NOTES & DOMINO ADMINCAMP

assono

AdminCamp 2010
Track 3, Session 6:
Notes-Entwicklung für Domino-Admins

Gelsenkirchen, 21. September 2010

Innovative Software-Lösungen.

www.assono.de

Thomas Bahn

Diplom-Mathematiker, Universität Hannover

seit 1997 entwickle ich mit Java und relationalen Datenbanken

seit 1999 mit Notes/Domino zu tun: Entwicklung, Administration, Beratung und Schulungen

regelmäßiger Sprecher auf nationalen und internationalen Fachkonferenzen zu IBM Lotus Notes/ Domino und Autor für THE VIEW

tbahn@assono.de
http://www.assono.de/blog
04307/900-401

assono
IT-Consulting & Solutions

Themen

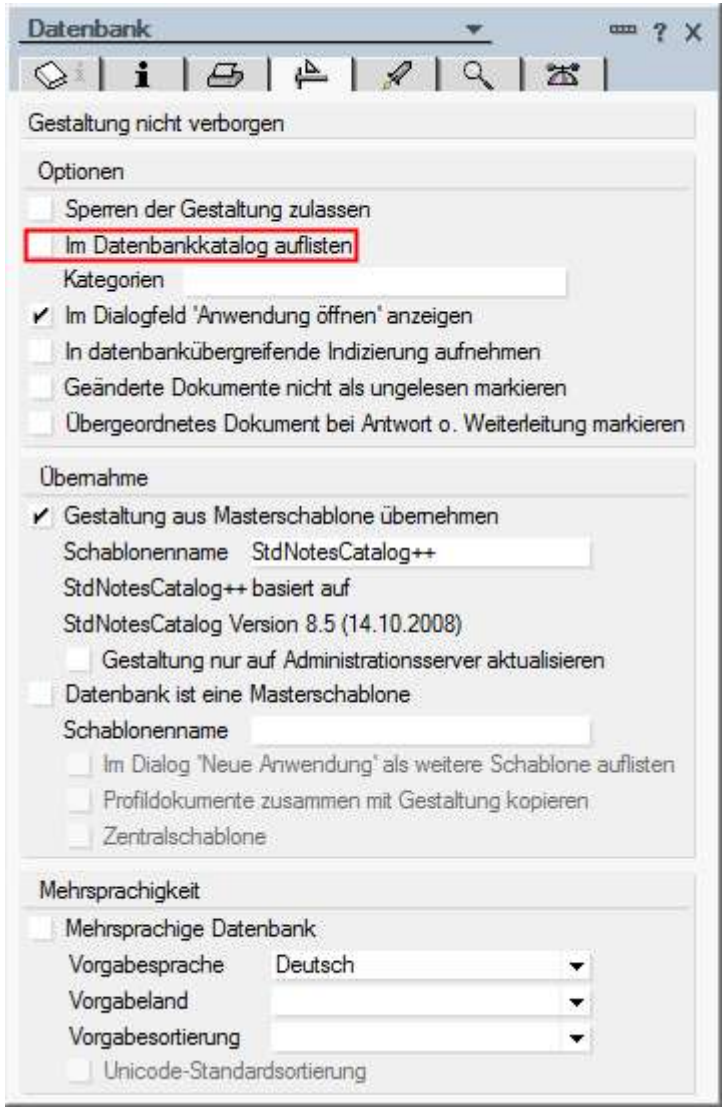
- vorhandene Ansichten anpassen
- neue Ansichten erstellen
- Felder zu Masken hinzufügen
- Wie überleben Änderungen Schablonen-Updates?
- Formelsprache
- Symbolleisten und Schaltflächen
- Agenten
- LotusScript
- weitere Programmiersprachen

Domino Designer

- muss extra installiert werden (früher gab es Tricks, ganz früher integriert im Notes-Client)
- kostenlos (zumindest eine Art von „kostenlos“)
- starten vom Betriebssystem aus, im Notes-Client aus den Bookmarks, Rechtsklick auf Datenbank-Reiter oder Datenbank-Icon, View – Design oder...
- DDE – Domino Designer on Eclipse
- langsamer als früher
- aber: einfach genial, erweiterbar, zukunftsorientiert
- Voraussetzung für XPages

Katalog und Domänen-Katalog

- catalog.nsf
- verschiedene Ansichten
- aber da fehlen doch Datenbanken...



Eine Datenbank im Domino Designer

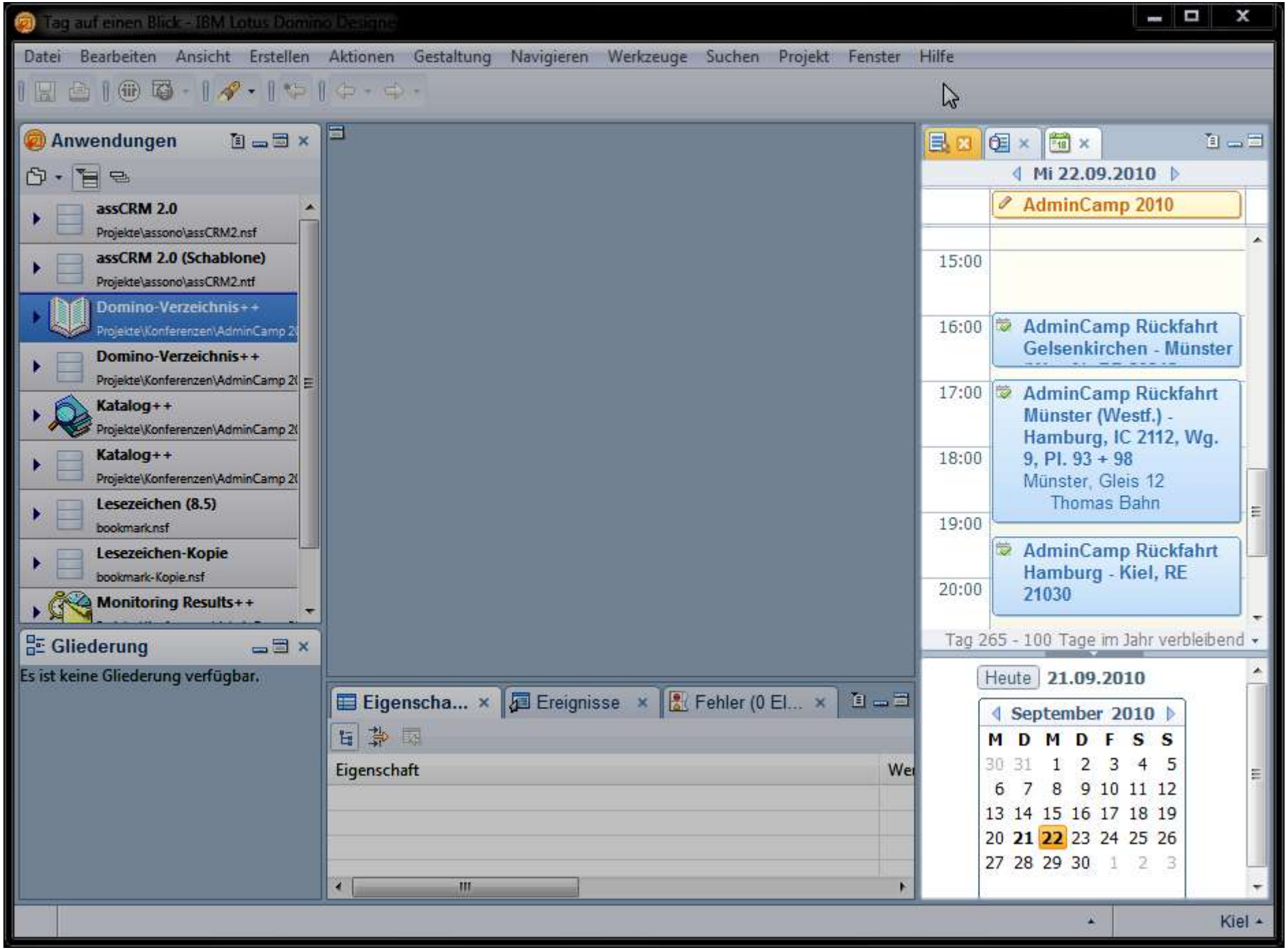
- links, rechts, oben, unten... es gibt viel zu sehen
- links: der Navigator mit der Liste der Datenbanken und in jeder Datenbank, die kategorisierte Liste aller Gestaltungselemente
- unten: Eigenschaften, Probleme und mehr...
- rechts: momentan nichts (in XPages wichtig)
- Mitte: der aktuelle Editor (kann auch eine Liste sein)

Navigieren im Domino Designer

- Doppelklick auf Reiter vergrößert Bereich
- und verkleinert auch wieder
- Reiter kann man auch ziehen und fallen lassen
- ein Bereich heißt „View“
- eine Kombination von Views heißt Perspektive
- es gibt mehrere Perspektiven und weitere Views...
- Doppelklick auf Kategorie von Gestaltungselementen öffnet entsprechende Liste
- Doppelklick auf Gestaltungselement öffnet es im Editor

