEDNUM,
```

assert_m_continuefor

Syntax: assert_m_continuefor msg,exp1,vgl1,exp2[,exp3,vgl2,exp4,...]

```
stellt sicher, dass alle aufgeführten Bedingungen erfüllt sind und macht
andernfalls continuefor und gibt die Meldung msg im Log aus. Die 1 bis 3
Vergleiche werden AND-verknüpft.

$a,newArrayE,a,,b,c
foreach s,$a
    assert_m_continuefor 'nicht gefuellt!,$s,FILLED,
    logi $s
endfor
```

assert_m_end

Syntax: assert_m_end msg,exp1,vgl1,exp2[,exp3,vgl2,exp4,...]

```
stellt sicher, dass alle aufgeführten Bedingungen erfüllt sind und beendet
andernfalls die Ausführung mit Meldung "msg". Die 1 bis 3 Vergleiche werden
AND-verknüpft.
```

```
assert_m_end 'bitte genau einen Satz auswählen,$p_eo,FILLED,
```

break

Syntax: break

```
beendet eine while-Schleife vorzeitig.

$i,=,0
$j,=,$a.count
while $i,<,$j
 $s,oai,$a,$i
 $i,+,1
 if $s,eq,'ende
    break
 endif
 logi $s
endfor
```

breakfor

Syntax: breakfor

```
beendet eine foreach-Schleife vorzeitig.

foreach s,$a
 if $s,eq,'ende
    breakfor
 endif
 logi $s
endfor
```

continue

Syntax: continue

```
leitet innerhalb einer while-Schleife vorzeitig einen neuen Durchlauf ein.

$i,=,0
$j,=,$a.count
while $i,<,$j
 $s,oai,$a,$i
 $i,+,1
 if $s,!FILLED,
    continue
 endif
 logi $s
endfor
```

continuefor

Syntax: continuefor

```
leitet innerhalb einer foreach-Schleife vorzeitig einen neuen Durchlauf ein.

foreach s,$a
 if $s,!FILLED,
```

```
        continuefor
     endif
     logi $s
    endfor
```

diaConfirm

Syntax: diaConfirm [msg,cancel,confirm]

```
Bringt eine modale Box mit der Meldung "msg" an die Oberfläche mit 2 Buttons
für Nein/Zurück/Abbruch und Ja/Weiter; wird keine "msg" übergeben, wird "Sind
Sie sicher?" gefragt. Funktioniert nur in firstLevel Scripts.

diaConfirm 'Löschen?,Nein,Ja
assert_m_end 'Abbruch,$_rv,eq,J
```

else

Syntax: else

```
definiert in Zusammenhang mit if else endif den Zweig der Verarbeitung, der
ausgeführt werden soll, wenn die if-Bedingung nicht zutrifft. Wie eben in
     jeder
Programmiersprache.

if $a,>,$b
   logi $a,'ist größer
else
   logi $a,'ist nicht größer
endif
```

end

Syntax: end

```
beendet den Programmfluss.

end
```

endfor

Syntax: endfor

```
beendet eine foreach-Schleife.

foreach s,$a,i
   logi $i:,$s
endfor
```

endif

Syntax: endif

```
beendet den bei if begonnenen bedingten Block der Ausführung

if $a,==,$b
    logi "a und b sind gleich.
```

```
endif
```

endsub

Syntax: endsub

```
Beendet eine Subroutine. Gibt die Subroutine einen Wert zurück, sollte dieser
gemäß Namenskonvention so benannt werden wie die Subroutine.

sub befehlVgl
 $befehle,css,$l_p1,<comma>
 $l_lma,newArray,10
 foreach befehl,$befehle
  //mehrere Schreibweisen
  $l_s,spf,'[op isCaseInsensitiveEqualToString:@"%@"],$befehl
  $l_lma,a,$l_s
 endfor
 $befehlVgl,cjs,$l_lma,' ||
endsub
```

endwhile

Syntax: endwhile

```
markiert das Ende einer while-Schleife.

$i,=,0
while $i,<,100
   $i,+,1
   logi <int>$i
endwhile
```

execute

Syntax: execute scriptName[,param1...n]

```
voreparstes Script namens "scriptName" ausfuehren mit derselben datasource u.
parmdict; varDict des gerufenen Scripts kommt in $_rv zururück, $scriptRC
     wird
direkt zurückgegeben

execute vid_lager/etikett,$p_eo
```

executeString

Syntax: executeString scriptName,scriptContent[,param1...n]

```
Script, das als Source in "scriptContent" vorliegt ausfuehren mit derselben
datasource u. parmdict unter dem Namen "scriptName"; imports werden
     aufgeloest;
varDict des gerufenen Scripts kommt in $_rv zururück, $scriptRC wird direkt
zurückgegeben

executeString tempScript,$s
```

execute_ds

Syntax: execute_ds datasource,scriptName[,param1...n]

```
voreparstes Script namens "scriptName" ausfuehren unter Kontrolle der
"datasource" und mit deren parmDict; varDict des gerufenen Scripts kommt in
$_rv zururück, $scriptRC wird direkt zurückgegeben

execute_ds $e,vid_lager/etikett,$p_eo
```

execute_inl

Syntax: execute_inl scriptName

```
voreparstes Script namens "scriptName" ausfuehren unter Kontrolle der
     gleichen
%datasource und mit gleichem parmDict; zusätzlich wird das gesamte varDict
     des
rufenden Scripts übergeben und danach wird das varDict des inline Scripts in
das des rufenden Scripts zurückkopiert. Entspricht somit der Wirkungsweise
eines import, nur dass das inline Script vorgparst ist.

execute_inl fusstext-inl
```

foreach

Syntax: foreach loopVarName,array[,loopIndexName]

```
beginnt eine Schleife über alle Objekte in "array". Das jeweilige Objekt wird
in "loopVarName" zur Verfügung gestellt. Unter dem "loopIndexName" werden die
Schleifendurchläufe hochgezählt.

foreach s,$a,i
   logi $i:,$s
endfor
```

gosub

Syntax: gosub subName[,parm1...n]

```
ruft die Subroutine namens "subName" auf und übergibt Parameter, die dort als
$l_p1, $l_p2, ... ankommen. Defaultmäßig haben Subroutinen denselben globalen
Variablenraum. Will man lokale Variablen, deren Gültigkeit explizit auf die
Subroutine beschränkt sind, müssen sie mit vorangestelltem "l_" benannt
     werden,
z.B. $l_a, $l_s.
"subName" muss konstant sein.

gosub befehlAnzOperands,$eo.min_anz_parm,$eo.max_anz_parm
```

if

Syntax: if exp1,vgl1,exp2[,exp3,vgl2,exp4,...]

```
beginnt eine bedingte Ausführung.
```

```
$a,=,1
$b,=,2
$c,=,haha
if $a,==,$b,$c,FILLED,
    logi "a und b sind gleich oder c ist gefüllt
endif
end

Als "vgl1" ..."vgl3" stehen zur Verfügung:
op = {eq,ne,==,=,>,<,>=,<=,FILLED,FILLEDNUM,
   exists,contains,like,prefix,suffix,in,d>,d>=,d<,d<=}
!
 negiert den Vergleich
eq,ne
 vergleichen Strings
==,=,>,<,>=,<=
 vergleichen numerisch
d>,d>=,d<,d<=
 vergleichen Datum
FILLED
 testet jedes Objekt, egal welcher Klasse ob gefüllt;
 op2 ist ohne Funktion, das Komma muss aber sein
FILLEDNUM
 wie FILLED, testet zusätzlich auf ungleich "0" und ungleich "00000000000000";
 op2 ist ohne Funktion, das Komma muss aber sein
exists
existsPath
 testet, ob es ein File namens op1 gibt
contains
 testet, ob op1 (NSString/NSDictionary/NSArray/NSSet)
 einen der Operanden enthält; ist ein Operand ein Array, wird sein Inhalt
 getestet;
like
 macht einen Vergleich mit WildCards: "*" und "?"
prefix,suffix
 testen, ob op1 den pre-/suffix op2 hat
in
 prüft, ob op1 mit op2 oder einem der folgenden
 Operanden übereinstimmt; op2 und folgende dürfen
 NSArrays sein; es wird der Vergleich mit jedem
 Element der Arrays durchgeführt;
```

import VERALTET !!!

Syntax: import fileName

```
importiert zur Ladezeit Statements aus "fileName" analog #import des
C-Preprozessors. Das import-Statement selbst bleibt als NOP stehen für XRef.
Sollte nicht mehr verwendet werden, führt zu Problemen, weil keine Isolation
zwischen dem importierenden und importierten Code besteht.

import _pos_common/_pos_didValToTarget_li
```

return

Syntax: return

```
springt aus der Subroutine zurück. Passiert implizit auch bei Erreichen von
endsub.

return
```

sub

Syntax: sub subName

```
Definiert eine Subroutine. Gibt die Subroutine einen Wert zurück, sollte
     dieser
gemäß Namenskonvention so benannt werden wie die Subroutine. Aufgerufen wird
sie mit gosub

sub befehlVgl
 $befehle,css,$l_p1,<comma>
 $l_lma,newArray,10
 foreach befehl,$befehle
  //mehrere Schreibweisen
  $l_s,spf,'[op isCaseInsensitiveEqualToString:@"%@"],$befehl
  $l_lma,a,$l_s
 endfor
 $befehlVgl,cjs,$l_lma,' ||
endsub
```

while

Syntax: while exp1,vgl1,exp2[,exp3,vgl2,exp4,...]

```
beginnt eine while-Schleife. Die 1 bis 3 Vergleiche werden OR-verknüpft.

$i,=,0
while $i,<,100
   $i,+,1
   logi <int>$i
endwhile
```

12.11 Foundation

*****

Syntax: target,*,zahl

```
multipliziert target mit "zahl"

$i,*,2
```

+

Syntax: target,+,zahl

```
addiert auf target "zahl" drauf

$i,+,1
```

-

Syntax: target,-,zahl

```
zieht von target "zahl" ab

$i,-,1
```

/

Syntax: target,/,zahl

```
dividiert target durch "zahl"

$i,/,2
```

=

Syntax: target,=,value

```
$a,=,b
```

a

Syntax: target,a,value[,values...]

```
add, append; polymorph; je nach target wird angefügt oder in eine Collection
aufgenommen;

$lma,a,bla,fasel,x
```

ah

Syntax: target,ah,hexValue[,...]

```
append hex to NSMutableData; einzelne Bytes

$mdata,ah,CA,FF,EE
```

allk VERALTET !!!

Syntax: target,allk,dictionary

```
all keys; liefert alle Keys des "dictionary".
Geht mittlerweile einfach durch keypath Adressierung:
$a,=,$d.allKeys

$a,allk,$d
```

allv VERALTET !!!

Syntax: target,allv,dicitionary

```
liefert das array aller Werte des "dictionary".
Geht mittlerweile geschickter mit keypath Adressierung:
$a,=,$d.allValues

$a,allv,$d
```

au

Syntax: target,au,value[,valuex,...]