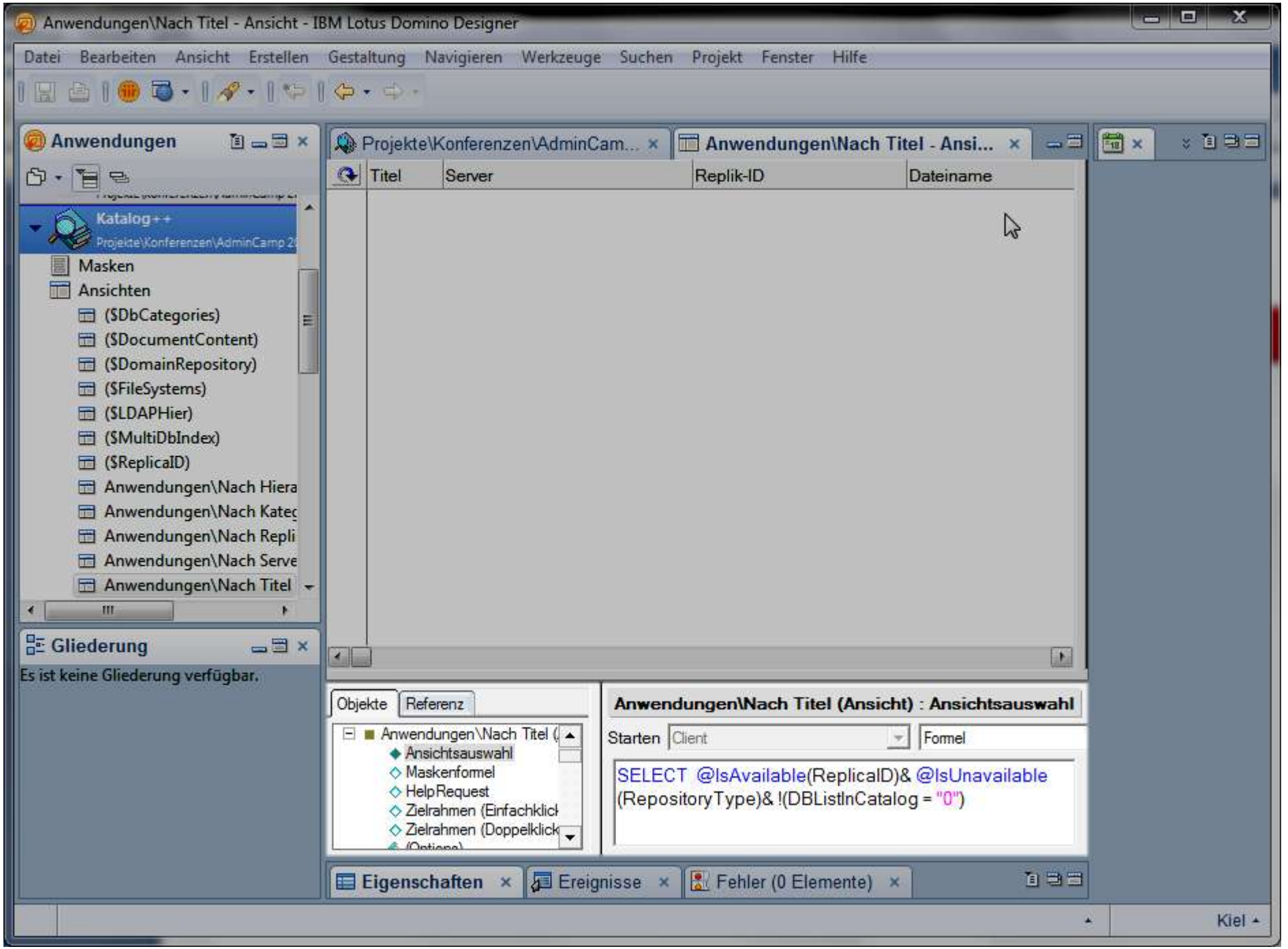
Eclipse-Ansicht öffnen

- „Tag auf einen Blick“ auch im Designer:
  - Fenster – Eclipse-Ansichten anzeigen – Andere...
  - Hannover-Kalender – Tag auf einen Blick


Tag auf einen Blick - IBM Lotus Domino Designer
Datei Bearbeiten Ansicht Erstellen Aktionen Gestaltung Navigieren Werkzeuge Suchen Projekt Fenster Hilfe
Anwendungen
assCRM 2.0
Projekte\assono\assCRM2.nsf
assCRM 2.0 (Schablone)
Projekte\assono\assCRM2.ntf
Domino-Verzeichnis++
Projekte\Konferenzen\AdminCamp 2
Domino-Verzeichnis++
Projekte\Konferenzen\AdminCamp 2
Katalog++
Projekte\Konferenzen\AdminCamp 2
Katalog++
Projekte\Konferenzen\AdminCamp 2
Lesezeichen (8.5)
bookmark.nsf
Lesezeichen-Kopie
bookmark-Kopie.nsf
Monitoring Results++
Gliederung
Es ist keine Gliederung verfügbar.
Eigenscha...
Ereignisse
Fehler (0 El...
Eigenschaft
Mi 22.09.2010
AdminCamp 2010
15:00
16:00
AdminCamp Rückfahrt Gelsenkirchen - Münster
17:00
AdminCamp Rückfahrt Münster (Westf.) - Hamburg, IC 2112, Wg. 9, Pl. 93 + 98
Münster, Gleis 12
Thomas Bahn
18:00
19:00
AdminCamp Rückfahrt Hamburg - Kiel, RE 21030
20:00
Tag 265 - 100 Tage im Jahr verbleibend
Heute
21.09.2010
September 2010
M D M D F S S
30 31 1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 1 2 3
Kiel

Eine Ansicht öffnen

- nehmen wir mal „Anwendungen\Nach Titel“
- öffnen im Domino Designer
  - über den Reiter,
  - im Menü oder
  - Designer öffnen, dann Datenbank öffnen
- Was ist wo...
- Formel für Ansichtsauswahl unten links auswählen und unten rechts ansehen


Anwendungen\Nach Titel - Ansicht - IBM Lotus Domino Designer
Datei Bearbeiten Ansicht Erstellen Gestaltung Navigieren Werkzeuge Suchen Projekt Fenster Hilfe
Anwendungen
Katalog++
Masken
Ansichten
($DbCategories)
($DocumentContent)
($DomainRepository)
($FileSystems)
($LDAPHier)
($MultiDbIndex)
($ReplicaID)
Anwendungen\Nach Hiera
Anwendungen\Nach Kateg
Anwendungen\Nach Repli
Anwendungen\Nach Serve
Anwendungen\Nach Titel
Gliederung
Es ist keine Gliederung verfügbar.
Projekte\Konferenzen\AdminCam...
Anwendungen\Nach Titel - Ansi...
Titel
Server
Replik-ID
Dateiname
Objekte
Referenz
Anwendungen\Nach Titel (
Ansichtsauswahl
Maskenformel
HelpRequest
Zielrahmen (Einfachklick
Zielrahmen (Doppelklick
Anwendungen\Nach Titel (Ansicht) : Ansichtsauswahl
Starten
Client
Formel
SELECT @IsAvailable(ReplicaID)& @IsUnavailable
(RepositoryType)& !(DBListInCatalog = "0")
Eigenschaften
Ereignisse
Fehler (0 Elemente)
Kiel

Formelsprache – die erste Berührung

- Ansichtsauswahl-Formel bestimmt, welche Dokumente dargestellt werden
- Spalten bestimmen, welche Daten aus dem Dokument dargestellt werden
  - Spezielle Funktionen,
  - Felder (genauer: Items) oder
  - Formeln

Und wie kann ich jetzt alle DBen sichtbar machen?

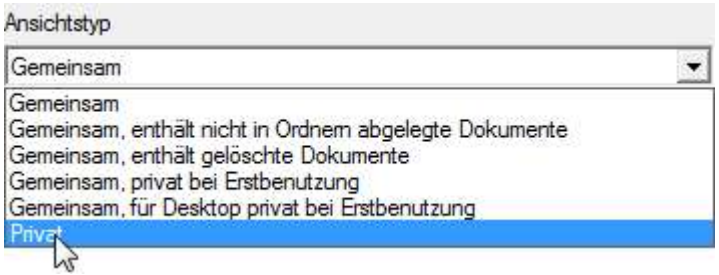
- Auswahlformel anpassen: Einschränkung entfernen
- SELECT wählt aus
- Boolesche Operatoren in der Formelsprache
  - ! Negation
    aus wahr mache falsch und anders herum
  - & logisches Und
    nur wahr, wenn beide Seiten wahr sind
  - | logisches Oder
    wahr, wenn mindestens eine Seite wahr ist

Testen

- im Notes-Client
- F9 = Aktualisieren (wie im Client)
- manchmal „klemmt“ F9,
  dann hilft nur Schließen und Öffnen der Ansicht
- selten: nur noch Replikationskonflikte...
  auch hier hilft Schließen und Öffnen

Ansichten

- Arten von Ansichten (und Ordnern)
  - privat
  - gemeinsam
  - gemeinsam, bei erster Benutzung privat
  - ...



- privat geht immer, mehr hängt von der ACL ab

Schutz gegen Gestaltungsaktualsierungen

- Aufgabe der Gestaltungsaktualisierungen ist das Ersetzen von Gestaltungselementen aus der Schablone
- Eigene Änderungen werden dabei überschrieben.
- Man kann sich dagegen schützen.
- kein (automatischer) Weg zurück
- Deshalb ist es manchmal besser, neue Elemente zu erstellen, als vorhandene zu ändern.
- automatischer Schutz, wenn Datenbank eine Schablone hat

Berechtigungen auf Ansichten

- Wer welche Dokumente sieht, liegt an der ACL und den Dokumenten (Leser- und Bearbeiter-Feldern).
- Wer welche Ansicht (oder welchen Ordner) sieht, liegt (fast) nur an ihren Berechtigungseinstellungen.
- Vorsicht bei vorhandenen Gestaltungselementen!

Best Practice: Berechtigungen nur über Rollen

- Rolle erstellen
- Rolle immer allen (!) Admin- und Server-Gruppen zuweisen
- Rolle in Ansichtenberechtigung benutzen
- kein „harter“ Schutz, da neue, notfalls private Ansichten erstellt werden können

Neue Ansicht als Kopie anlegen

- im Notes-Client: Erstellen – Ansicht...
  oder im Designer: Neue Ansicht
- Name eingeben
- Kopieren von: Ansicht auswählen
- Speichern und Bearbeiten
- Anpassen
- Berechtigungen setzen
- Speichern

Monitoring Results

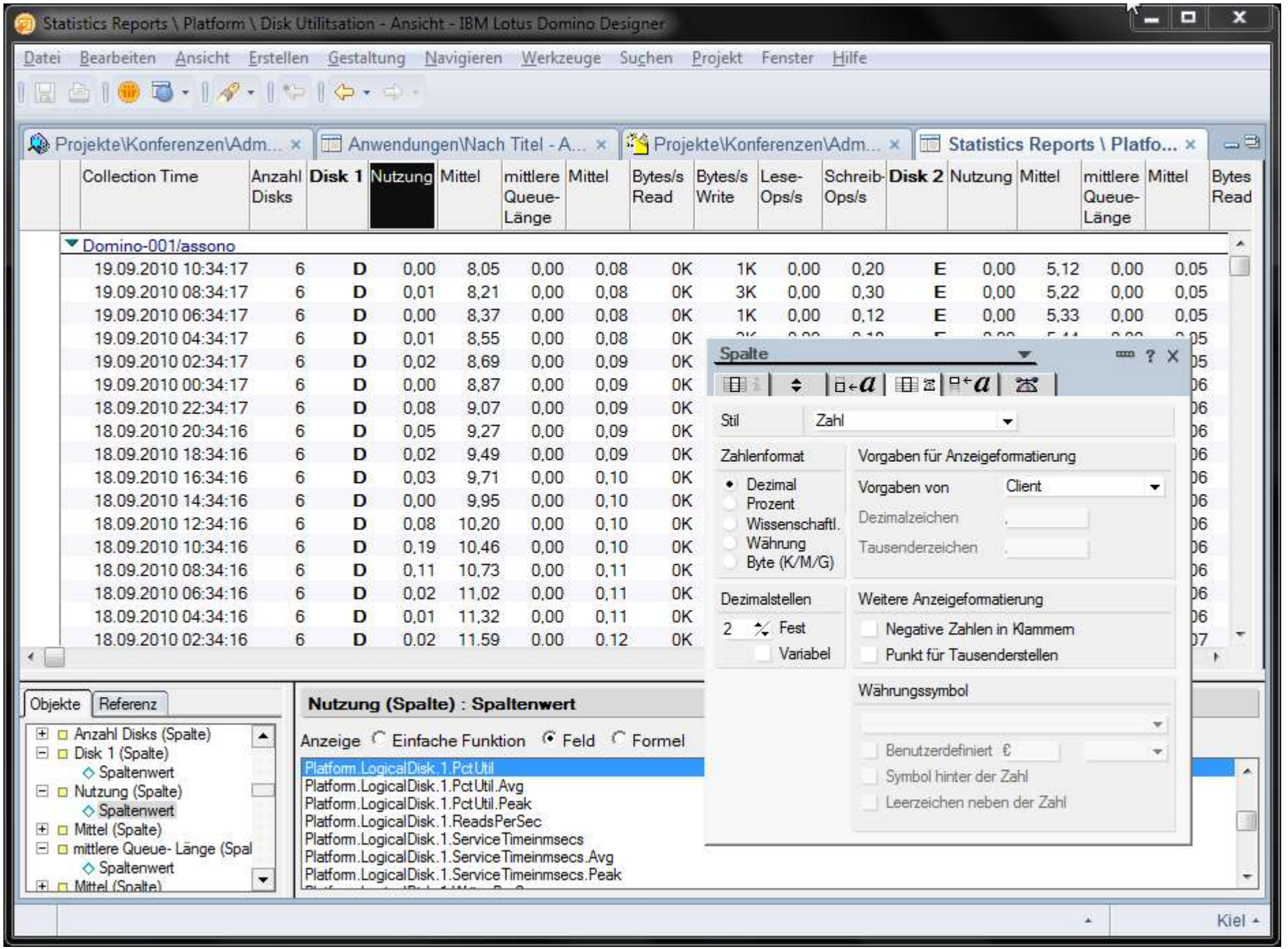
- aus statrep5.ntf erzeugte Datenbank
- wird automatisch gefüllt, wenn in Monitoring Configuration (events4.nsf) konfiguriert
- sollte man eigentlich immer machen
- So hat man immer historische Entwicklung parat, falls man mal Probleme hat oder man zukünftige Entwicklung abschätzen möchte.
- Besonders wertvoll: Plattform-Statistiken

Mir fehlt da was...

- Ansichten in Monitoring Results sind ein guter Startpunkt, aber grundsätzlich fehlt immer etwas.
- Do it yourself:
  - Spalten ergänzen
  - neue Ansichten bauen

Spalten ergänzen

- Vorhandene Ansicht öffnen
- Neue Spalte..., Doppelklick oder Copy and Paste
- das richtige Feld suchen oder
  Einfache Funktion auswählen oder
  Formel eingeben
- Spalteneigenschaften:
  - Breite, Twisties usw.
  - Sortierung und Kategorisierung
  - Formatierung (sparsam & einheitlich verwenden!)


Statistics Reports \ Platform \ Disk Utilitsation - Ansicht - IBM Lotus Domino Designer
Datei Bearbeiten Ansicht Erstellen Gestaltung Navigieren Werkzeuge Suchen Projekt Fenster Hilfe
Projekte\Konferenzen\Adm...
Anwendungen\Nach Titel - A...
Projekte\Konferenzen\Adm...
Statistics Reports \ Platfo...
Collection Time | Anzahl Disks | Disk 1 | Nutzung | Mittel | mittlere Queue-Länge | Mittel | Bytes/s Read | Bytes/s Write | Lese-Ops/s | Schreib-Ops/s | Disk 2 | Nutzung | Mittel | mittlere Queue-Länge | Mittel | Bytes Read
Domino-001/assono
19.09.2010 10:34:17 6 D 0,00 8,05 0,00 0,08 0K 1K 0,00 0,20 E 0,00 5,12 0,00 0,05
19.09.2010 08:34:17 6 D 0,01 8,21 0,00 0,08 0K 3K 0,00 0,30 E 0,00 5,22 0,00 0,05
19.09.2010 06:34:17 6 D 0,00 8,37 0,00 0,08 0K 1K 0,00 0,12 E 0,00 5,33 0,00 0,05
19.09.2010 04:34:17 6 D 0,01 8,55 0,00 0,08 0K
19.09.2010 02:34:17 6 D 0,02 8,69 0,00 0,09 0K
19.09.2010 00:34:17 6 D 0,00 8,87 0,00 0,09 0K
18.09.2010 22:34:17 6 D 0,08 9,07 0,00 0,09 0K
18.09.2010 20:34:16 6 D 0,05 9,27 0,00 0,09 0K
18.09.2010 18:34:16 6 D 0,02 9,49 0,00 0,09 0K
18.09.2010 16:34:16 6 D 0,03 9,71 0,00 0,10 0K
18.09.2010 14:34:16 6 D 0,00 9,95 0,00 0,10 0K
18.09.2010 12:34:16 6 D 0,08 10,20 0,00 0,10 0K
18.09.2010 10:34:16 6 D 0,19 10,46 0,00 0,10 0K
18.09.2010 08:34:16 6 D 0,11 10,73 0,00 0,11 0K
18.09.2010 06:34:16 6 D 0,02 11,02 0,00 0,11 0K
18.09.2010 04:34:16 6 D 0,01 11,32 0,00 0,11 0K
18.09.2010 02:34:16 6 D 0.02 11.59 0.00 0.12 0K
Spalte
Stil Zahl
Zahlenformat
Dezimal
Prozent
Wissenschaftl.
Währung
Byte (K/M/G)
Vorgaben für Anzeigeformatierung
Vorgaben von Client
Dezimalzeichen
Tausenderzeichen
Dezimalstellen
2 Fest
Variabel
Weitere Anzeigeformatierung
Negative Zahlen in Klammern
Punkt für Tausenderstellen
Währungssymbol
Benutzerdefiniert €
Symbol hinter der Zahl
Leerzeichen neben der Zahl
Objekte Referenz
Anzahl Disks (Spalte)
Disk 1 (Spalte)
Spaltenwert
Nutzung (Spalte)
Spaltenwert
Mittel (Spalte)
mittlere Queue- Länge (Spal
Spaltenwert
Mittel (Spalte)
Nutzung (Spalte) : Spaltenwert
Anzeige Einfache Funktion Feld Formel
Platform.LogicalDisk.1.PctUtil
Platform.LogicalDisk.1.PctUtil.Avg
Platform.LogicalDisk.1.PctUtil.Peak
Platform.LogicalDisk.1.ReadsPerSec
Platform.LogicalDisk.1.ServiceTimeinmsecs
Platform.LogicalDisk.1.ServiceTimeinmsecs.Avg
Platform.LogicalDisk.1.ServiceTimeinmsecs.Peak
Kiel