```
add uniq; target muss ein Mutable Array sein; value wird nur aufgenommen,
     wenn
nicht schon enthalten;

$lma,au,$s
```

autorelease

Syntax: autorelease

```
gibt Speicher frei, indem nicht mehr benötigte autoreleaste Objekte
     deallokiert
werden. Wird zwar vom Framework automatisch spätestens nach Ausführung von
     3000
Statements gemacht. Bei Erzeugung von vielen Objekten in einer Schleife mit
     nur
wenigen Statements kann es aber sinnvoll sein, zwischendurch zusätzlich
aufzuräumen.

autorelease
```

class

Syntax: target,class,className

```
liefert das Klassen-Objekt namens "className"

$NSString,class,NSString
```

count VERALTET !!!

Syntax: target,count,collection

```
liefert die Anzahl von Objekten der Collection "collection". Geht
     mittlerweile
einfacher: $i,=,$a.count

$i,count,$a
```

ff

Syntax: target,ff,o1[,o2,...,on]

```
first filled; liefert das erste gefüllte der aufgezählten Objekte; ist ein
aufgezähltes Objekt ein Array, wird sein Inhalt genommen;

$s,ff,$a,$s1
```

invk

Syntax: target,invk,receiver,message[,parm1,...parmn]

```
schickt die "message" an "reveiver" und liefert das Ergebnis des
Methodenaufrufes

$pbat,invk,$t,plainAttrNamed:,$s
```

invk,ps

Syntax: invk target,methodName[,parm1...n]

```
ruft an "target" die Objective-C Methode "methodName" auf und übergibt ggfs.
Parameter.

invk $po,clearCachedHtmlString
```

ioai

Syntax: ioai mutableArray,index,Object

```
insert Object at Index; fügt "Object" an Stelle "index" in das "mutableArray"
ein

$ma,newArrayE,a,b,c
logi $ma
ioai $ma,1,x
logi $ma
end
```

lang,sprache

Syntax: lang n

```
setzt die Fremdsprache

lang $kopf.lang
```

lock

Syntax: target,lock,lockName

```
acquire lock; schreibt Directory in Filesystem, ohne wait;

$lockName,concat,'rechnung_buchen_,$p_eo.nummer
$jn,lock,$lockName
if $jn,eq,J
   ...
   unlock $lockName
else
   logi 'wird bereits gebucht
endif
```

lookup

Syntax: target,lookup,lookupName,rowName,colName

```
schlägt einen Wert in einer mittels lookupBegin / lookupEnd definierten
     Tabelle
mit benamten Zeilen und Spalten nach

$_title,lookup,title_pos,$__form,title
```

lookupBegin

Syntax: lookupBegin lookupName

```
beginnt die Definition einer Lookup-Tabelle namens "lookupName";
s. "lookup"

lookupBegin label_gui
lookupFS |
x|gui
A|A > 15
B|B < 15
C|C < 5
lookupEnd
```

m

Syntax: target,m,zahl

```
liefert den Rest der Modulodivision von "target" und "zahl"
```

```
$i,m,10
```

matrixBegin

Syntax: matrixBegin matrixName

```
beginnt die Definition einer zweidimensionalen Matrix namens "matrixName" mit
konstanten Werten.

matrixBegin columndefs
lookupFS |
70|l|Best|nummer|goto||zur Bestellung|sortable
70|l|Nummer|kundennumm|goto_lief||zum Lieferanten|sortable
matrixEnd
```

ml

Syntax: target,ml,...

```
multi line; array of  strings ab naechster Zeile bis \end

$a,ml,
bla
fasel
\end
```

newArray

Syntax: target,newArray,capacity

```
erzeugt ein neues, leeres Mutable Array der initialen Kapazität "capacity"

$lma,newArray,10
```

newArrayE

Syntax: target,newArrayE,object1[,objectn...]

```
new Array with Entries
erzeugt ein neues Mutable Array mit den aufgezählten Objekten; ist ein
aufgezähltes Objekte selbst ein Array, wird dessen Inhalt genommen;

$a,newArrayE,bla,fasel
```

newData

Syntax: target,newData,capacity

```
erzeugt ein neues, leeres Mutable Data Objekt der anfänglichen Kapazität
"capacity"

$md,newData,1024
```

newDict

Syntax: target,newDict,capacity

```
erzeugt ein leeres Mutable Dictionary (auch assoziatives Array genannt) mit
     der
initialen Kapazität "capacity"

$md,newDict,10
```

newSet

Syntax: target,newSet,capacity

```
erzeugt ein neues leeres Mutable Set (eine unordered collection) der
     initialen
Kapazität "capacity". Ein Anwendungsfall ist bislang nicht bekannt.

$mset,newSet,10
```

newString

Syntax: target,newString,capacity

```
erzeugt einen leeren Mutable String

$ms,newString,1000
```

nls

Syntax: target,nls,origString,nlsString[,lang]

```
liefert gemäß "lang" oder wenn nicht angegeben Session-lang den origString
     oder
den aus nlsString gewonnenen übersetzten String;
nlsString kann durch "|" getrennt die Übersetzungen aufzählen;

$s,nls,$as.bezeichnun,$as.bezeichnun2
```

oai

Syntax: target,oai,array,index

```
liefert das Object at "index" aus "array"; wirft keine out of range Exception
sondern liefert stattdessen nil;
kann mittlerweile auch mit keypath-Adressierung gemacht werden:
$s,=,$a.2

$s,oai,$a,2
```

ofk VERALTET !!!

Syntax: target,ofk,dictionary,key

```
liefert aus "dictionary" das unter "key" gespeicherte Objekt; geht
     mittlerweile
einfacher mit keypath-Adressierung:
```

```
$s,=,$d.$key

$s,ofk,$d,$key
```

rao

Syntax: rao mutableCollection

```
remove all Objects; entfernt alle Objekte aus einer mutable Collection
(NSMutableArray, NSMutableDictionary)

rao $md
```

roai

Syntax: roai mutableArray,index

```
remove Object at Index; entfernt aus dem "mutableArray" das Objekt an Stelle
"index"

$ma,newArrayE,a,b,c
roai $ma,1
-> a,c
```

rofk

Syntax: rofk mutableCollection,key

```
remove Object for "key"; entfernt aus einem NSMutableDictionary das Objekte,
das unter "key" gespeichert ist;


rofk $md,bla
```

sort

Syntax: sort mutableArray[,keyOrderArray]

```
sortiert das "mutableArray" gemäß dem angegebenen "keyOrderArray". Ist keines
angegeben, werden die Objekte im "mutableArray" mit compare: verglichen.

$sortOrder,=,"bestand:d,artikelnum
sort $ma,$sortOrder.soa
```

sorted

Syntax: a,sorted,Array[,soaString]

```
sortiert das "Array" gemäß dem als String notierten "keyOrderArray". Ist
     keines
angegeben, werden die Objekte im "Array" mit compare: verglichen.

$a,sorted,$a,"bestand:d,artikelnum
```

system

Syntax: system cmd

```
es wird mittels des system() Bestriebssystemaufrufes ein neuer Prozess
     geforked
und darin das "cmd" an eine Shell (bash) zur Ausführung übergeben.
Der Exit-Code wird in $_rv zurückgeliefert.

system "svn update
```

tableNamed

Syntax: target,tableNamed,tableName

```
liefert die PBDDTable aus dem Repository namens "tableName"

$t,tableNamed,vid_lager
```

unlock

Syntax: unlock lockName

```
released den lock "lockName";
pendant zu target,lock,lockName

$lockName,concat,'rechnung_buchen_,$p_eo.nummer
$jn,lock,$lockName
if $jn,eq,J
   ...
   unlock $lockName
else
   logi 'wird bereits gebucht
endif
```

12.12 Modulsteuerung

abbrechen

Syntax: abbrechen [modul]

```
entspricht dem Click auf den Abbrechen-Button. Nicht gesicherte Editierungen
werden zurückgenommen in "modul" bzw. %datasource.

abbrechen $e
```

aktualisieren

Syntax: aktualisieren [modul]

```
entspricht dem Click auf den Akualisieren-Button. Es wird das selObj neu aus
der Datebank versorgt.

aktualisieren $e
```

allFieldsOff

Syntax: allFieldsOff

```
schaltet alle Felder/Buttons aus


allFieldsOff
```

allFieldsOn

Syntax: allFieldsOn

```
schaltet alle Felder/Buttons ein, wenn dem jeweils nichts anderes
     entgegensteht.

allFieldsOn
```

allFieldsVisible

Syntax: allFieldsVisible

```
schaltet alle Felder an der Oberfläche auf sichtbar.

allFieldsVisible
```

aValueChanged

Syntax: aValueChanged [modul]

```
signalisiert dem "modul" bzw. wenn nichts angegeben ist, der %datasource,
     dass
sich ein Wert im selObj geändert hat. Damit geht das needsSave Flag auf YES
     und
pendingAction auf "pa_upd", d.h. das "modul" befindet sich in einem Zustand
     als
wäre ein Wert über die Oberfläche geändert worden.
```

```
aValueChanged $e
```

createButton

Syntax: createButton on|off

```
schaltet den createButton an/aus

createButton off
```

deleteButton

Syntax: deleteButton on|off

```
schaltet den deleteButton an/aus

deleteButton off
```

didSetValueForKey

Syntax: didSetValueForKey key[,modul,eo]

```
dem "modul" bzw. der %datasource mitteilen, dass sich am "eo" bzw. selObj der
Wert namens "key" geändert hat. Löst dann den didValToTarget Event aus.

didSetValueForKey datum
didSetValueForKey datum,$e
didSetValueForKey datum,$e,$eo
```

dup

Syntax: dup [modul]

```
entspricht dem Click auf den Dup-Button. Es wird ein neues EO im "modul" bzw.
%datasource angelegt, als Kopie des markierten EO. didCreate, didDuplicate
     und
selObjChanged Events werden ausgelöst.

dup $e
```

duplicateButton

Syntax: duplicateButton on|off

```
schaltet den duplicateButton an/aus

duplicateButton off
```

empty

Syntax: empty [modul]

```
leert in "modul" bzw. %datasource das %ma und die selectedObjects;

empty $e
```

fieldInvisible

Syntax: fieldInvisible fieldName[,fieldName2,...]

```
schaltet Felder an Oberfläche unsichtbar; Wildcards und Arrays sind zulässig;

fieldInvisible lagerbest*,kdauftrbestges
```

fieldoff

Syntax: fieldoff fieldName[,...]

```
schaltet Felder und Buttons inaktiv; Feldnamen können mit Wildard '*'
     abgekürzt
angegeben werden; statt eines Feldnamens kann auch ein Array angegeben
     werden,
das Feldnamen enthält.

fieldoff artikelnum,bezeichnun,lager*,bu_*
```

fieldon

Syntax: fieldon fieldName[,...]

```
schaltet Felder eingabefähig und Buttons aktiv, soweit nichts anderes
entgegensteht. Feldnamen können mit Wildard '*' abgekürzt angegeben werden;
statt eines Feldnamens kann auch ein Array angegeben werden, das Feldnamen
enthält.

fieldon artikelnum,bezeichnun,lager*
```

fieldVisible

Syntax: fieldVisiblefieldName[,fieldName2,...]