Wir brauchen die Schuhgröße

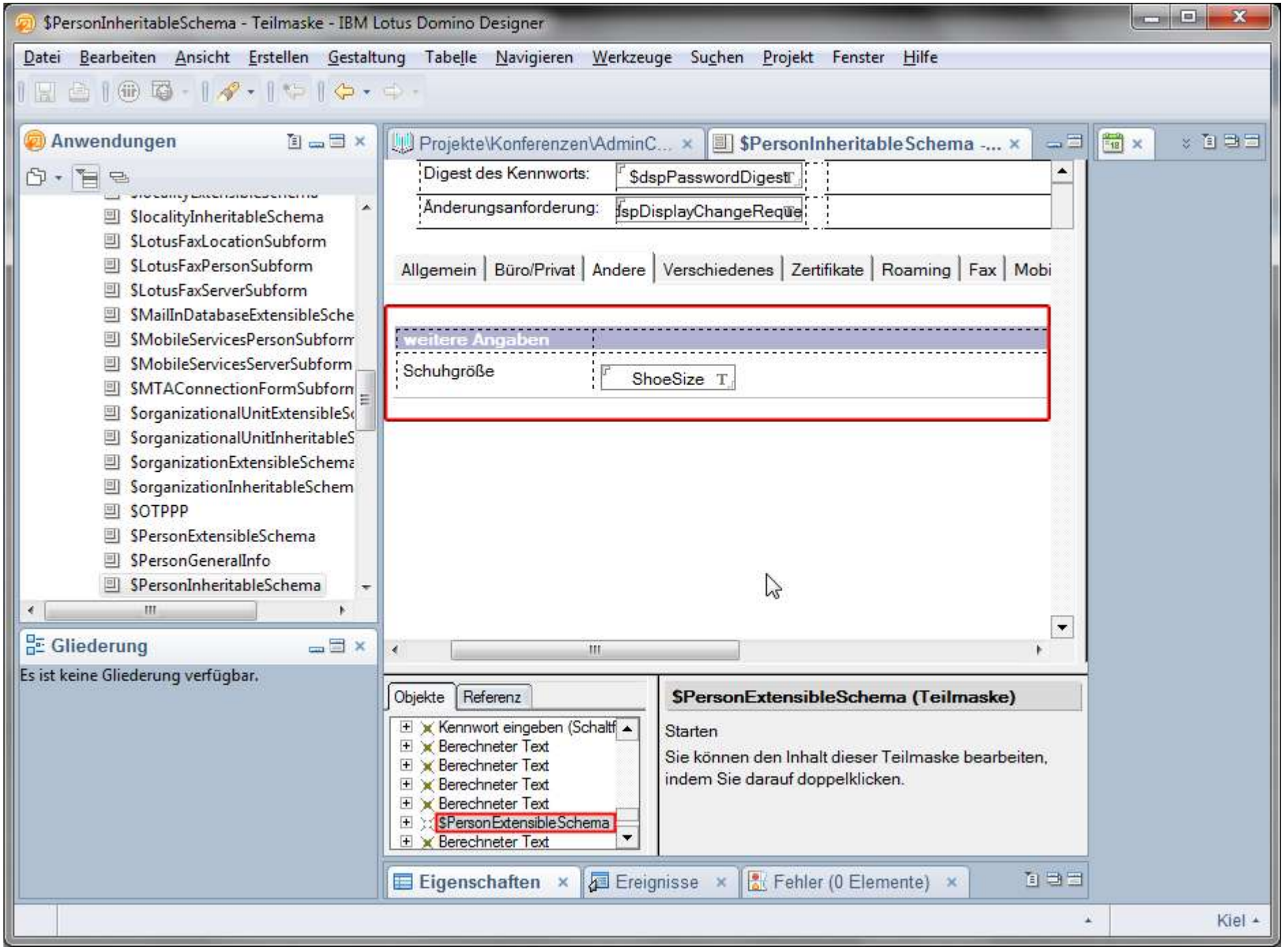
- das Domino-Verzeichnis anpassen
- zusätzliche Gestaltungselemente selten ein Problem
- Änderungen vorhandener Elemente: Was tun beim nächsten Update?
  - auf Aktualisierungen verzichten
  - eigene Änderungen immer wieder machen
- Domino-Verzeichnis wird immer angepasst
- häufig benötigte Anpassung: neue Felder in der Personen-Maske
- z. B. die Schuhgröße 

Masken und Teilmasken

- Maske Person öffnen
- Feld anklicken – geht nicht?!
- Teilmasken fassen Bereiche zusammen, z. B. zur einfacheren Wiederverwendung
- Doppelklick öffnet Teilmaske
- oder Liste der Teilmasken verwenden
- Es gibt auch berechnete Teilmasken: Erst beim Öffnen eines Dokuments wird ein Teilmasken-Name berechnet und die Teilmaske eingefügt.

Eine Teilmaske nur für Erweiterungen

- Teilmaske $PersonExtensibleSchema
- speziell da für Erweiterungen der Personen-Maske
- schützen gegen Gestaltungsaktualisierungen
- wird in neuen Versionen nie (!) geändert
- Personen-Maske selbst bleibt ungeschützt und wird immer aktualisiert
- dann ergänzen wir mal die Schuhgröße...


$PersonInheritableSchema - Teilmaske - IBM Lotus Domino Designer
Datei Bearbeiten Ansicht Erstellen Gestaltung Tabelle Navigieren Werkzeuge Suchen Projekt Fenster Hilfe
Anwendungen
$LocalityInheritableSchema
$LotusFaxLocationSubform
$LotusFaxPersonSubform
$LotusFaxServerSubform
$MailInDatabaseExtensibleSche
$MobileServicesPersonSubform
$MobileServicesServerSubform
$MTAConnectionFormSubform
$OrganizationalUnitExtensibleS
$OrganizationalUnitInheritableS
$OrganizationExtensibleSchema
$OrganizationInheritableSchem
$OTPPP
$PersonExtensibleSchema
$PersonGeneralInfo
$PersonInheritableSchema
Gliederung
Es ist keine Gliederung verfügbar.
Projekte\Konferenzen\AdminC...
$PersonInheritableSchema -...
Digest des Kennworts:
$dspPasswordDigest
Änderungsanforderung:
dspDisplayChangeReque
Allgemein Büro/Privat Andere Verschiedenes Zertifikate Roaming Fax Mobi
weitere Angaben
Schuhgröße
ShoeSize
Objekte Referenz
Kennwort eingeben (Schaltf
Berechneter Text
Berechneter Text
Berechneter Text
Berechneter Text
$PersonExtensibleSchema
Berechneter Text
$PersonExtensibleSchema (Teilmaske)
Starten
Sie können den Inhalt dieser Teilmaske bearbeiten, indem Sie darauf doppelklicken.
Eigenschaften
Ereignisse
Fehler (0 Elemente)
Kiel

Formelsprache, jetzt aber richtig

- Formelsprache ähnlich den Microsoft Excel-Formeln
- sehr, sehr lange Liste an Funktionen und Befehlen
- mit jeder neuen Version kommen weitere dazu
- die Designer-Hilfe ist dein Freund 
- generell: Semikolon zwischen Anweisungen und zwischen Argumenten bei Funktionsaufruf
- Groß-/Kleinschreibung ist egal

Literale

- Strings:
  "abc" {xyz}
  Sonderzeichen mit vorangestellten Backslash \
  z. B. "Verträge\\nach Vertragsnummer"

- Zahlen:
  1 -2.0 0.2E-4
  unabhängig von Konfiguration immer Dezimal**punkte**

- Zeit-/Datumswerte:
  [29.9.2010] [15:00] [29.9.2010 15:00]