```
schaltet Felder an Oberfläche sichtbar; Wildcards und Arrays sind zulässig;

fieldVisible lagerbest*,kdauftrbestges
```

focus

Syntax: focus feldName[,modul]

```
setzt den Cursor in das Feld namens "feldName" in "modul" bzw. %datasource

focus artikelnum,$e
```

loeschen

Syntax: loeschen [modul]

```
entspricht dem Click auf den Löschen-Button. Es wird das selObj im "modul"
     bzw.
%datasource gelöscht, falls der Löschen-Button nicht disabled ist. willDelete
und didDelete Events werden ausgelöst.

loeschen $e
```

markError

Syntax: markError fieldName[,fieldName2...n]

```
markiert ein Feld als Fehler und stellt den Cursor hinein; keine Wildcards,
keine Arrays;

markError artikelnum,bezeichnun
```

modul

Syntax: target,modul,modulName

```
liefert das Modul namens "modulName" bzw. nil, wenn es das nicht gibt oder
Berechtigung fehlt; allermeistens wird man aber modulOrEnd nehmen wollen.

$e,modul,vid_lager
```

modulOrEnd

Syntax: target,modulOrEnd,modulName

```
liefert das Modul namens "modulName" oder beendet das Script, falls es das
Modul nicht gibt oder keine Berechtigung besteht.

target,modulOrEnd,modulName
```

neu

Syntax: neu [modul]

```
entspricht dem Click auf den Neu-Button. Es wird ein neues EO im "modul" bzw.
%datasource angelegt, falls der Neu-Button nicht disabled ist. didCreate und
selObjChanged Events werden ausgelöst.

neu $e
```

neuSuchen

Syntax: neuSuchen [modul]

```
veranlasst "modul" bzw. %datasource mit bestehendem combinedQ ab aktuellem
offset neu zu suchen. Gibt es noch gar keinen combinedQ wird stattdessen
evaluateQualifiers aufgerufen. Versorgt das %ma neu, löst didTakeObjects
     Event
aus, refreshed die EOs in selectedObjects mit neuen Werten aus der Datenbank.
Ist %selectedOjbects dagegen leeer, wird das erste EO im %ma selektiert. Der
letzte Eintrag im stateArray (Suchvorgangs-Historie) wird aktualisiert.

neuSuchen $e
```

oeffneUebersicht

Syntax: oeffneUebersicht ueName,tableName[,modul]

```
öffnet für Tabelle "tableName" das PBWOUebersicht Modul unter dem Name
     "ueName"
für "modul" bzw. %datasource. Löst den Event getCMQForUebersicht aus und
```

```
blendet die Uebersicht auf.

oeffneUebersicht ms_artikel,vid_lager,$e
```

optionadd

Syntax: optionadd vlName,wert,bez[,bez1]

```
fügt einem Auswahllistenfeld "vlName" einen Eintrag hinzu; "wert" ist der
interne Wert, "bez" die Oberflächenbezeichnung desselben, "bez1" auf Sprache 1

optionadd versandart,1,'ab Werk,'CIF
```

optionrem

Syntax: optionrem vlName,vlWert

```
entfernt einen Eintrag aus einer Auswahlliste;

optionrem versandart,1
```

optionStd

Syntax: optionStd fieldName

```
setzt in einem Feld mit Werteliste zum Auswählen die verfügbaren Einträge auf
den im Modell definierten Standard zurück.

optionStd versandart
```

registerInvisible

Syntax: registerInvisible registerName[,...]

```
schaltet Register in der Detailansicht aus. Das ganze Register fehlt dann.
     Kein
e Wildcards, keine Arrays.

registerInvisible Lager
```

registerVisible

Syntax: registerVisible

```
schaltet alle Register auf sichtbar;

registerVisible
```

select

Syntax: select tableName[,q,soseq,msg,array]

```
öffnet das PBWOUebersicht Modul modal, konfiguriert auf Tabelle "tableName",
Suchbedingung "q", Sortierung "soseq" und Hinweis "msg". Der
getCMQForUebersicht Event wird nicht aufgerufen, man braucht keinen
Uebersichtsnamen, es wird intern "selectFromTableNamed" genommen.
Beim Click in der Uebersicht wird kein UebersichtDidClickRow Event ausgelöst,
stattdessen wird das angeklickte EO in $_rv gestellt. Funktioniert nur in
firstLevel Scripts.
```

```
Wird als letzer Parameter ein Array angegeben, wird nicht auf die Datenbank
zugegriffen, sondern die EOs des Arrays genommen.

select vid_lager,'bezeichnun like '%classic%',artikelnum,"bitte Artikslstamm
     wählen
logi selected:,$_rv
```

selObjChanged

Syntax: selObjChanged [modul]

```
löst den selObjChanged Event aus und damit Feldsteuerungen und Plugin Refresh.

selObjChanged $e
```

setDBValueForKey

Syntax: setDBValueForKey value,key[,modul]

```
setzt dem "modul" bzw. %datasource den "value" im dbFormat und geht dabei
     über
eine PBWOAsso namens "key", die dazu an der Oberfläche sein muss; simuliert
mithin einen Editiervorgang über die Oberfläche in das selObj hinein.
Der didValToTarget Event wird dabei ausgelöst.

setDBValueForKey 1234,artikelnum
```

setWodValueForKey

Syntax: setWodValueForKey value,key[,modul]

```
setzt dem "modul" bzw. %datasource den "value" im wodFormat und geht dabei
     über
eine PBWOAsso namens "key", die dazu an der Oberfläche sein muss; simuliert
mithin einen Editiervorgang über die Oberfläche in das selObj hinein.
WOD bedeutet Web Oberflächen Daten.
Der didValToTarget Event wird dabei ausgelöst.

setWodValueForKey '30.07.1966,datum
```

show

Syntax: show modul

```
bringt das "modul" an die Oberfläche

show $e
```

sichern

Syntax: sichern [modul]

```
entspricht dem Click auf den Sichern-Button in "modul" bzw. %datasource.
Das selObj wird gesichert, was sich in einem insert oder update niederschlägt.
Alle entsprechenden Events werden ausgelöst, will/did Save/Insert/Update

sichern $e
```

suchexternal

Syntax: suchexternal modul,qualifier

```
veranlasst das "modul" gemäß "qualifier" zu suchen; dem "modul" wird
%datasource (bzw. tvc in Plugins) als callingModul gesetzt, wenn es nicht
selbst die %datasource (bzw. der tvc in Plugins) ist. Anschließend wird
     "modul"
an die Oberfläche gebracht.


$qf,=,"artikelnum = 1234
$e,modulOrEnd,vid_lager
suchexternal $e,$q
```

zurueck

Syntax: zurueck [modul]

```
entspricht dem Click auf den Zurück-Button in "modul" bzw. %datasource. Es
     wird
in das callingModul zurückgesprungen, falls der Zurück-Button aktiv ist.


zurueck $e
```

12.13 Stringbearbeitung

=:,:=

Syntax: target,=:,value

```
zuweisung formatiert nach EO (target muss ein EO feld sein); der Wert wird so
formatiert, wie er aus der Datenbank kommen würde, damit nicht
fälschlicherweise eine Änderung des EOs erkannt wird;


$p_eo.bla,=:,$fasel
```

cai

Syntax: target,cai,string,index

```
Character at Index; liefert den Buchstaben, der in "string" an Stelle "index"
steht (ab 0). Der Character wird als 1-stelliger String geliefert.

$char,cai,$s,2
```

cat

Syntax: target,cat,string[,stringn...]

```
verkettet die aufgezählten Strings mit jeweils einem Space dazwischen

$s,cat,bla,fasel
-> "bla fasel"
```

cjs

Syntax: target,cjs,array,spearator

```
liefert einen String, der aus den mittels "separator" zusammengefügten
Einzelstrings aus "array" besteht. Pendant zu css.

$s,cjs,$a,\n
```

concat

Syntax: target,concat,string[,stringn...]

```
verkettet die aufgezählten Strings ohne Space dazwischen

$s,concat,bla,fasel
-> "blafasel"
```

css

Syntax: target,css,string,separator

```
liefert ein Array des mittels "separator" zerlegten "string";
Pendant zu cjs

$a,css,$file,\n
```

find

Syntax: target,find,string,suchstring[,start,option]

```
sucht in "string" nach "suchstring!; optional ab Position "start"; option
     "ci"
ist caseinsensitive; "back" backward search; options sind kombinierbar;
     liefert
als Ergebnis die erste Position des gefundenen suchstring oder wenn nicht
gefunden -1

target,find,$s,$search
target,find,$s,$search,123,ci
target,find,$s,$search,456,'back.ci
```

spf

Syntax: target,spf,format[,arg1...n]

```
entspricht dem sprintf() bzw. -[NSString stringWithFormat:...]
%i
%f
%@

$sqlf,=,"mitkurz = '%@'
$sql,spf,$sqlf,$_user
```

ss

Syntax: target,ss,string,start,length

```
liefert substring von "string" ab "start" in Länge "length", "start" beginnt
bei 0; "length" gibt an wieviele Zeichen der substring enthalten soll;
     length=e
bedeutet: bis zum Ende von "string"

$s,ss,$line,10,20
```

st

Syntax: target,st,lr,sim,length,string

```
formatiert den "string" links/rechtsbünding als String/Integer/Money in Länge
"length"

$s,st,r,m,10,$summe
```

12.14 Tutorial

12.14.1 Überblick

Wir bauen einen Pizza-Service

Szenario

Als Übungsbeispiel soll eine Pizza-Stammdatenverwaltung für einen Pizza-Express erstellt werden.

Eine Pizza hat eine Nummer, einen Namen, eine Größe, einen Preis (abhängig von Größe) und einen Belag.

Vorgehensweise

Zunächst wird die Pizza-Tabelle und das dazugehörige Modul im Modul "Modell Tabellen" angelegt, und dann im Modul "Workbench" mit Feldern versehen. Dann wird die Datenbank angepasst.

Nach einem "Reconfig" ist das Modul verfügbar. Nun können Sie erste Testdaten erfassen und die Oberfläche im "Config"-Modus anpassen.

Beim Arbeiten stellt man fest, was sinnvolle Feldvorbelegungen sind, Pflichtfelder, evtl. noch fehlende Felder, wie man suchen will (Kombi-Suchfelder markieren und Indexe) und wie die Bezeichnung der Objekte sein soll (Descri-Felder). Diese Erkenntnisse modelliert man in der Workbench nach und passt ggf. die Datenbank an.

Das Verhalten eines Moduls wird mit Event-Skripten angepasst: Feldsteuerung, Farben und Reaktion auf Eingaben (Vorbelegungen, Prüfungen).

12.14.2 neue Tabelle

Vorgehensweise

1. Öffnen Sie das Modul "Modell Tabellen".
2. Geben Sie eine Tabellenbezeichnung an.
3. Legen Sie sie als echte Tabelle fest.
4. Legen Sie Attribute an (Wechsel zur Workbench).
5. Klicken Sie auf Reconfig.
6. Legen Sie das Modul an.
7. Rufen Sie das Modul auf.
8. Gestalten Sie das Lif.
9. Tragen Sie Testdaten ein.
10. Verbessern Sie eventuell.

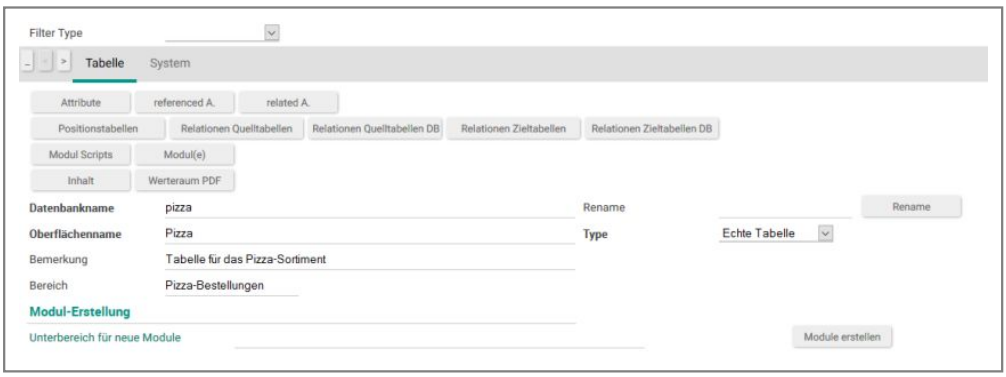
Pizza-Tabelle

Eine Pizza besitzt auf alle Fälle eine Nummer, einen Namen, einen Preis (je nach Größe) und einen Belag. Da sie in der Datenbank gehalten wird, ist es eine "real"-Table. Auf die Pizza-Nummer will man Einfluss nehmen können, sie soll nicht automatisch vergeben werden. Nummer ist ein Integer-Feld, Name ein Character, Preise vom Typ Money und der Belag ein Memofeld.

Tabelle anlegen

Öffnen Sie das Modul "Modell Tabellen". Sie finden es unter Home, System, Modell Tabellen.

1. Klicken Sie auf "Neu" und füllen Sie die Felder aus:
2. Datenbankname: pizza (unter diesem Namen wird die Tabelle in der MySQL-Datenbank angelegt)
3. Oberflächenname: Pizza (dieser Name wird in der Anwendung verwendet)
4. Type: Echte Tabelle (so wird sie in der Datenbank angelegt).
5. Bemerkung: Tabelle für das Pizza-Sortiment (dient als Hilfe für den Anwender)
6. Bereich: Pizza-Bestellungen (dient nur zur Information)
7. Klicken Sie auf "Sichern".
8. Anschließend klicken Sie auf den Button "Module erstellen".



Anlegen einer neuen Tabelle

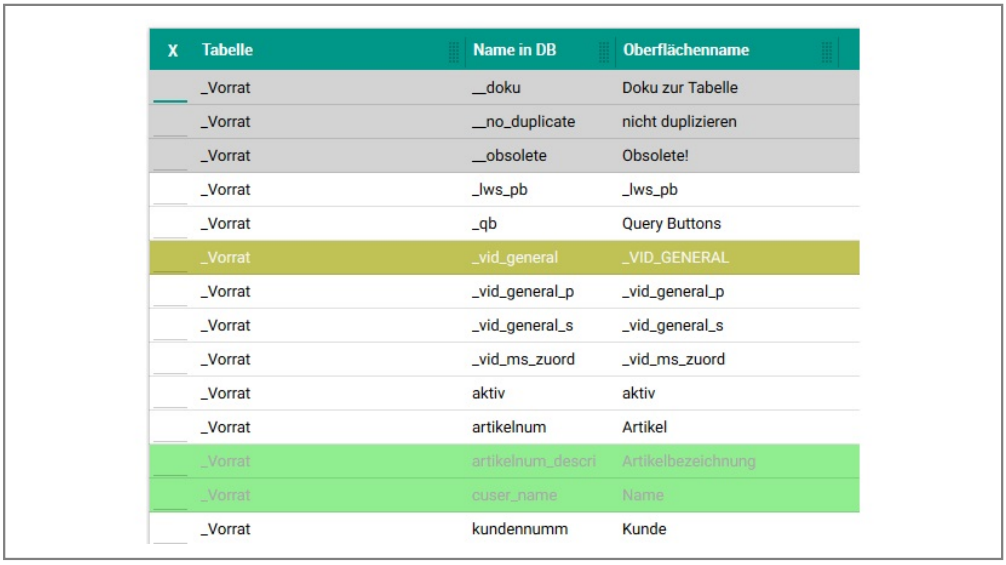
Attribute anlegen

Als Nächstes muss die Tabelle noch mit Spalten versehen werden. Klicken Sie auf den Button "Attribute". Damit öffnet sich die Workbench.

zu den Attributen

Übernehmen Sie einige Standard-Elemente in die neue Tabelle, indem Sie im Feld von Tabelle "_Vorrat" eingeben und dann auf "Übernehmen" klicken.

Wählen Sie hier _VID_GENERAL aus und klicken auf "Zurück".



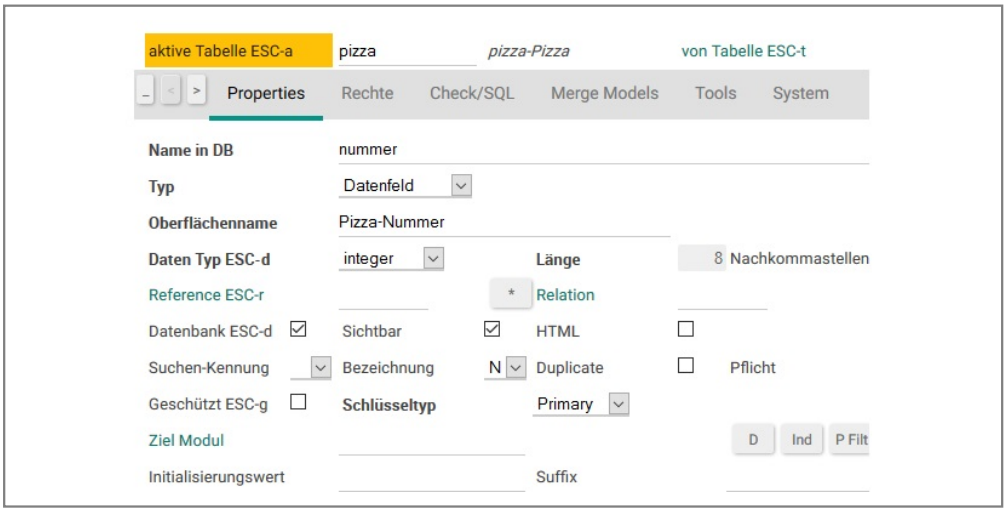
X	Tabelle	Name in DB	Oberflächenname
	_Vorrat	__doku	Doku zur Tabelle
	_Vorrat	__no_duplicate	nicht duplizieren
	_Vorrat	__obsolete	Obsolete!
	_Vorrat	_lws_pb	_lws_pb
	_Vorrat	_qb	Query Buttons
	_Vorrat	_vid_general	_VID_GENERAL
	_Vorrat	_vid_general_p	_vid_general_p
	_Vorrat	_vid_general_s	_vid_general_s
	_Vorrat	_vid_ms_zuord	_vid_ms_zuord
	_Vorrat	aktiv	aktiv
	_Vorrat	artikelnum	Artikel
	_Vorrat	artikelnum_descri	Artikelbezeichnung
	_Vorrat	cuser_name	Name
	_Vorrat	kundennumm	Kunde

Hinzufügen von abstractTables

Nun erzeugen Sie das primary Key-Feld "Pizza-Nummer".

1. Klicken Sie auf "Neu".
2. Füllen Sie dbName mit "nummer" (dieser Name wird später in Scripts verwendet)

3. Oberflächenname: Pizza-Nummer (dient zur Anzeige und sollte verständlich für den Benutzer sein)
4. Datentyp: integer (eine Zahl ohne Nachkommastellen)
5. Schlüsseltyp: Primary (jede Pizza-Nummer darf nur einmal vorkommen und muss vergeben sein)
6. Abschließend klicken Sie auf "Sichern".



Anlegen eines neuen Feldes

Im Register "Doku" kann neben der englischen Bezeichnung auch die Hilfe für das Feld in doku0 eingetragen werden. Das Feld doku1 ist für die englische Hilfe. Die Hilfe wird abhängig von den Spracheinstellungen des Benuzters sowohl als Tool-Tip für das Feld angezeigt, als auch in der Modul-Hilfe, die durch klicken auf den aktuellen Modulnamen geöffnet werden kann.

Auf die gleiche Art können Sie weitere Felder anlegen. Allerdings darf es nur einen primary key geben.

Legen Sie folgende Felder an:

```
+-------------+-------+
| dbName      | Typ   |
+-------------+-------+
| pizzaname   | char  |
| belag       | char  |
| preis_gross | money |
| preis_klein | money |
+-------------+-------+
```

Tipp:

Wenn Sie mehrere Felder anlegen, die sich nur in wenigen Merkmalen unterscheiden, können Sie ein Attribut mit Klick auf "Dup" duplizieren. Dadurch werden alle Felder mit den Angaben des kopierten Feldes vorbelegt. Der Datenbankname muss dann aber in jedem Fall geändert werden, da dieser immer nur einmal je Tabelle auftauchen darf.

Anpassen der Datenbank und Reconfig

Jetzt müssen die Änderungen, die an der Datenbank vorgenommen wurden, in die Datei "Mandant.idm" geschrieben werden, um dann daraus das Modell neu zu laden.

Das geht folgendermaßen:

1. Klicken Sie auf "DB Diff" (Nun werden alle Änderungen analysiert und dann im Plugin angezeigt. Die Unterschiede werden in einen SQL-Befehl umgewandelt und im Feld SQL angezeigt. Hier kann dieser verändert und erweitert werden.)
2. Klicken Sie auf "Exec" (Führt den SQL-Befehl aus dem Feld "SQL" aus.)
3. Klicken Sie auf "Save to IDM" (Speichert das Modell in die Datei "Mandant.idm)
4. Zu guter Letzt klicken Sie auf "Reconfig" (Baut die laufende Anwendung auf Basis vom "Mandant.idm" um.)

Datenbank anpassen und Modell neu laden

12.14.3 neues Modul

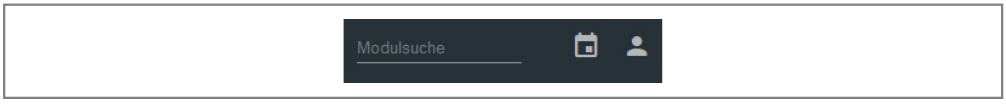
Überblick

Das geänderte Modell wurde durch Rekonfiguration der Anwendung aktiviert. Nun ist die neue Tabelle in IntarS 7 bekannt. Um etwas damit anfangen zu können, benötigen Sie jedoch noch ein Modul. Erzeugen Sie ein Modul, indem Sie im "Modell Tabellen"-Modul ein neues Modul erzeugen und einen Platz im Menü zuweisen. Auf diese Weise werden Sie ein "Pizza"-Modul erstellen. Nach dem Anlegen ist das neue "Pizza"-Modul sofort im Menü verfügbar und Sie können damit arbeiten, Erfahrungen sammeln, die Oberfläche einrichten und weiterführende Funktionalitäten entwickeln.

Schritt für Schritt

"Modell Tabellen"-Modul

Melden Sie sich als Administrator an und geben Sie in der Modulsuche "Tabelle" ein. Alternativ finden Sie die Tabellenverwaltung unter dem Modul "Home" im "System"-Menü.

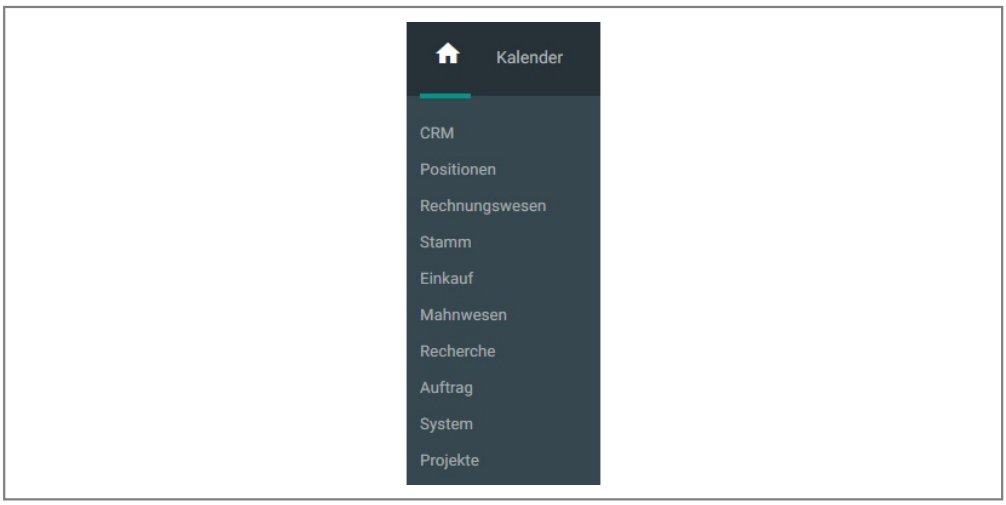


Module Suchen

Neues Pizza-Modul anlegen

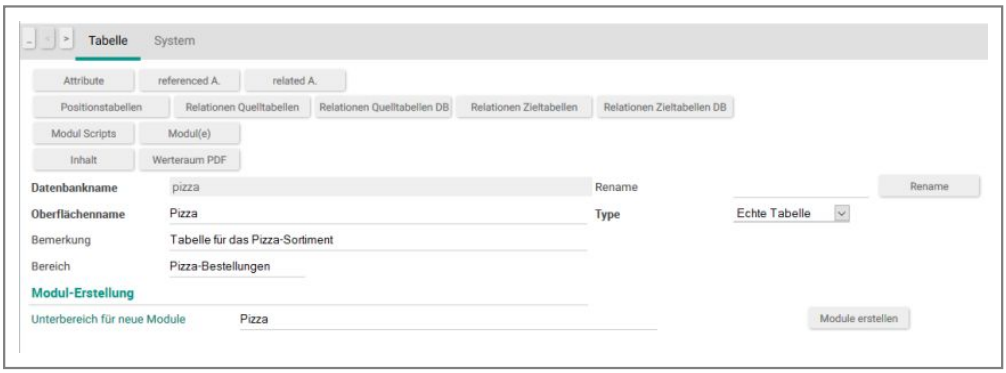
Geben Sie "Pizza" im Suchfeld ein, um die eben erzeugte Tabelle zu finden.

Im Home-Modul sind die Module in Unterbereiche gruppiert (z.B. System oder CRM).



Unterbereiche für die Navigation im Home-Modul

Um das Modul anzulegen, geben Sie einen solchen Unterbereich (Dieser muss noch nicht existieren. Gibt es ihn nicht, wird er von IntarS 7 angelegt.), im Feld "Unterbereich für neue Module" ein und klicken Sie auf Module erstellen.



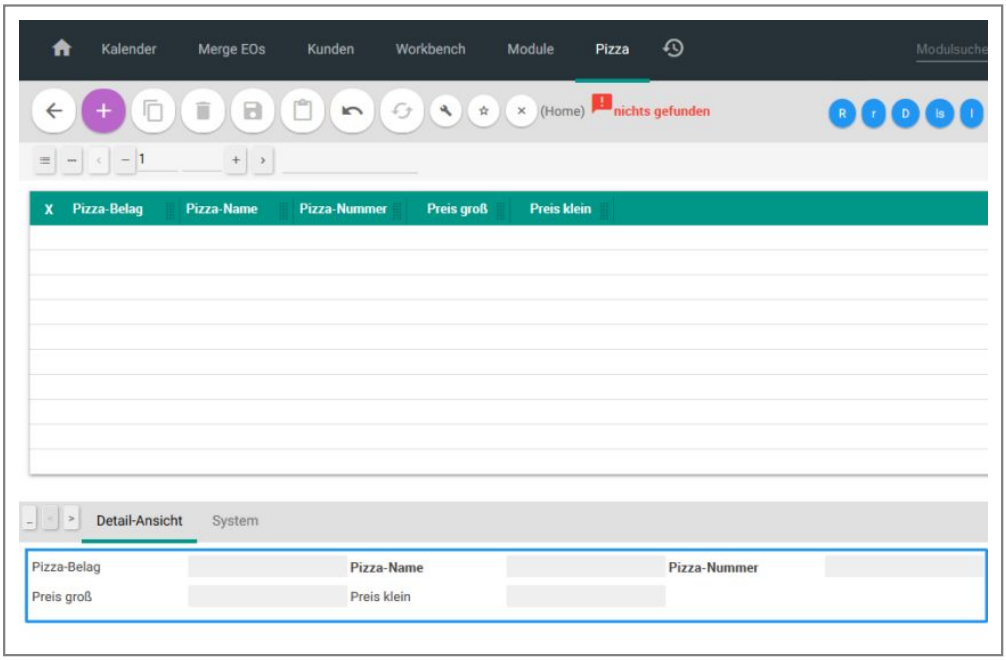
Anlegen eines Neuen Moduls

Das "Pizza"-Modul aufrufen

Danach ist das neue "Pizza"-Modul im Menü verfügbar und kann durch Anklicken aufgerufen werden.

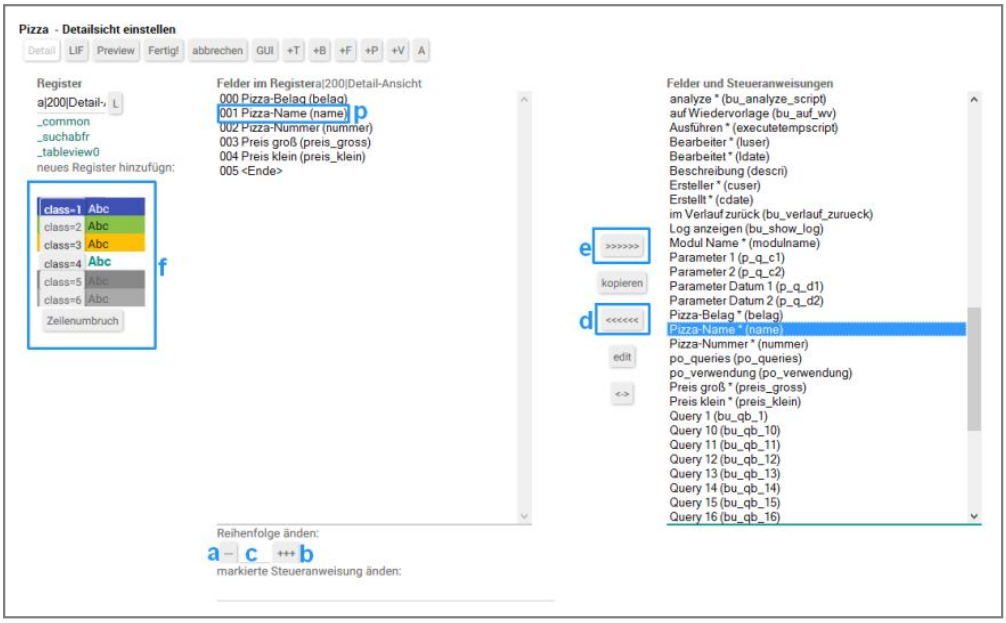
Oberfläche einrichten

Beim ersten Aufruf werden die Oberflächenelemente automatisch vom Framework regelbasierend angeordnet. Schlüsselfelder und Pflichtfelder werden z. B. vor die restlichen Felder gestellt. Das sieht etwa so aus:



Ohne Oberflächen-Einstellung

Dieses Layout kann manuell angepasst werden. Sie können dies im "Config"-Modus tun, indem Sie auf den Schraubenschlüssel klicken oder die Tastenkombination "Alt-o" verwenden.



Oberflächeneinstellung-Detailsicht

Hier sehen Sie links die vorhandenen Register mit den dazu gehörenden Oberflächenelementen. Auf der rechten Seite werden alle Felder und Steuerelemente angezeigt.

Reihenfolge der Felder verändern

Markieren Sie "Pizzaname" (siehe Oberflächeneinstellung-Detailsicht > p) und drücken Sie den Button "nach oben" (a). Damit wird der Pizzaname um eine Stelle nach oben verschoben. Analog verschiebt ihn ein Klick auf den Button "nach unten" (b) eine Position nach unten. Geben Sie in das Feld (c) 003 ein, um den Pizzaname direkt auf Position 3 zu schieben.

Felder einfügen oder ausblenden

Markieren Sie "Pizzaname" und drücken anschließend "in Register aufnehmen" [<<<<<<] (d) oder verwenden Sie die Tastenkombination "Alt-a". Der Pizzaname wandert nach links ins Register. Er erscheint damit auf der Oberfläche. Analog wird er durch Klick auf "aus Register entfernen" [>>>>>>] (e) oder der Tastenkombination "Alt-r" aus dem Register und damit von der Oberfläche entfernt.

Umbruch einfügen

Markieren Sie links "Pizzaname" und klicken anschließend auf "Zeilenumbruch" oder verwenden die Tastenkombination "Alt-x", um einen Umbruch vor "Pizzaname" einzufügen. Dies bewirkt, dass "Pizzaname" in einer neuen Zeile links steht.

Felder farblich markieren

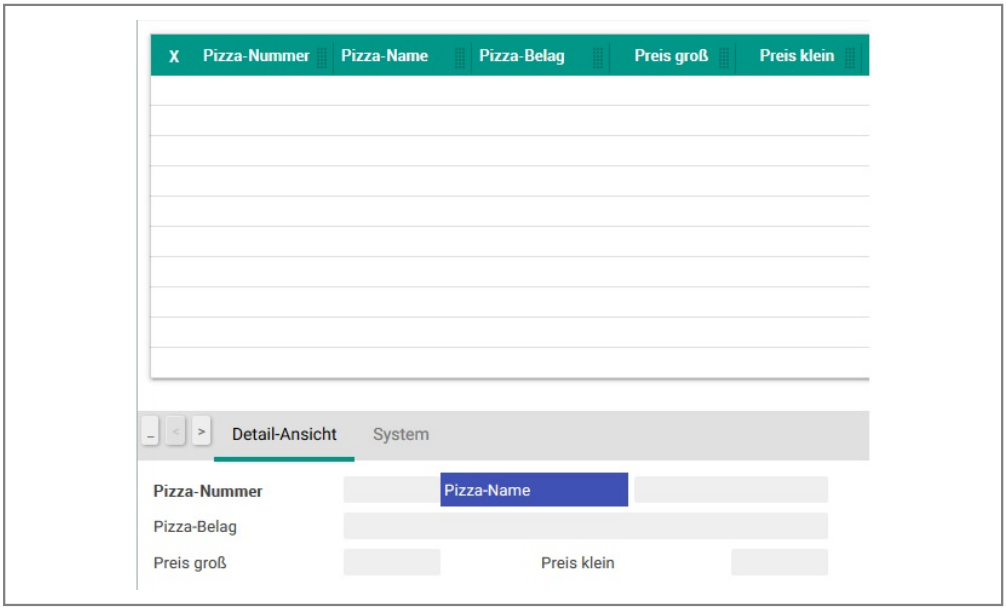
Markieren Sie "Pizzaname" (p) und klicken Sie auf <class=1> (f), um diesen in der entsprechenden Farbe zu markieren.

Auswahlliste Spalten einstellen (Spaltenbreite, Reihenfolge)

Klicken Sie auf "Liste" (g), markieren Sie "Bearbeiter" und klicken Sie auf ">>>>>", um ihn aus der Liste zu entfernen. Markieren Sie "Pizzaname", geben Sie in das Spaltenbreitenfeld unten ("Breite verändern") 200 ein und drücken Sie die "Return"-Taste. Damit haben Sie die Spalte auf 200 Pixel Breite konfiguriert.

Gelegentlich können Sie "Preview" klicken, um sich anzeigen zu lassen, wie es aussehen würde. Sind Sie mit dem Ergebnis zufrieden, klicken Sie auf "Fertig!" oder verwenden die Tastenkombination "Alt-s". Sie können den "Config"-Modus jederzeit wieder aufrufen.

Nachdem die Oberfläche angepasst wurde, kann das Ergebnis wie folgt aussehen:



Nach der Oberflächen Einstellung

12.14.4 Daten erfassen

Überblick

Nachdem Sie die Tabelle "Pizza" mit den zugehörigen Attributen angelegt haben, müssen die Pizzen manuell in das System eingegeben werden.

Schritt für Schritt

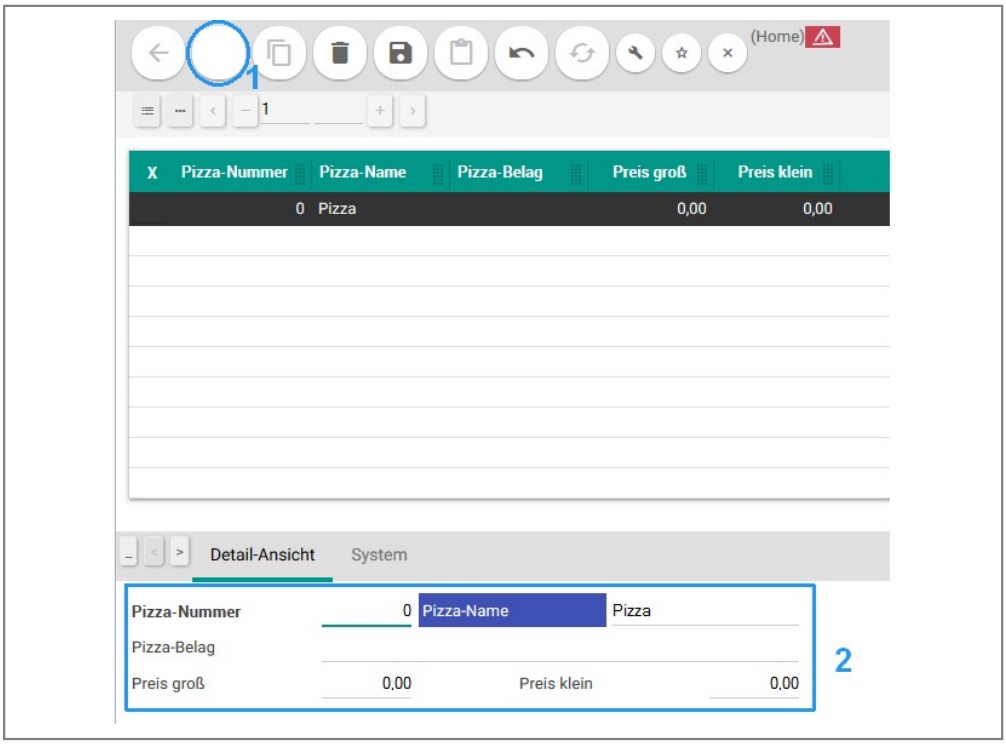
1. Daten erfassen

Um die Pizzen zu erfassen, klicken Sie auf "Neu" oder verwenden die Tastenkombination "Alt-n" und füllen die entsprechenden Felder aus. Positionieren Sie den Cursor dabei mit der "Tab"-Taste weiter. Mit "Sichern" (oder der Tastenkombination "Alt-s") wird der neue Satz dauerhaft in der Datenbank gespeichert.