- Listen:
  "a" : "b" : "c" 1 : 2.0 : -1.0E3
  mehrere Werte gleichen Typs mit Doppelpunkten

Operatoren

- die üblichen Verdächtigen:
- Rechnen mit *, /, +, - und Klammern
- Vergleichen mit <, <=, =, >=, > und != (ungleich)
- Logische Verknüpfungen:
  ! nicht & logisches Und | logisches Oder
- Zuweisungen mit :=
- Strings zusammenfügen mit +
- Listenverkettung mit :
  Zugriff auf Element mit Listenname[index]

Reservierte Worte

- SELECT Ausdruck
  speziell für Ansichtsauswahl und in Agenten
- REM "Kommentar" oder REM {Kommentar}
- FIELD Name := Wert
  (Zuweisungen mit :=, Vergleiche mit =)
- DEFAULT Name := Wert
  wie FIELD, aber wenn es Feld schon gibt, behält es seinen Wert
- ENVIRONMENT Name := Wert
  schreibt Name-Wert-Paar in die notes.ini

Funktionen

- meistens auf das aktuell geöffnete Dokument beschränkt (mit wichtigen Ausnahmen)
- meistens keine „Nebeneffekte“
- Aufruf: @Funktionsname(arg_1; arg_2; ...; arg_n)
- „Keyword“-Argumente in eckigen Klammern, z.B. @Prompt([OK]; "Überschrift"; "Nachricht")

Befehle

- Befehle entsprechen meistens Aktionen in der Benutzeroberfläche und beziehen sich nicht auf ein Dokument.
- Befehlsname entsprechend zusammengesetzt: FileSave, FileClose, EditDocument, OpenView, ...
- Format:
  @Command([Befehl]; arg_1; arg_2; ...)
  @PostedCommand([Befehl]; arg_1; arg_2; ...)
- @PostedCommands werden immer gesammelt am Schluss ausgeführt, @Commands häufig auch
- Seit Version 6 gibt es aber auch einige Befehle, die sofort ausgeführt werden

Programmablauf

- bedingte Ausführung:

```
@If(
  Bedingung_1;
  Anweisung_1;
  Bedingung_2;
  Anweisung_2;
  ...
  Sonst_Anweisung
)
```

- Minimalform:

```
@If(Bedingung_1; Anweisung_1; Sonst_Anweisung)
```

- @If ist Funktion und gibt immer einen Wert zurück.

Programmablauf (forts.)

- Code-Block:

```
@Do(
    Anweisung_1;
    Anweisung_2;
    ...;
    Anweisung_n
)
```

- @Do ist eine Funktion und gibt den Rückgabewert der letzten Anweisung_n als Ergebnis zurück.
- Überall, wo eine Anweisung erlaubt ist, kann man auch einen @Do-Block verwenden.

Programmablauf (forts.)

- Vorzeitiges Ende der aktuellen Formel:
  @Return(Ausdruck)
- Normalerweise wird eine Formel bis zur letzten Anweisung komplett ausgeführt.
- @Return verlässt die aktuelle Formel und gibt den Wert des Ausdrucks zurück.

Programmablauf (forts.)

- bis Version 5 von Notes/Domino gab es nicht mehr
- seit Version 6 gibt es endlich Schleifen

Schleifen

- @For-Schleife:

```
@For(
    Initialisierung;
    Bedingung;
    Erhöhung;
    Anweisung_1;
    Anweisung_2;
    Anweisung_3;
    ...
)
```

Schleifen (forts.)

- Beispiel:

```
@For(
        n := 1;
        n <= @Elements(Categories);
        n := n + 1;
        @Prompt([OK]; @Text(n); Categories[n])
)
```

- Wenn es im aktuellen Dokument ein (Mehrfachwert-) Item Categories gibt, werden nacheinander alle Werte ausgegeben.

Schleifen (forts.)

- @DoWhile- und @While-Schleifen:

```
@DoWhile(
    Anweisung_1;
    Anweisung_2;
    ... ;
    Bedingung
)
```

```
@While(
    Bedingung;
    Anweisung_1;
    Anweisung_2;
    ...
)
```

Ein- und Ausgabe

- @StatusBar(Nachricht)

- @Prompt([OK]; Titel; Nachricht)
  @Prompt([YesNo]; ...)
  @Prompt([YesNoCancel]; ...)
  @Prompt([OkCancelEdit]; ...)
  @Prompt([OkCancelList]; ...)
  @Prompt([OkCancelCombo]; ...)
  @Prompt([OkCancelEditCombo]; ...)
  @Prompt([OkCancelListMulti]; ...)
  @Prompt([ChooseDatabase]; ...): Datenbank wählen
  @Prompt([LocalBrowse]; ...): Datei wählen
  @Prompt([Password]; ...)

Ein- und Ausgabe (forts.)

- @PickList([Name]; ...): Person(en) auswählen
  @PickList([Custom] ...): Dokument aus Ansicht wählen

- @DialogBox(...): (Teil-)Maske öffnen

- Werte aus notes.ini lesen mit
  @Environment(Name) (mit einem Parameter)
  und in die notes.ini schreiben mit:
  ENVIRONMENT Name := Wert;
  @Environment(Name; Wert); (mit zwei Parametern)
  @SetEnvironment(Name; Wert);

Wichtige Funktionen rund um Zahlen

- @Integer(Wert): ganzahliger Anteil der Zahl

- @Round(Wert; Faktor): gerundeter Wert, z. B.
  @Round(1.234; 0.1) ergibt 1.2
  @Round(1234.56; 100) ergibt 1200

- @Random: Zufallszahl zwischen 0 und 1

- @Modulo(Zahl_1; Zahl_2)

- @Min(Zahl_1; Zahl_2; ...)
  @Max(Zahl_1; Zahl_2; ...)
  @Sum(Zahl_1; Zahl_2; ...)
  funktioniert auch mit Zahlen-Listen

Wichtige Funktionen rund um Strings

- @Text(Wert): konvertiert in Text
  @ToNumber(Wert): konvertiert in Zahl
  @ToTime(Wert): konvertiert in Datum/Zeit

- @IsNumber(Wert)
  @IsTime(Wert)

- @Char(Zahl): Zeichen mit dieser Nummer in CP850
  @NewLine: „Konstante“ für Zeilenumbruch
  @Repeat(String, Anzahl)

- @Length(String): Anzahl Zeichen

- @UpperCase(String),
  @LowerCase(String)
  @Trim(String)

Wichtige Funktionen rund um Strings (forts.)

- @Contains(String; Teilstring)
  @Begins(String; Teilstring)
  @Ends(String; Teilstring)

- @Left(String; Anzahl) @Left(String; Zeichen)
  @LeftBack(String; Anzahl)
  @LeftBack(String; Zeichen)
  @Right(String; Anzahl) @Right(String; Zeichen)
  @RightBack(String; Anzahl)
  @RightBack(String; Zeichen)
  @Middle(String; Anzahl/Zeichen; Anzahl/Zeichen)
  @MiddleBack(String; Anzahl/Zeichen; Anz./Zeichen)

Wichtige Funktionen rund um Strings (forts.)

- @ReplaceSubstring(String; Von; Nach)
  auch mit Text-Listen
- @Replace(String-Liste; Von; Nach)
  wie vorher, aber ganzer Wert muss übereinstimmen
- @Word(String oder Liste; Trenner; Position)
  holt Wert an der gegebenen Position, z. B.
  @Word("abc~def~ghi~jkl"; "~"; 2) ergibt "def"
- @Like und @Matches: Mustervergleich

Wichtige Funktionen rund um Zeit und Datum

- @Now, @Today, @Tomorrow, @Yesterday

- @Date(y; m; d): Datum
  @Time(h; min; s): Zeit
  @Date(y; m; d; h; min; s): Zeit-/Datumswert

- @Date(ZeitDatum): Datumsanteil
  @Time(ZeitDatum): Zeitanteil
  @Year(ZeitDatum), @Month(...), @Day(...)
  @Hour(...), @Minute(...), @Second(...)
  @Weekday(...): 1 ist Sonntag bis 7 ist Samstag

- @Adjust(ZeitDatum; y; m; d; h; min; s)
  addiert y zum Jahr, m zum Monat usw.
  auch negative Werte; berücksichtigt Umbrüche, z. B.
  @Adjust(@Today; 0; 0; 1; 0; 0; 0) ergibt Morgen

Wichtige Funktionen rund um Listen

- @Elements(Liste): Anzahl der Werte in der Liste

- @IsMember(Liste; Wert)
  @IsNotMember(Liste; Wert)
  @Member(Liste; Wert): Position des Werts in der Liste

- @Keywords(Liste1; Liste2; Trenner):
  findet Worte in Liste1, die es auch in Liste2 gibt, z. B.

```
@Elements(@Keywords(
      @UserRoles;
      "[Admin]" : "[Entwickler]";
      ""
)) > 0
```

  ergibt wahr, wenn der aktuelle Benutzer mindestens eine der beiden Rollen zugeordnet bekommen hat.

Wichtige Funktionen rund um Listen (forts.)

- @Explode(String; Trenner): zerteilt String in eine Liste
  @Implode(Liste; Trenner): fügt Elemente zusammen zu einem String

- @Sort(Liste): sortiert Liste
  @Trim(Liste): entfernt leere Elemente
  @Unique(Liste): entfernt Duplikate

- @Subset(Liste; Anzahl): Teilliste mit Anzahl Elementen; ist Anzahl < 0, dann extrahiere Teilliste von hinten

Wichtige Funktionen rund um die aktuelle Umgebung

- @UserName: aktueller Benutzer (vollständig)
  @Name([Format]; NotesName): wandelt um, z. B.
  @Name([CN]; @UserName) ergibt "Thomas Bahn"

- @UserRoles: Liste der Rollen, die der Benutzer hat
  @UserNamesList: Liste der Rollen, Namen und Gruppen des aktuellen Benutzers
  @UserAccess(...): Was darf der Benutzer?

- @Version: aktuelle Notes-Version (Build-Nummer)

- @DbTitle: Titel der aktuellen Datenbank
  @ReplicaID: ihre Replik-ID
  @DbName: Liste aus Servername und Dateiname
  @ServerName: aktueller Server
  @ViewTitle: Namen der aktuellen Ansicht

Wichtige Funktionen rund um das aktuelle Dokument

- @NoteID: innerhalb der Replik eindeutige Note-ID
  @DocumentUniqueID: UNID des Dokuments

- @DocLength: Größe des Dokuments

- @Attachments: Anzahl Anhänge
  @AttachmentNames: ihre Dateinamen
  @AttachmentLengths: ihre Dateigrößen

- @IsNewDoc: neues, d. h. ungspeichertes Dokument?
  @IsDocBeingLoaded, @IsDocBeingEdited,
  @IsDocBeingRecalculated, @IsDocBeingSaved,
  @IsDocBeingMailed: gerade in dieser Phase?

- @IsAvailable(Feldname): gibt es das Feld?
  @IsUnavailable(Feldname): oder nicht?

Wichtige Funktionen rund um andere Dokumente

- @GetDocField(UNID; Feldname)
  @SetDocField(UNID; Feldname; Wert)

- @GetProfileField(Profilname; Feldname)
  @SetProfileField(Profilname; Feldname; Wert)

- @DbColumn("Notes" : "Cache"; Server : Datei; Ansicht; Spaltennummer)
  gibt alle Werte in einer Spalte einer Ansicht zurück, wobei
  Cache: "Cache", "NoCache" oder "ResetCache";
  Server : Datei: "" bedeutet aktuelle DB nehmen,
  statt Server : Datei geht auch die Replik-ID

Wichtige Funktionen rund um andere Dokumente (forts.)

- @DbLookup("Notes" : "Cache"; Server : Datei; Ansicht; Wert; Spaltennummer oder Item-Name): sucht in erster sortierter Spalte nach dem Wert und gibt von dem oder den passenden Dokumenten die Werte in der gegeben Spalte oder des angegeben Items zurück
- @DbColumn und @DbLookup funktionieren nicht in Ansichten, sind aber vorallem in Masken und Aktionen unverzichtbar.

Wichtige Formelsprachen-Befehle

- @Command([FileSave])
- @Command([FileCloseWindow])
- @Command([FileSave])
- @Command([Compose]; Server : DB; Form)
- @Command([MailComposeMemo])
- @Command([ToolsRefreshSelectedDocs])
- @Command([ToolsRunMacro]; Agent)
- @Command([WindowWorkspace])

Wichtige Formelsprachen-Befehle (forts.)

- einfach mal in der Designer-Hilfe blättern...

AdminCertify @Command
AdminCreateGroup @Command
AdminCrossCertifyIDFile @Command
AdminCrossCertifyKey @Command
AdminDatabaseAnalysis @Command
AdminDatabaseQuotas @Command
AdminIDFileClearPassword @Command
AdminIDFileExamine @Command
AdminIDFileSetPassword @Command
Administration @Command
AdminNewOrganization @Command
AdminNewOrgUnit @Command
AdminOpenAddressBook @Command
AdminOpenCatalog @Command
AdminOpenCertLog @Command
AdminOpenGroupsView @Command
AdminOpenServerLog @Command
AdminOpenServersView @Command
AdminOpenStatistics @Command
AdminOpenUsersView @Command
AdminOutgoingMail @Command
AdminRegisterFromFile @Command
AdminRegisterServer @Command
AdminRegisterUser @Command
AdminRemoteConsole @Command
AdminSendMailTrace @Command
AdminStatisticsConfig @Command
AdminTraceConnection @Command

FileDatabaseACL @Command
FileDatabaseCompact @Command
FileDatabaseCopy @Command
FileDatabaseDelete @Command
FileDatabaseInfo @Command
FileDatabaseRemove @Command
FileDatabaseUseServer @Command
FileExit @Command
FileExport @Command
FileFullTextCreate @Command
FileFullTextDelete @Command
FileFullTextInfo @Command
FileFullTextUpdate @Command
FileImport @Command

UserIDCertificates @Command
UserIDClearPassword @Command
UserIDCreateSafeCopy @Command
UserIDEncryptionKeys @Command
UserIDInfo @Command
UserIDMergeCopy @Command
UserIDSetPassword @Command
UserIDSwitch @Command

ToolsCall @Command
ToolsCategorize @Command
ToolsHangUp @Command
ToolsMarkAllRead @Command
ToolsMarkAllUnread @Command
ToolsMarkSelectedRead @Command
ToolsMarkSelectedUnread @Command
ToolsRefreshAllDocs @Command
ToolsRefreshSelectedDocs @Command
ToolsReplicate @Command
ToolsRunBackgroundMacros @Command
ToolsRunMacro @Command
ToolsScanUnreadChoose @Command
ToolsScanUnreadPreferred @Command
ToolsScanUnreadSelected @Command
ToolsSetupLocation @Command
ToolsSetupMail @Command
ToolsSetupPorts @Command
ToolsSetupUserSetup @Command
ToolsSmartIcons @Command
ToolsSpellCheck @Command
ToolsUserLogoff @Command

Fehlerbehandlung

- Syntaxfehler merkt man beim Speichern:
  es geht nicht... 
- @IsError(Ausdruck): wahr, wenn Ausdruck ein Fehler ist, normalerweise Rückgabewert einer Funktion, gespeichert in einer Variablen
- @Error: erzeugt Fehler
- @Failure(Grund) und @Success:
  nur als Ergebnis der Eingabevalidierung
- Wenn zur Laufzeit ein Fehler auftritt, kann es sein, dass die Ausführung der Formel einfach ohne weitere Meldung abbricht. 

Debugging

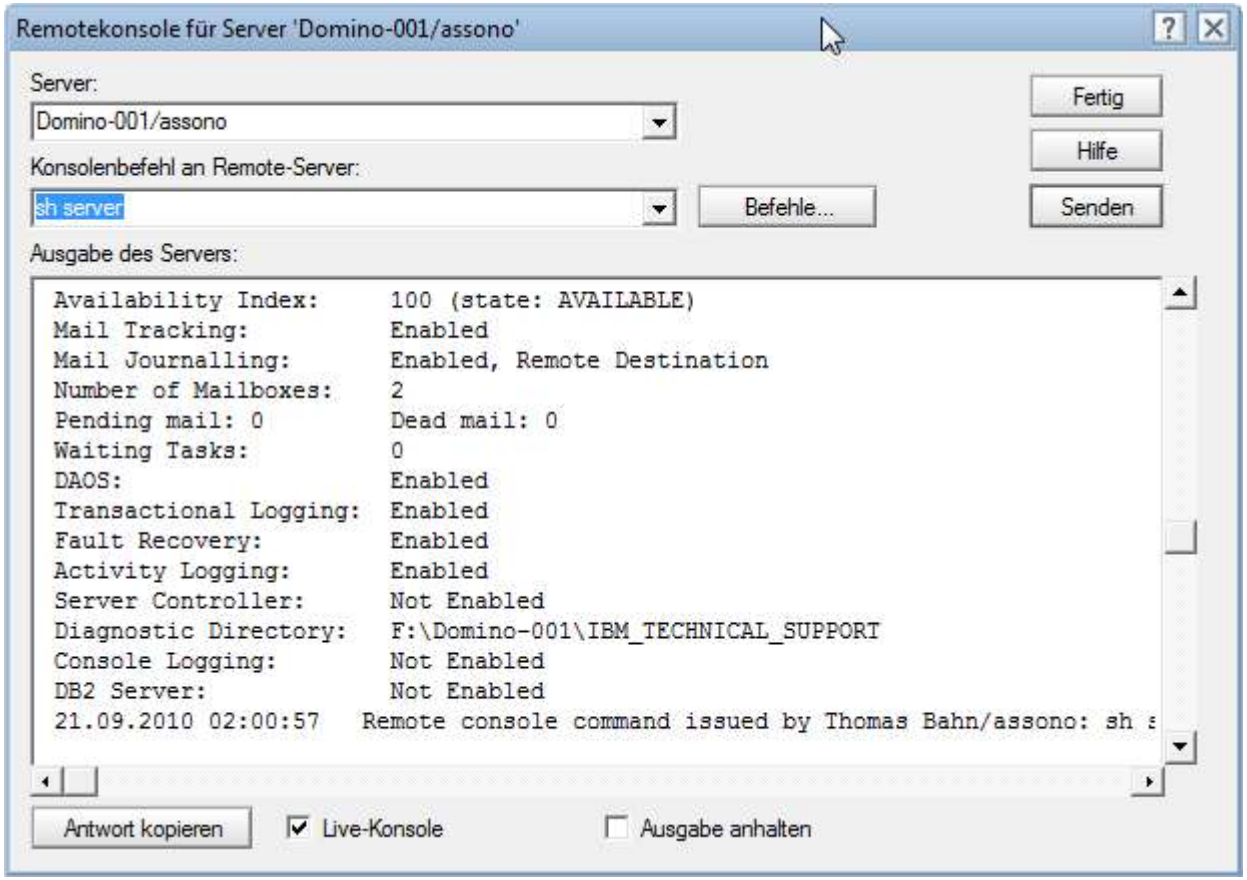
- @Prompt ist dein Freund

Debugging (forts.)