Geben Sie bei "Pizza-Nummer" "1", bei "Pizzaname" "Pizza Salami", bei "Preis_gross" "5,10", bei "Pizza_klein" "3,10" und bei "Belag" "mit Salami" ein.

Erfassen Sie nun bei "Pizza-Nummer" "2", bei "Pizzaname" "Pizza Tonno", bei "Preis_gross" "6,10", bei "Pizza_klein" "3,60" und bei "Belag" "mit Thunfisch und Zwiebeln".

Geben Sie bei "Pizza-Nummer" "3", bei "Pizzaname" "Pizza Italia", bei "Preis_gross" "6,60", bei "Pizza_klein" "4,00" und bei "Belag" "mit Spinat, Paprika, Mozzarella" ein.



Daten erfassen

2. Daten bearbeiten

Wenn Sie einige Pizzen erfasst haben, sind diese in der Auswahlliste zu sehen. Sind es mehr als 10, werden die Buttons für seitenweises blättern aktiv. Um eine bereits erfasste Pizza nachträglich zu ändern, klicken Sie diese in der Auswahlliste an. Der Datensatz erscheint daraufhin in der Detailsicht, der Cursor steht auf dem ersten editierbaren Feld. Ändern Sie nun ein Detail an der Pizza (z. B. "Preis klein") und drücken Sie die "Return"-Taste. Daraufhin erscheint ein blinkendes Warnzeichen, welches signalisiert, dass ungespeicherte Änderungen vorliegen. Gleichzeitig wurden Buttons deaktiviert, die in diesem Stadium nicht sinnvoll sind. Klicken Sie auf

"Speichern" (oder verwenden die Tastenkombination "Alt-s"), um die Änderung zu sichern. Das Warnzeichen verschwindet, die Änderung wird dauerhaft gespeichert und die deaktivierten Buttons werden wieder aktiv.

3. Löschen

Wählen Sie eine Pizza aus und klicken Sie auf "Löschen" (oder verwenden die Tastenkombination "Alt-r"). Bestätigen Sie die Sicherheitsabfrage. Daraufhin verschwindet die gewählte Pizza aus der Trefferliste. Es wird automatisch auf die nächste positioniert. Die gelöschte Pizza ist dauerhaft entfernt, es gibt keine Möglichkeit, sie wiederherzustellen. Löschen Sie daher nur, wenn Sie sich sicher sind, dass Sie diese nicht mehr benötigen. Sie können diese jedoch neu erfassen.

4. weitere Buttons

Positionieren Sie den Mauszeiger auf die anderen Buttons und lesen Sie die Tooltipps. Machen Sie sich mit den Druck- und Export-Möglichkeiten vertraut.

12.14.5 Verfeinerungen

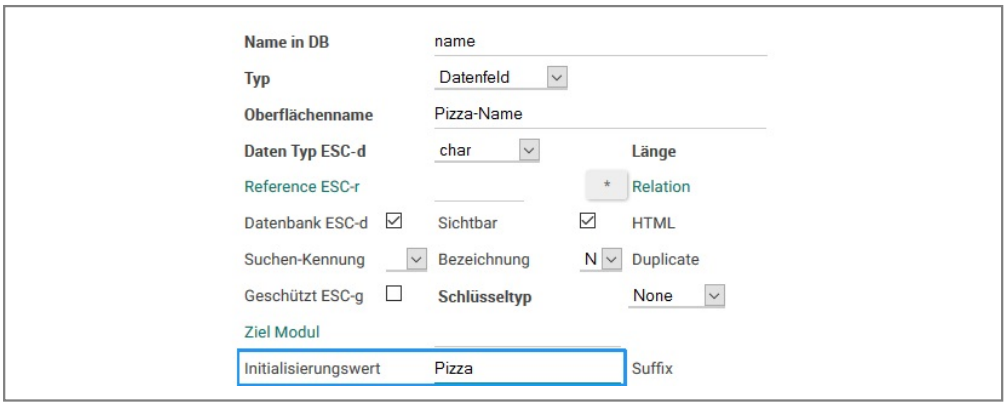
Überblick

Beim Verwenden des Moduls sind Ihnen evtl. ein paar Verbesserungs-Ideen eingefallen. So ist es möglich, Pizzen ohne Name zu erfassen. Beim nachträglichen Ändern ist es mühsam, durch wiederholte Tabulatorsprünge auf ein weiter unten stehendes Feld zu gelangen. Wenn Sie versucht haben, Pizzen nach Name zu suchen, wurde nichts gefunden. Alle Pizzanamen fangen mit "Pizza" an, daher wäre es optimal, wenn der Name schon so initialisiert ist. Auf diese Ideen wären Sie ohne die Benutzung des Moduls nur mit entsprechender Erfahrung gekommen. Diesem Umstand wird die iterative Vorgehensweise gerecht. In kurz aufeinanderfolgenden Design-Implementierung-Testzyklen entwickelt sich die Software bedarfsgerecht. IntarS 7 ist für iterative Vorgehensweise optimiert.

Schritt für Schritt

1. Vorbelegungen

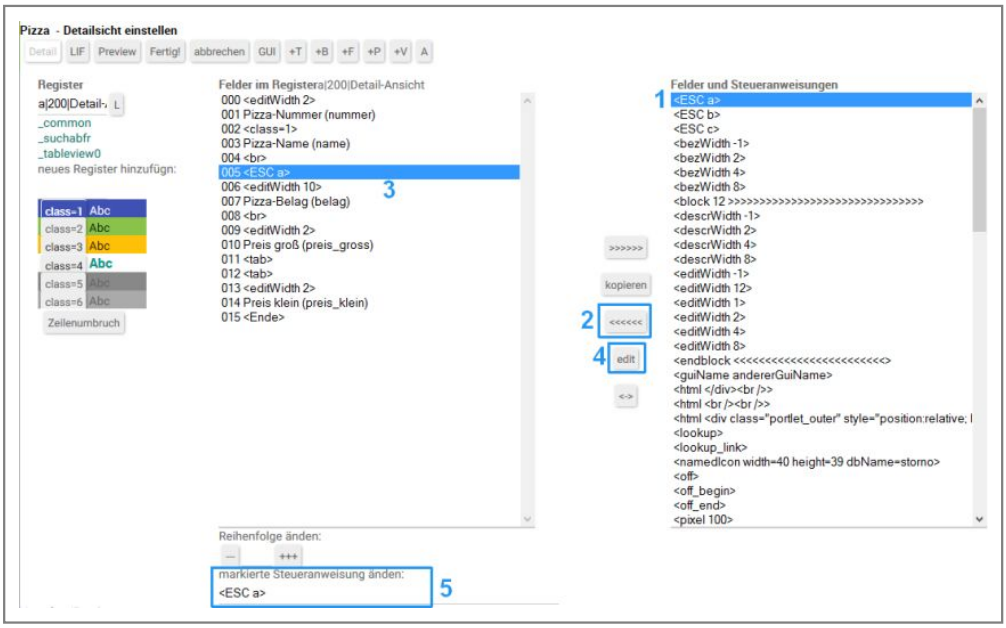
Bei der Datenerfassung fällt auf, dass alle Pizzen mit "Pizza" und Belag immer mit "mit" anfangen. Deshalb ist es sinnvoll, die Felder "Pizzaname" mit "Pizza" und "Belag" mit "mit" zu belegen. Dies erreicht man, indem in der Workbench der Initialwert für diese Felder hinterlegt wird. Dazu rufen Sie aus dem Modul die Workbench über den Button "A" auf. Dort markieren sie das Feld "pizzaname" und geben in die Eigenschaft "Initialisierungswert" "Pizza" ein. Anschließend "speichern". Nach dem gleichen Prinzip verfahren Sie auch für den Belag. Um Änderungen an Tabellenattributen zu aktivieren ist wieder ein Reconfig nötig.



Vorbelegung

2. ESC-Sprungziele

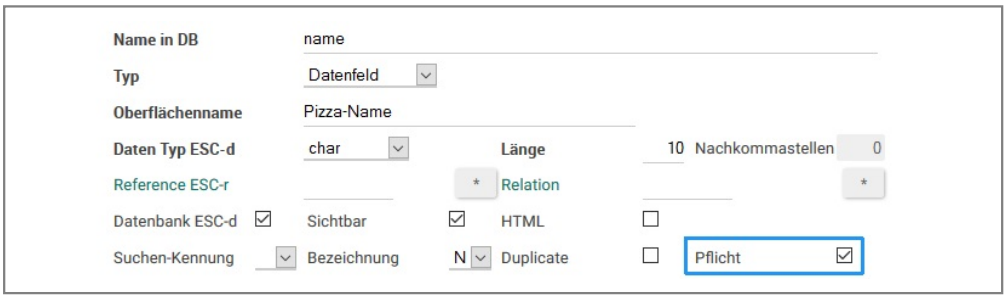
Sie können Eingabefelder im Modul mit einem Sprungziel versehen, um wiederholtes Tabbing zu sparen. Ein Sprungziel wird angesprochen durch die "ESC"-Taste und eine anschließende Folgetaste. Um z. B. das Eingabefeld "Belag" mit einem Sprungziel zu versehen, fügen Sie im "Config"-Modus das Tag "<ESC a>" vor dem Belag ein. An der Oberfläche wird das Sprungziel angezeigt. Der Belag heißt jetzt "Belag (ESC-a)". Drücken Sie die "ESC"-Taste und danach "a", so wird der Cursor in das Eingabefeld "Belag" positioniert.



Sprungzeile definieren

3. Pflichtfelder

Um zu bestimmen, dass alle Pizzen zwingend einen Pizzanamen bekommen, muss das Feld als Pflichtfeld markiert werden. Dazu öffnen Sie aus dem Modul "Pizza" die Workbench und wählen darin das Feld "pizzaname" und tragen in das Feld machen bei Pflicht ein Häkchen. Anschließend "speichern". Damit ist "Pizzaname" bei der Dateneingabe ein Pflichtfeld. Führen Sie gleiche Vorgehensweise bei "Pizza-Nummer", "Preis gross" und "Preis klein" durch. Die Pflichtfelder werden in der Anwendung Fett dargestellt.



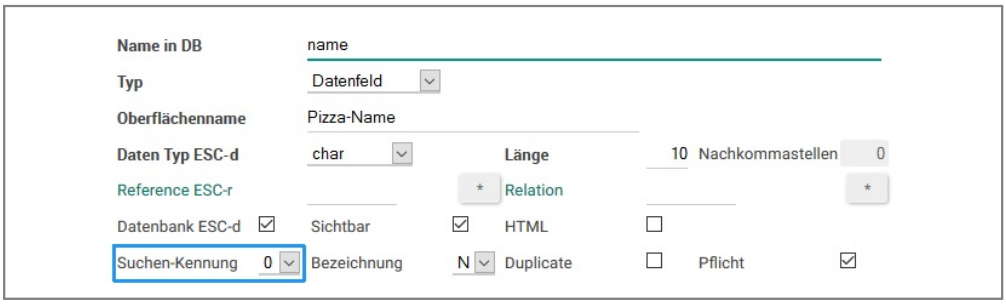
Pflichtfeld

4. Suchen

Kombisuchfelder:

Kombisuchfeld

Damit eine Pizza auch bequem über das Kombi-Suchfeld nach Name gesucht werden kann, muss dieser dafür markiert werden. Das ist in der Workbench möglich. Rufen Sie in der Workbench das Feld "Pizzaname" auf und setzen den Wert von "Suchen-Kennung" auf 0. Beim Feld "Nummer" setzen Sie Suchen-Kennung auf leer. Damit wird nicht mehr nach der Nummer sondern nach Name gesucht.



Kombisuchfeld WB

13 Anhang

13.1 FAQ

Welche Directories sind wichtig und wofür?

- MANDANTPATH ${NEXT_ROOT}/Local/Library/IntarS/_K_000201; enthält das gesamte Repository
- Source des Frameworks ${NEXT_ROOT}/Local/Projects/IntarS7; enthält Source des Frameworks
- ${NEXT_ROOT}/Local/Library/IntarS/_K_000201/Scripts enthält alle Skripte der Geschäftslogik.

Wie finde ich diese schnell?

Speichern Sie sie als Favoriten.

Wie und wann mache ich ein SVN-Update?

In der Anwendung unter Home/Service durch Klick auf den entsprechenden Button.

Oder im Filesystem: Drücken Sie die rechte Maustaste auf dem Directory, welches upgedated werden soll und wählen Sie "SVN-Update" aus. Wenn Sie sich nicht sicher sind, machen Sie auf den gesamten MANDANTPATH ein "SVN-Update". Außerdem machen Sie dies auf den GLOBALCONFIGPATH und den Source des Frameworks.

Sie sollten ein Update ausführen, bevor Sie selbst das Programmieren beginnen. Schließen Sie bitte vorher den Editor, um zu vermeiden, dass Sie anschließend veraltete Versionen speichern.