- Ernsthaft, das war's!
- Es gibt **keinen** Debugger für die Formelsprache.

Remote Server Console öffnen

- Wer kennt noch Notes-Server 4.x?
- Damals gab es noch keinen Domino-Administrator, sondern nur die Remove Server Console:

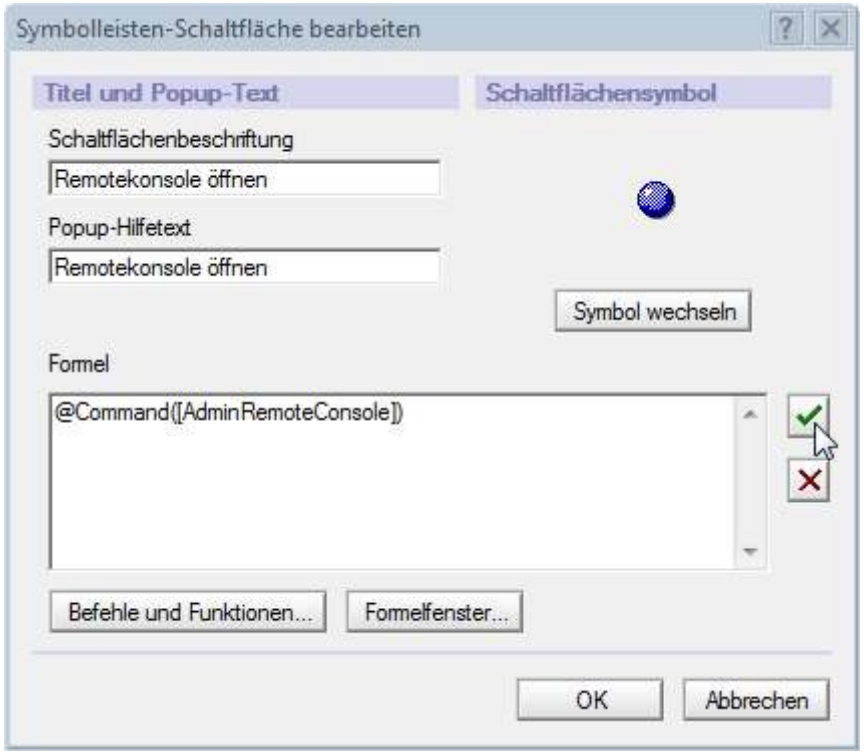


Remote Server Console öffnen (forts.)

- Die gibt es immer noch — nur ein wenig versteckt...
- Sie öffnet sich viel, viel schneller als der Admin-Client. Genial, wenn man nur mal schnell was starten oder nachsehen will.
- Zum Öffnen benötigt man den folgenden Befehl: @Command([AdminRemoteConsole])
- Um die Konsole jederzeit schnell mit einem Klick öffnen zu können, erstellen wir ein Toolbar-Button.

Remote Server Console öffnen (forts.)

- dazu: File – Preferences... – Toolbar – Customize
- vorhandene Toolbar auswählen oder neue erstellen
- New – Button...
- Überschrift eingeben
- Formel eingeben
  (oder auswählen)
- Icon ändern,
  wenn gewünscht
- 2x Ok und ausprobieren



Und was passiert hier?

```
_names := @LocationGetInfo([HomeServer]) :
   "names.nsf";

_servers := @PickList([Custom]; _names; "Servers";
   "Wähle Server"; "Wähle die Server"; 3);

_db := @Prompt([OkCancelEdit]; "Wähle Datenbank";
   "Gib den Dateinamen (inkl. Pfad) der Datenbank ein.";
   "log.nsf");

@For(
   n := 1;
   n <= @Elements(_servers);
   n := n + 1;
   @Command([AddDatabase]; _servers[n] : _db)
)
```

Arbeitsbereich für Administrator vorbereiten

- alle Protokoll-Datenbanken von allen Servern zu einem Arbeitsbereich hinzufügen
- Das kann dauern (bei größeren Domänen).
- danach alle Administrationsanforderungen (admin4.nsf), alle Monitoring Configuration (events4.nsf), alle Monitoring Results (statrep5.nsf), ...
- Gemeinsamkeit: jeweils gleicher Dateiname auf allen Servern
- und im Domino-Verzeichnis gibt es Liste aller Server...

Gewählte Dokumente aktualisieren

- Dokument aktualisieren:
  Öffnen, Bearbeiten, F9, Speichern, Schließen
- oder: alle gewünschten Dokumente in einer Ansicht markieren und einmal Agent aufrufen
- Benötigter Formelsprache-Befehl:
  @Command([ToolsRefreshSelectedDocs])

Lasst uns einen Agenten bauen...

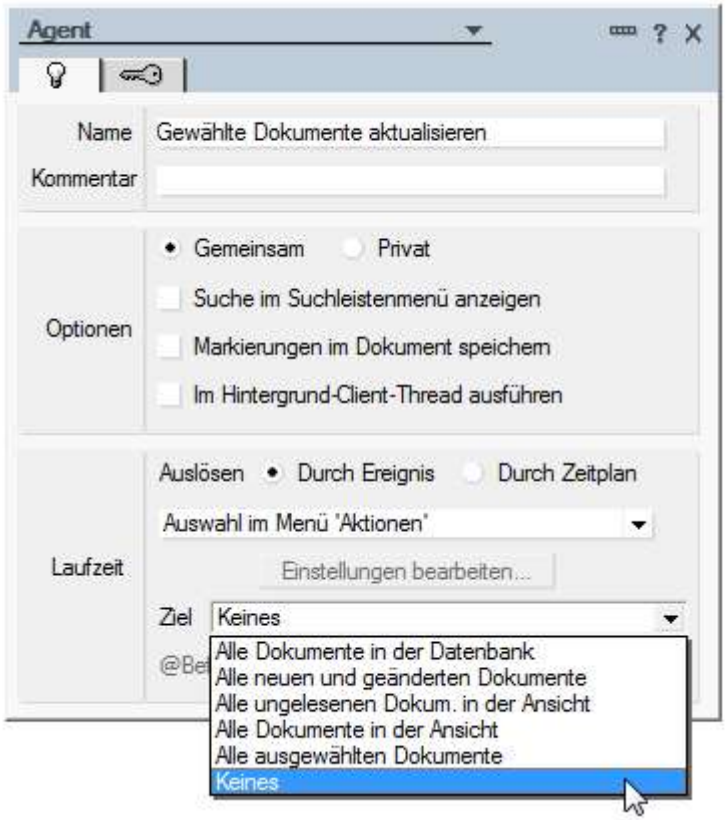
- in einer Datenbank im Notes-Client:
  Create – Agent oder View – Agents und New Agent
  oder gleich im Domino Designer
- im Eclipse-basiertem Designer muss man sich gleich für einen Typ entscheiden:
  - Einfache Aktionen,
  - Formelsprache,
  - LotusScript,
  - Java oder
  - importiertes Java

Lasst uns einen Agenten bauen... (forts.)

- Je nach Typ können Agenten verschiedene Auslöser haben:
  - zeitgesteuert oder
  - ereignisgesteuert, dann:
    - Aufruf aus dem Aktionen-Menü,
    - Aufruf aus der Agentenliste,
    - vor Eingang neuer E-Mail,
    - nach Eingang neuer E-Mail,
    - neues oder geändertes Dokument,
    - nach dem Einfügen von Dokumenten,
    - nach dem Server-Start

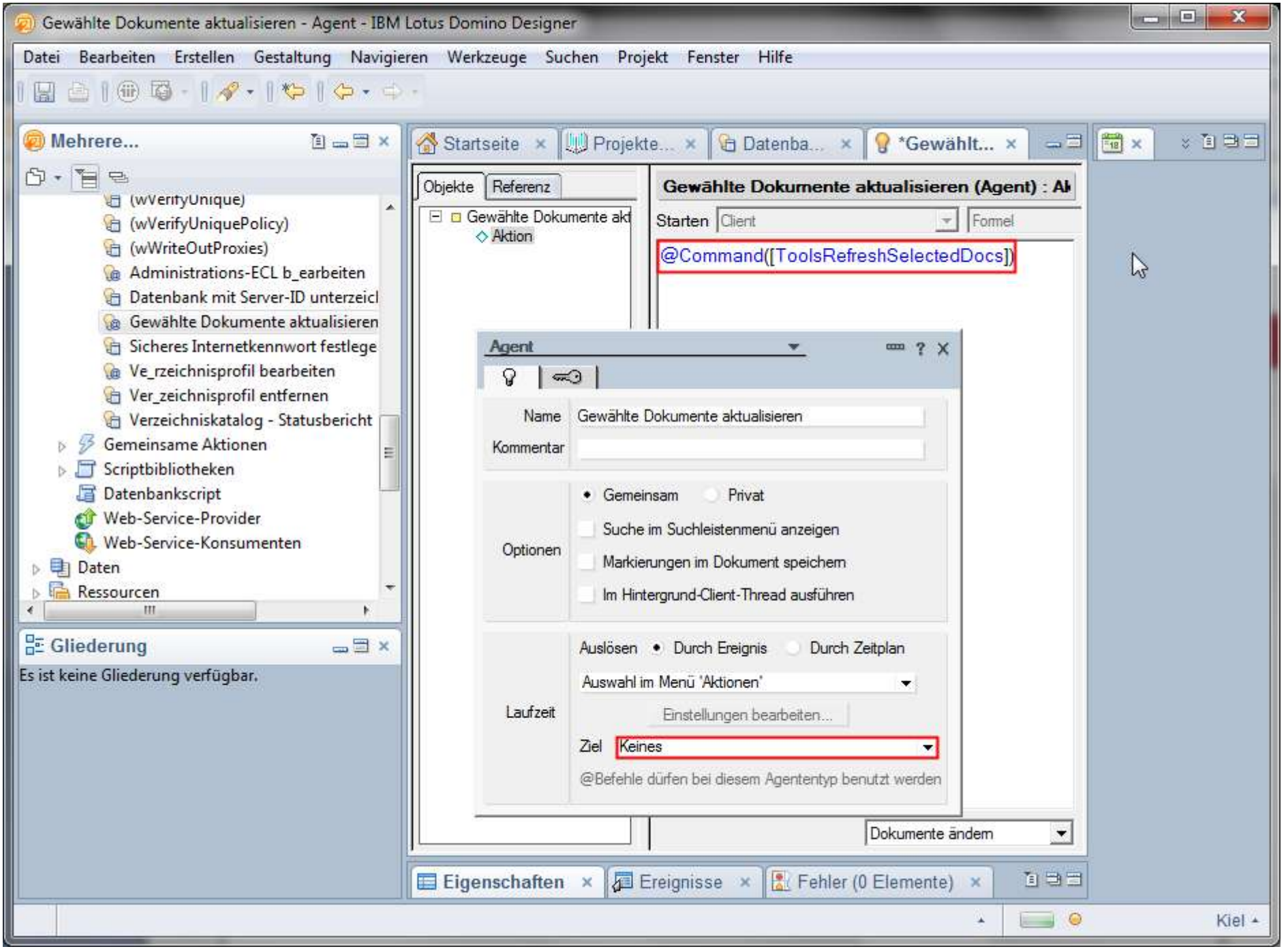
Lasst uns einen Agenten bauen... (forts.)

- Und je nach Auslöser gibt es verschiedene Ziele, auf denen der Agent laufen kann:
  - alle Dokumente in der Datenbank
  - neue und geänderte Dokumente
  - alle ungelesene Dokumente in der Ansicht
  - alle Dokumente in der Ansicht
  - alle ausgewählten Dokumente
  - keine Dokumente
- wir brauchen für unseren Agenten, der alle ausgewählten Dokumenten aktualisiert: ...



Lasst uns einen Agenten bauen... (forts.)

- wir brauchen für unseren Agenten, der alle ausgewählten Dokumenten aktualisiert: keine Dokumente?!
- sonst laufen keine Formelsprachen-Befehle
- der Rest ist einfach: Formel in das große Fenster eingeben oder auswählen
- Speichern, Schließen und Testen


Gewählte Dokumente aktualisieren - Agent - IBM Lotus Domino Designer
Datei Bearbeiten Erstellen Gestaltung Navigieren Werkzeuge Suchen Projekt Fenster Hilfe
Mehrere...
(wVerifyUnique)
(wVerifyUniquePolicy)
(wWriteOutProxies)
Administrations-ECL b_earbeiten
Datenbank mit Server-ID unterzeicl
Gewählte Dokumente aktualisieren
Sicheres Internetkennwort festlege
Ve_rzeichnisprofil bearbeiten
Ver_zeichnisprofil entfernen
Verzeichniskatalog - Statusbericht
Gemeinsame Aktionen
Scriptbibliotheken
Datenbankscript
Web-Service-Provider
Web-Service-Konsumenten
Daten
Ressourcen
Gliederung
Es ist keine Gliederung verfügbar.
Startseite
Projekte...
Datenba...
*Gewählt...
Objekte
Referenz
Gewählte Dokumente akt
Aktion
Gewählte Dokumente aktualisieren (Agent) : Al
Starten
Client
Formel
@Command([ToolsRefreshSelectedDocs])
Agent
Name
Gewählte Dokumente aktualisieren
Kommentar
Optionen
Gemeinsam
Privat
Suche im Suchleistenmenü anzeigen
Markierungen im Dokument speichern
Im Hintergrund-Client-Thread ausführen
Laufzeit
Auslösen
Durch Ereignis
Durch Zeitplan
Auswahl im Menü 'Aktionen'
Einstellungen bearbeiten...
Ziel
Keines
@Befehle dürfen bei diesem Agententyp benutzt werden
Dokumente ändern
Eigenschaften
Ereignisse
Fehler (0 Elemente)
Kiel

LotusScript

- Basic-Dialekt – sehr ähnlich zu Visual Basic
- sehr vielseitig, recht schnell
- Zugriff auf (fast) alles, was Notes kann und bietet
- Zugriff auch auf im Client geöffnete Dokumente und Ansichten (Front-End)
- über OLE/COM Zugriff auf andere Programme (Windows)
- Möglichkeit, C-API und Windows-DLLs zu nutzen
- Möglichkeit via LS2J auf Java-Klassen und -Objekte zuzugreifen

DB mit Server-ID unterzeichnen (lassen)

```
Sub Initialize
        Const PROMPT_CHOOSEDATABASE = 13
        Dim uiws As New NotesUIWorkspace
        Dim session As New NotesSession
        Dim server As String
        Dim filename As String
        Dim adminp As NotesAdministrationProcess
        Dim selectedDBInfos As Variant

        selectedDBInfos = uiws.Prompt(PROMPT_CHOOSEDATABASE, _
          "Sign DB with Server ID", "Choose database to be signed:")
        If Not IsEmpty(selectedDBInfos) Then
            server = selectedDBInfos(0)
            filename = selectedDBInfos(1)
            Set adminp = session.CreateAdministrationProcess(server)
            Call adminp.SignDatabaseWithServerID(server, filename)
        End If
End Sub
```

Java

- sehr vielseitig und schnell
- Zugriff auf (fast) alles, was Notes kann und bietet
- **kein** Zugriff auf im Client geöffnete Dokumente und Ansichten (Front-End) – naja, seit 8.5 ein wenig...
- Möglichkeit, viele existierende APIs zu nutzen, z. B.
  - Schnittstellen zu anderen Systemen,
  - Zugriff auf das Netzwerk,
  - parallele Ausführung mehrerer Aufgaben
- extern erstellen und importieren oder direkt im Domino Designer programmieren

JavaScript

- vor Version 8.5 Standard nur innerhalb von Masken und bei Web-Anwendungen
- jetzt auch in XPages-Anwendungen sowohl im Browser, Client (Client-Side JavaScript = CSJS) und auf dem Server (Server-Side JavaScript = SSJS)
- SSJS wird beim Speichern in Java kompiliert – und dadurch sehr schnell ausgeführt
- SSJS hat Zugriff auf Java-Klassen und -Objekte und enthält viele Elemente, die denen aus LotusScript und Formelsprache nachempfunden wurden

C- und C++-API

- sehr schnelle Ausführung
- voller Zugriff auf alles, was Notes & Domino kann
- auch Server-Erweiterungen möglich
- leider relativ kompliziert und fehleranfällig beim Entwickeln
- plattformabhängig

Fragen?

jetzt stellen – oder später:

tbahn@assono.de
http://www.assono.de/blog
04307/900-401

assono
IT-Consulting & Solutions

Folien unter:
http://www.assono.de/blog/d6plinks/AdminCamp2010-Notes-Entwicklung-fuer-Domino-Admins