Was kann dabei fehlschlagen und was ist dann jeweils zu tun?

Beim Update kann es zu Konflikten kommen, wenn ein File an derselben Stelle lokal und von einen anderem verändert wurde. In diesem Fall müssen die Konflikte aufgelöst werden:

1. In der Liste vom SVN doppelklicken Sie auf die Zeile mit dem Konflikt.
2. Es öffnet sich eine Merge-Anwendung.
3. Navigieren Sie zum ersten Konflikt und wählen eine der vier Möglichkeiten aus.
4. Positionieren Sie zum nächsten Konflikt. Machen Sie dies so lange, bis alle aufgelöst wurden.
5. Speichern Sie.
6. Klicken Sie auf das Icon "resolved".
7. Schließen Sie die Merge-Anwendung. Speichern Sie nochmal, falls nötig.
8. Bearbeiten Sie die weiteren Files mit Konflikt wie eben beschrieben.

Beachten Sie bitte, dass beim Kopieren von Directories, ein .svn-Directory anschließend gelöscht wird. Es gehen sonst beide Directories auf dasselbe SVN-Directory und beeinträchtigen sich gegenseitig.

Sie benötigen eine Verbindung zu einem externen SVN-Server (Firewall, Netzwerk) und eine gültige Anmeldeinformation, sodass SVN etwas tun kann. Unter TortoiseSVN/Settings/Saved Data/Authentication Data/clear können evtl. gespeicherte und nicht mehr gültige Anmeldeinformationen gelöscht werden.

Probleme gibt es außerdem, wenn Sie .svn-Directories manuell gelöscht haben. In diesem Fall müssen Sie das entries-File im übergeordneten Directory mit dem Editor korrigieren.

Wie und wann mache ich SVN-Commit?

In der Anwendung unter Home/Service durch Klick auf den entsprechenden Button.

Oder im Filesystem: Klicken Sie mit der rechten Maustaste auf das gewünschte Directory und wählen Sie "SVN-Commit" aus.

Nach jeder abgeschlossenen und getesteten Bearbeitung sollten Sie ein "SVN-Commit" durchgeführen, sodass Ihre Arbeit auf dem Server landet.

Was kann dabei missglücken und was ist dann jeweils zu tun?

- Sie haben keine Verbindung zum Server.
- Sie besitzen keine ausreichende Berechtigung.
- Auf den Server wurde zwischenzeitlich von jemand anderem committed. Es muss erst ein "SVN-Update" gemacht werden.
- Sie haben vergessen, im Kommentarfeld etwas einzutragen. Der Server verweigert dann den Commit.
- Wenn Sie manuell .svn-Directories gelöscht haben, müssen Sie das entries-File im übergeordneten Directory mit dem Editor korrigieren.
- Rekursive Deletes verursachen Probleme. Am besten Sie arbeiten sich von unten nach oben durch.
- Sind von einem Commit noch Locks gesetzt, müssen Sie zuerst ein "SVN clean" machen. Dies wird Ihnen aber mitgeteilt.

Wie kann ich weitere Informationen aus dem System entlocken?

- Aktivieren des Debug Modus durch Klick auf den "D" Button oben im Modul.
- Klick auf den "L" Button oben im Modul, um das Log anzuzeigen.
- "Config"-Modul: hier gibt es einige Flags, um den Log-Ouptput zu erhöhen.
- "debug"-Befehl in Scripting: von debug bis enddebug werden alle Statements mit ihren Parametern gelogged.
- "log"-Befehl in Scripting: es können gezielt Informationen ausgegeben werden; sie landen dann im Log.
- "logi"-Befehl: ähnlich wie "log"-Befehl, wird aber auch interaktiv am Bildschirm ausgegeben.
- Im Temp-Script eines jeden Moduls können Sie Script-Befehle eingeben, um den Zustand des Systems zu untersuchen und Informationen auszugeben.

Wo finde ich Hilfe?

- Online-Hilfe mit dem "Hilfe"-Button; dort ist auch dieses FAQ enthalten
- Modul-Hilfe: Klicken Sie hierzu das Modul ein zweites Mal an
- Handbuch (PDF)
- Kommentare der Skripte
- Kommentare in den Sourcen des Frameworks
- Zu den GNUstep Klassen und in der Apple (tm) Foundation Classes Dokumentation.

Wie finde ich heraus, wie ein/e Feld/Modul/Tabelle heißt?

- Im "System"-Register wird interne "Modul Name" und der "Tabellenname" angezeigt.
- Durch einen Klick auf den "C"-Button bzw. die Tastenkombination "Alt-o" gelangen Sie in den "Config"-Modus. Dort sehen Sie die internen Namen aller Bildschirmelemente. In der Vorratsliste rechts kann die Sortierung zwischen GUI/dbName umgeschaltet werden.
- Klick auf den "A" Button oben im Modul, um die Workbench mit den Feldern der Tabelle des Moduls aufzurufen.

Wie finde ich heraus, welches File ich editieren muss?

Klick auf den "E" Button oben im Modul öffnet eine Ansicht, in der alle implementierten Events des Moduls angeboten werden. Ausserdem werden die letzten 20 ausgeführten Scripts angezeigt. Die Namen nicht implementierter Events erscheinen farbliche abgesetzt. Klickt man darauf, wird das Eventscript nach Bestätigung angelegt.

Klick auf den "S" Button oben im Modul zeigt die Scripts an, die zum Modul gehören.

Verwendung der Script-List und Scripts Module, um nach Scripts zu suchen und diese im Editor zu öffnen bzw. online zu editieren.

Handelt es sich um das Script eines Buttons, muss zunächst der Button in der Workbench gefunden werden (s. vorherigen Punkt). Steht in seinem Expression-Feld nichts, heißt das Script so wie der Button und liegt in dem Scriptdirectory, welches wie die Tabelle heißt. Sonst steht der Name des Skripts in dem Expression-Feld. Bei sehr kleinen Skripten steht das Skript auch komplett im Expression-Feld und wird dann auch dort editiert. Neben dem Expressionfeld liegt ein "Open"-Button, der das zu editierende Scriptfile öffnet.

Skripte eines Plugins befinden sich alle als subScripts in einem Skript-File. Dieses heißt wie das Plugin.

Was ist ein EnterpriseObject (EO)?

Ein EO enthält die Daten eines Datensatzes aus einer Datenbanktabelle. Man kann auf die Daten über die Namen der Attribute zugreifen. Man kann Relationen verfolgen und die Bezeichnungen von Dropdown-Feldern abrufen. Verändert man die Daten, kann das EO feststellen, was verändert wurde und die Änderungen auch wieder rückgängig machen. EOs liefern außerdem die Informationen über Tabelle und Attribute wie sie im Modell definiert wurden. EOs entstehen, indem man sie aus der Datenbank beschafft. Sie werden dabei durch den Object-Relational-Mapper erzeugt. Man kann ein EO auch selbst erzeugen, mit Daten versehen und in die Datenbank schreiben. EOs sind die Datenobjekte, welche im PBWOEditor in der Trefferliste liegen und als selObj bearbeitet werden.

Was kann ich alles mit einem EO machen?

Sie können ein EO lesen, verändern, erzeugen, insert, updaten und entfernen.

Beispiel:

```
$a,getEOsQSoa,vid_lager,$q,$soa
```

beschafft ein Array von EnterpriseObjects aus der Datenbanktabelle "vid_lager" unter Anwendung des Qualifiers $q und des Sortorderarrays $soa (serverside sort).

```
$eo,=,$a.firstObject
```

$eo.preis1,=,10.5

updat $eo

dem ersten EO aus dem Array wird als preis1 10,50 gesetzt und dann wird das EO in die Datenbank zurückgeschrieben.

```
delete $eo
```

das EO wird gelöscht.

Was ist ein Qualifier und wozu braucht man ihn?

Ein Qualifier ist die objektorientierte Beschreibung einer Treffermenge analog der where-Clause eines SQL-Statements. Qualifier dienen dazu, der Datenbank zu beschreiben, welche Eigenschaften EOs haben sollen, die man aus der Datenbank holen will.

Effizienter Umgang mit dem Editor Notepad++

Verwenden Sie Tastenkombinationen, um schneller arbeiten zu können.

F2: Markierung setzen und hinspringen

F3: Schnellsuche, verwendet letzten Suchstring

Ctrl-F: Sucht in Datei

Ctrl-H Sucht in Dateien und ersetzt durch gewünschten Ausdruck

Ctrl-M: Markiert identische Strings

Doppel-Klick: Markiert nur das Wort

3-fach-Klick: Markiert die ganze Zeile

Was mache ich, wenn die Anwendung ein völlig unerklärliches Verhalten zeigt?

Cleanen Sie das Framework und builden Sie es neu.

Welche Bedeutung hat mutable (z. b. MutableString , MutableArray)?

Mutable vor einem Datentyp bedeutet, dass dieser Datentyp änderbar ist. In ein MutableArray und MutableDictionary können weitere Objekte eingefügt werden und bestehende entfernt werden. Ein MutableString kann mit appendString: verlängert werden.

Was ist ein rufendes Modul?

Das ist das Modul, aus welchem Sie zu dem gekommen sind, in dem Sie sich gerade befinden. Z. B.: Sie befinden sich gerade im Modul "Kunde". Sie möchten für diesen Kunden ein neues Angebot erstellen. Klicken Sie dazu auf den Button "neues Angebot". Jetzt können Sie sehen, dass Sie aus dem Modul "Kunde" in das Modul "Angebot" gelangt sind. Das Modul "Kunde" ist nun das rufende Modul.

Wie lege ich ein neues Stichwort an?

Rufen Sie das Modul "Stichworte" auf. Klicken Sie auf "Neu", füllen Sie die Felder aus und klicken Sie auf "Sichern".

Wie kann ich Stichworte einem Kunden zuordnen?

Rufen Sie das Modul "Kunde" auf. Im Register "Start" lokalisieren Sie das Stichwort-Plugin. Bei diesem gibt es einen Button "Hinzufügen". Klicken Sie darauf. Wählen Sie die Stichworte aus, die Sie hinzufügen möchten. Klicken Sie dann auf den "Zurück"-Button.

Kann ich Spezialpreise definieren, die nur für einen bestimmten Zeitraum gelten sollen?

Ja, dies funktioniert über das Modul "Spezialpreise". Sowohl im Kunde als auch im Artikelstamm gibt es einen Button "neuer Kundenpreis", der einen neuen Spezialpreis anlegt und dabei Kunde bzw. Artikel vorbelegt. Als dritte Möglichkeit gibt es den Button "Kundenpreise" im Kundenmodul, der alle bestehenden Spezialpreise des Kunden aufruft. Dort kann man sie dann verwalten.

Kann ich Staffelmengen für Einkaufspreise eingeben?

Ja. Klicken Sie im Modul "Artikelstamm" auf das Register ">EK". Hier können Sie entsprechende Staffelpreise hinterlegen.

Warum wird bei einem Kunden Umsatzsteuer ausgewiesen?

Es gibt zwei Möglichkeiten.

1. Das Lieferland ist Deutschland.
2. Das Lieferland ist in der EG und der Kunde hat keine gültige UStID für dieses Lieferland.

13.2 Begriffserklärung

Applikationsinstanz

Kurz "Instanz". Ein laufender IntarS 7-Prozess. Jede Instanz hat eine fortlaufende Nummer und wird im /etc/init.d/int7000201 gestartet. Es gibt mindestens eine Instanz mit der Nummer 1. Zur Skalierung kann IntarS 7 in mehreren parallelen Instanzen betrieben werden, auch auf mehrere Applikationsserver verteilt. Die Instanz 1 hat dabei instanzübergreifende Sonderaufgaben wie z.B. das Metadatenmodell aus mandant.idm in die Datenbank zu laden.

batch

wörtlich "Stapel", im übertragenen Sinne "Stapelverarbeitung" und heutzutage "Hintergrundverarbeitung".

Speziell in IntarS 7: zeitlich gesteuerte Hintergrundverarbeitung von Scripts, konfiguriert in der /etc/crontab.

Binary

ausführbares, kompiliertes Programm.

Community

Gemeinschaft, die ein Open Source Produkt wie IntarS 7 gemeinschaftlich weiterentwickelt.

Decker

stellt Material zur Deckung von Material-Bedarfen zur Verfügung, z.B. Bestellungen, Fertigungsaufträge, Lagerbestand.

Dump

Vollständiger Datenexport einer Datenbank für Sicherungszwecke.

Framework

Interne, kompilierte technische Verarbeitungsschicht von IntarS 7, im Gegensatz zur gescripteten, individualisierten Gechäftslogik.

GUI

Graphical User Interface = grafische Bedienoberfläche.

Host

Server, auf dem IntarS 7 läuft, im Gegensatz zum Client, auf dem der Browser zur Bedienung über die GUI läuft.

Implementierung

Programmierung.

Interpreter

Teil des Frameworks, das die Scripts abarbeitet.

Layout-Engine

Teil des Frameworks, das aufgrund der Einstellungen in den Layout-Info-Files (LIF) die GUI in HTML oder XHTML für den Browser rendert.

LIF

Layout-Information-File: definiert, welche Register im Detailbereich in welcher Reihenfolge angezeigt werden und welche Elemente in welcher Reihenfolge in den Registern liegen und welche Spalten in welcher Reihenfolge und Breite die Trefferliste anzeigt. (-> Oberflächenkonfiguration).

Metadatenmodell

Auch Repository, DataDictionary, enthalten im Mandant.idm, beschreibt die Datenbankstruktur von IntarS 7. Wird im Workbench-Modul verwaltet.

Modul

In sich logisch abgeschlossener Teilbereich von IntarS 7, unterster Gliederungspunkt im Menü. Besitzt eine eigene Oberflächenkonfiguration (LIF) und meist eine eigene Datenbanktabelle.

Parser

Teil des Interpreters, analysiert den Quellcode der Scripts und bereitet sie zur Ausführung vor.

PDF-Viewer

Programm zur Anzeige von PDF-Dateien, z.B. SumataraPDF, oft als Plugin im Browser.

Performance

Geschwindigkeit einer Verarbeitung.

Plugin

Kleine Unterfenster in einem Modul mit eigener Verarbeitungslogik, z.B. Wiedervorlageliste, anstehende Termine oder Aufgabenliste.

Popup

plötzlich erscheinendes Fenster.

Query

Abfrage.

Read-Only-Zugriff

Rein lesender Zugriff ohne Änderungsmöglichkeiten.

Redundanz

Darstellung von Information mit mehr Daten als nötig. Dient der Performancesteigerung und Widerstandsfähigkeit gegen Datenverlust.

Refresh

Anzeige aktualisieren.

Rendering

Prozess der Erzeugung der grafischen Darstellung einer Bildschirmseite oder eines PDF-Dokumentes.

Restore

Datenwiederherstellung.

Scope

Gültigkeitsbereich.

Session

Sitzung, private Datenumgebung im Speicher, erzeugt beim Anmelden eines Benutzers, gegen andere Sessions abgeschottet, wird beim Abmelden oder nach zu langer Inaktivität wieder gelöscht.

Session-ID

eindeutige, nicht erratbare Identifikation einer Session. Wird von der Application zu Beginn einer Session erzeugt, zwischen Client und Server immer mitgeschickt und zur Zuordnung von Requests zu einer Session benötigt.

Session-Hijacking

Verwendung einer fremden Session mit einer ausgespähten Session-ID.

Session-Timeout

Ende einer Session durch zu lange Untätigkeit.

Shell

Befehlsinterpreter.

Shortcut

Tastenkombination, Tastaturkürzel.

Shutdown

Herunterfahren oder Ausschalten des PCs.

SQL Insertion

Angriffsmethode, bei der SQL in Textfelder geschrieben wird.

standalone

für sich alleine stehend.

tag

Stichwort, das zur Klassifizierung an ein Objekt angehängt wird.

Timestamp

Zeitstempel, ordnet einem Ereignis einen eindeutigen Zeitpunkt zu.

Thumbnail

Vorschaubild.

Tool-Tip

kurze, gelbe Hilfe, die zu einem Oberflächenelement aufpoppt, wenn der Mauszeiger darauf platziert wird.

User

Benutzer.

Validierung

Prüfung von Benutzereingaben.

Workflow

Arbeitsablauf.

13.3 GFDL

GNU Free Documentation License

Version 1.2, November 2002

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.